**Projekt**

**PROGRAM NAUCZANIA ZAWODU**

**TECHNIK PROGRAMISTA**

opracowany w Ośrodku Rozwoju Edukacji w oparciu o Rozporządzenie Ministra Edukacji Narodowej z dnia 16 maja 2019 r.  
w sprawie podstaw programowych kształcenia w zawodach szkolnictwa branżowego   
oraz dodatkowych umiejętności zawodowych w zakresie wybranych zawodów szkolnictwa branżowego

**program przedmiotowy o strukturze spiralnej**

**SYMBOL CYFROWY ZAWODU 351406**

**KWALIFIKACJE WYODRĘBNIONE W ZAWODZIE:**

INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych

INF.04. Projektowanie, programowanie i testowanie aplikacji

**Warszawa 2019**

Spis treści

[PLAN NAUCZANIA ZAWODU 4](#_Toc18484394)

[WSTĘP DO PROGRAMU 7](#_Toc18484395)

[Opis zawodu 7](#_Toc18484396)

[Charakterystyka programu 8](#_Toc18484397)

[Założenia programowe 9](#_Toc18484398)

[Wykaz przedmiotów z podziałem na kwalifikacje: 9](#_Toc18484399)

[CELE KIERUNKOWE ZAWODU 10](#_Toc18484400)

[PROGRAMY NAUCZANIA DLA POSZCZEGÓLNYCH PRZEDMIOTÓW 11](#_Toc18484401)

[Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych 11](#_Toc18484402)

[Przygotowanie do zawodu programisty 11](#_Toc18484403)

[Bazy danych 22](#_Toc18484404)

[Tworzenie i zarządzanie bazami danych 28](#_Toc18484405)

[Strony i aplikacje internetowe 34](#_Toc18484406)

[Tworzenie stron i aplikacji internetowych 48](#_Toc18484407)

[Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych – klasa II 68](#_Toc18484408)

[Kwalifikacja INF.04. Projektowanie, programowanie i testowanie aplikacji – klasa III i IV 68](#_Toc18484409)

[Język angielski zawodowy 68](#_Toc18484410)

[Kwalifikacja INF.04. Projektowanie, programowanie i testowanie aplikacji 76](#_Toc18484411)

[Podstawy programowania 76](#_Toc18484412)

[Tworzenie aplikacji desktopowych 87](#_Toc18484413)

[Aplikacje desktopowe i mobilne 105](#_Toc18484414)

[Aplikacje webowe 118](#_Toc18484415)

[Tworzenie aplikacji webowych 125](#_Toc18484416)

[Tworzenie aplikacji mobilnych 134](#_Toc18484417)

[Praktyka zawodowa 141](#_Toc18484418)

[PROJEKT EWALUACJI PROGRAMU NAUCZANIA ZAWODU TECHNIK PROGRAMISTA 145](#_Toc18484419)

[ZALECANA LITERATURA DO ZAWODU 154](#_Toc18484420)

PLAN NAUCZANIA ZAWODU z podziałem na przedmioty

# PLAN NAUCZANIA ZAWODU

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nazwa i symbol cyfrowy zawodu: technik programista 351406** | | | | | | | | |
| **Nazwa i symbol kwalifikacji: INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych** | | | | | | | | |
| **Nazwa i symbol kwalifikacji: INF.04. Projektowanie, programowanie i testowanie aplikacji** | | | | | | | | |
| **Lp.** | Kształcenie zawodowe Nazwa przedmiotu (Obowiązkowe zajęcia edukacyjne ustalone przez dyrektora) | Tygodniowy wymiar godzin w klasie | | | | | **Razem  w 5-letnim okresie nauczania** | **Uwagi o realizacji\*** |
| **I** | **II** | **III** | **IV** | **V** |
|  | **Kwalifikacja:0 INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych** | | | | | | | |
|  | Przygotowanie do zawodu programisty |  |  |  |  |  |  | **T** |
|  | Bazy danych |  |  |  |  |  |  | **T** |
|  | Strony i aplikacje internetowe |  |  |  |  |  |  | **T** |
|  | Język angielski zawodowy |  |  |  |  |  |  | **T** |
|  | Tworzenie i zarządzanie bazami danych |  |  |  |  |  |  | **P** |
|  | Tworzenie stron i aplikacji internetowych |  |  |  |  |  |  | **P** |
|  | Razem liczba godzin w kwalifikacji **INF.03.**  : |  |  |  |  |  |  |  |
|  | **Kwalifikacja: INF.04. Projektowanie, programowanie i testowanie aplikacji** | | | | | | | |
|  | Podstawy programowania |  |  |  |  |  |  | **T** |
|  | Aplikacje desktopowe i mobilne |  |  |  |  |  |  | **T** |
|  | Aplikacje webowe |  |  |  |  |  |  | **T** |
|  | Język angielski zawodowy |  |  |  |  |  |  | **T** |
|  | Tworzenie aplikacji desktopowych |  |  |  |  |  |  | **P** |
|  | Tworzenie aplikacji mobilnych |  |  |  |  |  |  | **P** |
|  | Tworzenie aplikacji webowych |  |  |  |  |  |  | **P** |
|  | Razem liczba godzin w kwalifikacji **INF.04.**  : |  |  |  |  |  |  |  |
|  | **Razem** liczba godzin **kształcenia w zawodzie:** |  |  |  |  |  |  |  |
|  | **Praktyka zawodowa** |  |  |  |  |  |  | **Klasa III i IV** |
|  | **Egzamin zawodowy w zakresie kwalifikacji:**  **INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych odbywa się na zakończenie klasy drugiej.**  **INF.04. Projektowanie, programowanie i testowanie aplikacji odbywa się na zakończenie klasy czwartej.** | | | | | | | |

**\*Uwagi o realizacji:**

T - przedmioty w kształceniu zawodowym teoretycznym

P - przedmioty w kształceniu zawodowym organizowane w formie zajęć praktycznych

|  |  |
| --- | --- |
| **„ § 4.** 5. Godziny stanowiące różnicę między sumą godzin obowiązkowych zajęć edukacyjnych z zakresu kształcenia zawodowego określoną w ramowym planie nauczania dla danego typu szkoły a minimalną liczbą godzin kształcenia zawodowego dla kwalifikacji wyodrębnionych w zawodzie określoną w podstawie programowej kształcenia w zawodzie szkolnictwa branżowego przeznacza się na:  1) zwiększenie liczby godzin obowiązkowych zajęć edukacyjnych z zakresu kształcenia w zawodzie lub  2) realizację obowiązkowych zajęć edukacyjnych:  a) przygotowujących uczniów do uzyskania dodatkowych umiejętności zawodowych związanych z nauczanym zawodem, …….. lub  b) przygotowujących uczniów do uzyskania kwalifikacji rynkowej funkcjonującej w Zintegrowanym Systemie Kwalifikacji, związanej z nauczanym zawodem, lub  c) przygotowujących uczniów do uzyskania dodatkowych uprawnień zawodowych przydatnych do wykonywania nauczanego zawodu, lub  d) uzgodnionych z pracodawcą, których treści nauczania ustalone w formie efektów kształcenia są przydatne do wykonywania nauczanego zawodu.”  *Rozporządzenie Ministra Edukacji Narodowej z dnia 3 kwietnia 2019 r. w sprawie ramowych planów nauczania dla publicznych szkół* [*Dz.U. z 2019 r. poz. 639*](http://prawo.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20190000639) | |
|  | |
| *Kompetencje personalne i społeczne* | *Nauczyciele wszystkich obowiązkowych zajęć edukacyjnych z zakresu kształcenia zawodowego powinni stwarzać uczniom warunki do nabywania kompetencji personalnych i społecznych.*  *W programie nauczania zawodu muszą być uwzględnione wszystkie efekty kształcenia z zakresu Kompetencji personalnych i społecznych* |
| *Organizacja pracy małych zespołów* | Nauczyciele wszystkich obowiązkowych zajęć edukacyjnych z zakresu kształcenia zawodowego powinni stwarzać uczniom warunki do nabywania umiejętności w zakresie organizacji pracy małych zespołów.  *W programie nauczania zawodu muszą być uwzględnione wszystkie efekty kształcenia z zakresu* |

# WSTĘP DO PROGRAMU

## Opis zawodu

Programista to osoba, która tworzy programy komputerowe w pewnym języku programowania oraz nadzoruje proces ich wdrażania. Zazwyczaj programiściznają co najmniej kilka języków programowania (np. C, C++, Java, Python, PHP, JavaScript), lecz specjalizują się tylko w wybranym z nich. Dla podkreślenia specjalizacji nazwa głównego języka jest dodawana do nazwy stanowiska, np. *programista Java*.

Współcześni programiści pracują najczęściej w biurach lub laboratoriach komputerowych wyposażonych w sprzęt niezbędny do testowania aplikacji, często też podróżują, by kontaktować się z klientami. Wykorzystują telekonferencje oraz pracę zdalną, ponieważ wiele zadań zawodowych może być wykonywanych bez konieczności przebywania w miejscu pracy.

Programiści, w zależności od specjalizacji, dzielą się na programistów:

* aplikacji, zajmujących się tworzeniem aplikacji komputerowych realizujących określone zadanie, np. wspomaganie zarządzania przedsiębiorstwem,
* systemowi, rozwijający aplikacje oraz systemy nadzorujące pracę sprzętu komputerowego, np. systemy operacyjne, sterowniki, czy systemy zarządzania bazami danych.
* aplikacji internetowych działających w środowisku www,
* aplikacji mobilnych, działających na urządzeniach przenośnych, takich jak telefony komórkowe, smartfony, palmtopy czy tablety.

Zawód technik programista, o symbolu cyfrowym zawodu 351406, należy do branży teleinformatycznej. Według Polskiej Ramy Kwalifikacji (PRK), będącej układem odniesienia dla kwalifikacji nadawanych w Polsce i zawierającej 8 poziomów, ma przypisany dla kwalifikacji pełnej V poziom PRK, a dla obu kwalifikacji cząstkowych wyodrębnionych w zawodzie INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych oraz INF.04. Projektowanie, programowanie i testowanie aplikacji – 5. poziom PRK. Każdy poziom opisywany jest za pomocą ogólnych charakterystyk zakresu i stopnia skomplikowania wiedzy, umiejętności i kompetencji społecznych, wymaganych od osób posiadających kwalifikacje danego poziomu. Polska Rama Kwalifikacji pozwala na odniesienie polskich kwalifikacji do poziomów Europejskiej Ramy Kwalifikacji (ERK) i poprzez ERK do poziomów kwalifikacji w poszczególnych państwach UE.

Kształcenie w zawodzie technik programista [351406] może się odbywać w technikum (pięcioletnim po szkole podstawowej i czteroletnim po gimnazjum). Istnieje możliwość zdobywania wiedzy, umiejętności i kompetencji społecznych z zakresu zawodu technik programista na kwalifikacyjnych kursach zawodowych lub na kursach umiejętności zawodowych.

Absolwent szkoły prowadzącej kształcenie w zawodzie technik programista powinien być przygotowany do wykonywania zadań zawodowych:

1) w zakresie kwalifikacji **INF.03.** Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych:

a) tworzenia i administrowania stronami internetowymi,

b) tworzenia, administrowania i użytkowania relacyjnych baz danych,

c) programowania aplikacji internetowych,

d) tworzenia i administrowania systemami zarządzania treścią;

2) w zakresie kwalifikacji **INF.04.** Projektowanie, programowanie i testowanie aplikacji:

a) projektowania, programowania i testowania zaawansowanych aplikacji webowych,

b) projektowania, programowania i testowania aplikacji desktopowych,

c) projektowania, programowania i testowania aplikacji mobilnych.

Technik programista jest zawodem dwu-kwalifikacyjnym, więc w cyklu kształcenia przewidywane są dwa egzaminy zawodowe. Pierwszy będzie dotyczył tworzenia i administrowania stronami i aplikacjami internetowymi oraz bazami danych. Drugi umożliwi uczniom sprawdzenie wiedzy i umiejętności w zakresie projektowania, programowania i testowania aplikacji. Uzyskanie pozytywnego wyniku w obu kwalifikacjach będzie równoznaczne z uzyskaniem tytułu technika programisty. Egzamin zawodowy w kwalifikacji INF.03 przewidziany jest pod koniec drugiej klasy, w drugiej kwalifikacji INF.04 – pod koniec klasy czwartej.

Dla zawodu technik programista podstawia programowa określa możliwe do zdobycie przez ucznia dodatkowe umiejętności zawodowe związane z nauczanym zawodem:

1. Bezpieczeństwo sieci komputerowych
2. Bezpieczeństwo systemów komputerowych
3. Budowa i konfiguracja sieci komputerowych
4. Eksploatacja baz danych
5. Grafika 3D i wydruk 3D
6. Programowanie mikrokontrolerów i prostych układów
7. Programowanie w języku Python
8. Serwis urządzeń techniki komputerowej
9. Tworzenie i testowanie aplikacji

Absolwent szkoły prowadzącej kształcenie w zawodzie technik programista, mający potwierdzoną kwalifikację INF.03.Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych, dodatkowo po potwierdzeniu kwalifikacji INF.02. Administracja i eksploatacja systemów komputerowych, urządzeń peryferyjnych i lokalnych sieci komputerowych może uzyskać dyplom zawodowy w zawodzie technik informatyk.

## Charakterystyka programu

Nazwa programu: PROGRAM NAUCZANIA ZAWODU **TECHNIK PROGRAMISTA**,

Rodzaj programu: program przedmiotowy

Struktura treści: spiralna,

Typ szkoły: technikum,

Symbol cyfrowy zawodu: 351406

Kwalifikacje wyodrębnione w zawodzie:

INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych,

INF.04. Projektowanie, programowanie i testowanie aplikacji

Program nauczania dla zawodu technik programista [351406] jest programem przedmiotowym o strukturze spiralnej. Opracowany został dla technikum pięcioletniego. Obejmuje 1470 godzin kształcenia zawodowego, które w części określonej podstawą programową zostaje zakończone w klasie czwartej, dzięki czemu istnieje łatwość dopasowania go do technikum czteroletniego.

Program nauczania uwzględnia wszystkie treści z podstawy programowej kształcenia w zawodzie. Treści nauczania wynikające z oczekiwanych efektów kształcenia realizowanych w ramach INF.03.2 Podstawy informatyki nie są powtarzane w kwalifikacji INF.04.

Programista musi być dokładny, skupiony na powierzonym zadaniu, systematyczny, samodzielny, ale jednocześnie dobrze współpracujący w grupie, pasjonujący się programowaniem, odporny na stres, stale się doskonalący, zdobywający nową wiedzę i poznający nowe technologie, takie więc cechy osobowości powinien mieć kandydat do tego zawodu, a także nauczyciel wdrażający program.

Dwie jednostki efektów kształcenia:

* INF.04.10. Kompetencje personalne i społeczne,
* INF.04.11. Organizacja pracy małych zespołów

powinny być realizowane przez nauczycieli wszystkich obowiązkowych zajęć edukacyjnych z zakresu kształcenia zawodowego.

## Założenia programowe

Firmy informatyczne wciąż mają problem z brakiem wykwalifikowanych specjalistów w dziedzinie programowania. Ministerstwo Nauki i Szkolnictwa Wyższego podało, że w latach 2010–2025 będą oni najbardziej poszukiwaną grupą spośród zawodów technicznych. W 2016 roku liczbę brakujących na rynku pracy specjalistów z branży IT, w tym programistów, szacowano na ok. 50 tys. osób. (źródło Wikipedia) Kształcenie w zawodzie technik programista jest więc niezbędne i oczekiwane przez rynek pracy.

## Wykaz przedmiotów z podziałem na kwalifikacje:

**Kwalifikacja INF.03.** Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych

**Kształcenie teoretyczne**

* Przygotowanie do zawodu programisty
* Bazy danych
* Strony i aplikacje internetowe
* Język angielski zawodowy

**Kształcenie praktyczne**

* Tworzenie i zarządzanie bazami danych
* Tworzenie stron i aplikacji internetowych

**Kwalifikacja INF.04.** Projektowanie, programowanie i testowanie aplikacji

**Kształcenie teoretyczne**

* Podstawy programowania
* Aplikacje desktopowe i mobilne
* Aplikacje webowe
* Język angielski zawodowy

**Kształcenie praktyczne**

* Tworzenie aplikacji desktopowych
* Tworzenie aplikacji mobilnych
* Tworzenie aplikacji webowych

**Przedmiot wspólny dla obu kwalifikacji: INF.03.** Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych oraz **INF.04.** Projektowanie, programowanie i testowanie aplikacji

**Praktyka zawodowa**

# CELE KIERUNKOWE ZAWODU

Absolwent szkoły prowadzącej kształcenie w zawodzie technik proramista powinien być przygotowany do wykonywania zadań zawodowych w zakresie:

1. tworzenia stron i aplikacji internetowych,
2. tworzenia i zarządzania bazami danych,
3. tworzenia aplikacji desktopowych,
4. tworzenia aplikacji mobilnych,
5. testowania i dokumentowania aplikacji.

# PROGRAMY NAUCZANIA DLA POSZCZEGÓLNYCH PRZEDMIOTÓW

# Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych

## Przygotowanie do zawodu programisty

**Cele ogólne**

1. Poznanie pojęć związanych z bezpieczeństwem i higieną pracy, ochroną przeciwpożarową, ochroną środowiska i ergonomią.
2. Nabywanie umiejętności stosowania zasad bezpieczeństwa i higieny pracy, ochrony przeciwpożarowej, ochrony środowiska i ergonomii.
3. Nabywanie umiejętności stosowania wiedzy związanej z prawną ochroną pracy.
4. Nabywanie umiejętności określania zagrożeń dla zdrowia i życia człowieka oraz mienia i środowiska oraz sposobów przeciwdziałania zagrożeniom podczas wykonywania zadań zawodowych.
5. Kształtowanie umiejętności identyfikowania czynników niebezpiecznych, szkodliwych i uciążliwych podczas wykonywania zadań zawodowych.
6. Doskonalenie umiejętności udzielania pierwszej pomocy poszkodowanym podczas wykonywania zadań zawodowych.
7. Poznanie pojęć związanych z siecią komputerową.
8. Poznanie parametrów sprzętu komputerowego.
9. Nabywanie umiejętności definiowania elementów architektury systemów komputerowych.
10. Nabywanie umiejętności charakteryzowania systemów informatycznych.
11. Kształtowanie umiejętności identyfikowania ułatwień dostępności serwisów internetowych dla osób niepełnosprawnych.
12. Kształtowanie umiejętności posługiwania się systemami liczbowymi stosowanymi w informatyce.
13. Kształtowanie umiejętności stosowania zasad cyberbezpieczeństwa.

**Cele operacyjne**

1. rozróżniać pojęcia związane z bezpieczeństwem i higieną pracy, ochroną przeciwpożarową, ochroną środowiska i ergonomią,
2. stosować zasady dotyczące bezpieczeństwa i higieny pracy, ochrony przeciwpożarowej, ochrony środowiska i ergonomii,
3. rozróżniać zadania i uprawnienia instytucji działających w zakresie ochrony pracy i ochrony środowiska w Polsce,
4. rozróżniać zadania i uprawnienia służb działających w zakresie ochrony pracy i ochrony środowiska w Polsce,
5. określić prawa i obowiązki pracownika w zakresie bezpieczeństwa i higieny pracy,
6. określić prawa i obowiązki pracodawcy w zakresie bezpieczeństwa i higieny pracy,
7. rozróżniać czynniki niebezpieczne w środowisku pracy,
8. scharakteryzować czynniki niebezpieczne w środowisku pracy,
9. rozróżniać czynniki szkodliwe w środowisku pracy,
10. scharakteryzować czynniki szkodliwe w środowisku pracy,
11. rozróżniać czynniki uciążliwe w środowisku pracy,
12. scharakteryzować czynniki uciążliwe w środowisku pracy,
13. rozróżniać środki ochrony indywidualnej podczas wykonywania prac zawodowych,
14. rozróżniać środki ochrony zbiorowej podczas wykonywania prac zawodowych,
15. scharakteryzować środki ochrony indywidualnej podczas wykonywania prac zawodowych,
16. scharakteryzować środki ochrony indywidualnej podczas wykonywania prac zawodowych,
17. dobrać środki ochrony indywidualnej podczas wykonywania prac zawodowych,
18. dobrać środki ochrony indywidualnej podczas wykonywania prac zawodowych,
19. określić zasady udzielania pierwszej pomocy,
20. zastosować zasady udzielania pierwszej pomocy,
21. udzielić pierwszej pomocy poszkodowanym w wypadkach przy pracy oraz w stanach zagrożenia zdrowia i życia,
22. przewidywać zagrożenia dla zdrowia i życia człowieka oraz mienia i środowiska związanych z wykonywaniem zadań zawodowych,
23. określić parametry urządzeń techniki komputerowej,
24. przeliczyć jednostki pojemności pamięci,
25. dobrać urządzenia techniki komputerowej zgodnie z wymaganiami technicznymi stanowiska,
26. opisać zasadę działania procesora,
27. identyfikować system informatyczny,
28. identyfikować systemy przetwarzania informacji,
29. scharakteryzować miejsca przechowywania informacji,
30. dobrać systemy informatyczne do określonych zadań,
31. wskazać przykłady systemów informatycznych stosowanych w biznesie,
32. scharakteryzować funkcje portali społecznych,
33. określić bezpieczne zasady korzystania z portali społecznościowych,
34. zastosować zasady netykiety podczas korzystania z zasobów Internetu,
35. scharakteryzować dostępne udogodnienia dla osób niepełnosprawnych,
36. rozróżniać pojęcia dotyczące sieci komputerowej,
37. scharakteryzować topologie sieci,
38. identyfikować cechy modelu TCP/IP i protokołów komunikacji,
39. scharakteryzować sieć synchroniczną i asynchroniczną,
40. opisywać sieć przewodową i bezprzewodową,
41. zastosować oprogramowanie do monitorowania sieci,
42. scharakteryzować komunikatory sieciowe,
43. scharakteryzować poszczególne pozycyjne systemy liczbowe: BIN, OCT, HEX,
44. zapisywać liczby w wybranym systemie liczbowym,
45. zapisywać liczby w kodzie uzupełnieniowym do dwóch,
46. wykonywać działania arytmetyczne i logiczne na liczbach binarnych,
47. rozróżnić rodzaje ataków hackerskich,
48. rozróżnić rodzaje szkodliwego oprogramowania,
49. stosować zasady bezpiecznego przechowywania danych,
50. stosować zasady bezpiecznego dokonywania transakcji w sieci,
51. zastosować zasady prywatności w cyfrowym świecie,
52. korzystać ze źródeł informacji dotyczących norm międzynarodowych, europejskich i krajowych stosowanych w informatyce.

**MATERIAŁ NAUCZANIA PRZYGOTWANIE DO ZAWODU PROGRAMISTY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dział programowy | Tematy jednostek metodycznych | Liczba godz. | Wymagania programowe | | Uwagi o realizacji |
| Podstawowe  **Uczeń potrafi:** | Ponadpodstawowe  **Uczeń potrafi:** | Etap realizacji |
| I. Bezpieczeństwo i higiena pracy – wprowadzenie | Podstawowe informacje o bezpieczeństwie i higienie pracy. Instytucje oraz służby w zakresie prawa pracy i ochrony środowiska | 5 | * wymienić pojęcia związane z bezpieczeństwem i higieną pracy, * wymienić zasady dotyczące ochrony przeciwpożarowej, * wyjaśnić zasady dotyczące ochrony przeciwpożarowej, * wymienić środki gaśnicze, * scharakteryzować środki gaśnicze, * rozróżnić środki gaśnicze, * wymienić instytucje oraz służby z zakresu ochrony pracy i ochrony środowiska, * rozróżnić instytucję oraz służby z zakresu ochrony pracy i ochrony środowiska, | * zastosować zasady dotyczące ochrony przeciwpożarowej, * dobierać środki gaśnicze, * rozróżniać dokumenty dotyczące przepisów bezpieczeństwa i higieny pracy oraz ochrony przeciwpożarowej i ochrony środowiska, * wymienić zadania instytucji oraz służb z zakresu ochrony pracy i ochrony środowiska. | Klasa I |
| Podstawy ergonomii oraz ochrona środowiska naturalnego | 5 | * wyjaśnić pojęcie ergonomii, * wymienić sposoby organizowania stanowiska pracy zgodnie z obowiązującymi wymaganiami ergonomii, * wymienić sposoby organizowania stanowiska pracy zgodnie z przepisami bezpieczeństwa i higieny pracy, * wymienić sposoby organizowania stanowiska pracy zgodnie z przepisami ochrony przeciwpożarowej i ochrony środowiska. | * zorganizować stanowisko pracy zgodnie z zasadami ergonomii, * zorganizować stanowisko pracy zgodnie z przepisami bezpieczeństwa i higieny pracy, * zorganizować stanowisko pracy zgodnie z przepisami ochrony przeciwpożarowej i ochrony środowiska. | Klasa I |
| 2. Prawna ochrona pracy | Prawa i obowiązki pracownika oraz pracodawcy | 5 | * wymienić podstawowe akty prawne w zakresie praw i obowiązków pracownika i pracodawcy, * wymienić prawa pracownika w zakresie bezpieczeństwa i higieny pracy, * wymienić obowiązki pracownika w zakresie bezpieczeństwa i higieny pracy, * wymienić prawa pracodawcy w zakresie bezpieczeństwa i higieny pracy, * wymienić obowiązki pracodawcy w zakresie bezpieczeństwa i higieny pracy * scharakteryzować prawa i obowiązki pracownika w zakresie bezpieczeństwa i higieny pracy, * scharakteryzować prawa i obowiązki pracodawcy w zakresie bezpieczeństwa i higieny pracy. | * określić zakres odpowiedzialności pracodawcy i pracownika, * podać przykłady regulacji w opracowywaniu regulaminów, układów zbiorowych pracy w części dotyczącej warunków pracy instrukcji obsługi, | Klasa I |
| Zagrożenia na stanowisku pracy. Pierwsza pomoc | 5 | * wymienić zagrożenia dla zdrowia i życia związane z wykonywaniem zadań zawodowych, * wymienić zagrożenia mienia i środowiska związane z wykonywaniem zadań zawodowych, * charakteryzować zagrożenia dla zdrowia i życia związane z wykonywaniem zadań zawodowych, * charakteryzować zagrożenia mienia i środowiska związane z wykonywaniem zadań zawodowych, * dobrać sposoby przeciwdziałania zagrożeniom mienia i środowiska, związane z wykonywaniem zadań zawodowych, * wymienić zasady udzielania pierwszej pomocy, * wyjaśnić zasady udzielania pierwszej pomocy, * ocenić stan poszkodowanego. | * ocenić zagrożenia dla zdrowia i życia człowieka związane z wykonywaniem zadań zawodowych, * ocenić zagrożenia mienia i środowiska związane z wykonywaniem zadań zawodowych, * udzielić poszkodowanemu pierwszej pomocy. | Klasa I |
| Czynniki szkodliwe w środowisku pracy | 5 | * wyjaśnić pojęcie czynników fizycznych, * wymienić czynniki fizyczne, * zdefiniować pojęcie czynników chemicznych, * wymienić czynniki chemiczne, * wyjaśnić pojęcie czynników biologicznych, * wymienić czynniki biologiczne, * wyjaśnić pojęcie czynników psychofizycznych, * wymienić czynniki psychofizyczne, * wyjaśnić pojęcie czynników uciążliwych, * wymienić czynniki uciążliwe, * rozróżniać czynniki fizyczne, chemiczne, biologiczne, psychofizyczne występujące na stanowisku pracy, * rozróżniać czynniki uciążliwe występujące na stanowisku pracy. | * dobierać sposoby przeciwdziałania czynnikom fizycznym, biologicznym, chemicznym, psychofizycznym i uciążliwym, występującym na stanowisku pracy, * ocenić skutki oddziaływania czynników fizycznych, chemicznych, biologicznych, psychofizycznych i uciążliwych na organizm człowieka. | Klasa I |
| Środki ochrony indywidulanej i zbiorowej | 5 | * wyjaśnić pojęcie ochrony indywidualnej i zbiorowej, * wymienić środki ochrony indywidualnej i zbiorowej, * charakteryzować środki ochrony indywidualnej i zbiorowej, * rozróżniać środki ochrony indywidualnej i zbiorowej. | * dobrać środki ochrony indywidualnej i zbiorowej do określonych prac. | Klasa I |
| III. Podstawowe zagadnienia związane z informatyką. | 1. Urządzenia techniki komputerowej | 6 | - określać parametry urządzeń techniki komputerowej,  - dobierać urządzenia techniki komputerowej zgodnie z wymaganiami technicznymi stanowiska, | - opisać zasadę działania procesora,  - przeliczać jednostki pojemności pamięci,  - dobierać systemy informatyczne do określonych zadań, | Klasa I |
| 2. Systemy informatyczne | 6 | - identyfikować system informatyczny,  - identyfikować systemy przetwarzania informacji,  - charakteryzować miejsca przechowywania informacji,  - wskazać przykłady systemów informatycznych stosowanych w biznesie,  - charakteryzować funkcje portali społecznych,  charakteryzować dostępne udogodnienia dla osób niepełnosprawnych, | - dobierać systemy informatyczne do określonych zadań,  - określać bezpieczne zasady korzystania z portali społecznościowych,  - stosować zasady netykiety podczas korzystania z zasobów Internetu, | Klasa I |
| 3. Sieć komputerowa | 6 | - rozróżniać pojęcia dotyczące sieci komputerowej,  - charakteryzować topologie sieci,  - charakteryzować sieć synchroniczną i asynchroniczną,  - opisywać sieć przewodową i bezprzewodową,  - charakteryzować komunikatory sieciowe, | - identyfikować cechy modelu TCP/IP i protokołów komunikacji,  - stosować oprogramowanie do monitorowania sieci, | Klasa I |
| 4. Liczbowe system pozycyjne stosowane w informatyce. | 6 | - charakteryzować poszczególne pozycyjne systemy liczbowe: BIN, OCT, HEX,  - zapisywać liczby w wybranym systemie liczbowym, | - zapisywać liczby w kodzie uzupełnieniowym do dwóch,  - wykonywać działania arytmetyczne na liczbach binarnych,  - wykonywać działania logiczne na liczbach binarnych, | Klasa I |
| 5. Cyberbezpieczeństwo | 6 | - rozróżniać rodzaje ataków hackerskich,  - rozróżniać rodzaje szkodliwego oprogramowania,  - korzystać ze źródeł informacji dotyczących norm międzynarodowych, europejskich i krajowych stosowanych w informatyce. | - stosować zasady bezpiecznego przechowywania danych,  - stosować zasady bezpiecznego dokonywania transakcji w sieci,  - stosować zasady prywatności w cyfrowym świecie, | Klasa I |
| **RAZEM** | | **60** |  |  |  |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągnięcia założonych celów kształcenia w zakresie przedmiotu bezpieczeństwo i higiena pracy jest opracowanie odpowiednich procedur, w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych, jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie liczby osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności, w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

**METODY NAUCZANIA**

Dla przedmiotu Przygotowanie do zawodu programisty, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych, takich jak:

* wykład informacyjny,
* pokaz z objaśnieniem,
* wykład problemowy,
* metoda przypadku,
* dyskusja dydaktyczna,
* burza mózgów.

Zajęcia mogą także odbywać się w grupach. Dominującą metodą kształcenia powinna być metoda tekstu przewodniego (ułatwi uczniom samodzielne zbieranie i analizowanie informacji) oraz metoda przypadku, polegająca na analizowaniu przypadku opisującego problem.

**ŚRODKI DYDAKTYCZNE**

Pracownia, w której prowadzone będą zajęcia, powinna być wyposażona w: zbiory przepisów prawa z zakresu bezpieczeństwa i higieny pracy, kodeks pracy, filmy i prezentacje multimedialne (dotyczące przepisów w zakresie bezpieczeństwa i higieny pracy, ochrony przeciwpożarowej, ochrony środowiska i ergonomii), komputer z dostępem do internetu, urządzenia multimedialne, głośniki.

**FORMY ORGANIZACYJNE**

Zajęcia powinny być prowadzone z wykorzystaniem różnych form organizacyjnych: indywidualnie i zespołowo. Bardzo ważną kwestią w kształceniu zawodowym jest indywidualizacja pracy w kierunku potrzeb i możliwości ucznia w zakresie metod, środków oraz form kształcenia. Nauczyciel realizujący program powinien:

* motywować uczniów do pracy,
* dostosowywać stopień trudności planowanych ćwiczeń do możliwości i potrzeb uczniów,
* planować zadania do wykonywania przez uczniów z uwzględnieniem ich zainteresowań,
* przygotowywać zadania o różnym stopniu trudności i złożoności,
* zachęcać uczniów do korzystania z różnych źródeł informacji zawodowej.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Testy wielokrotnego wyboru, testy zawierające zadania otwarte, odpowiedzi ustne, prezentacje uczniów.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu, na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* ustnych sprawdzianów poziomu wiedzy i umiejętności,
* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń,
* projektu i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjach zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest, aby także uczniowie dokonywali samooceny własnej i kolegów z zespołu pracy, wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

**Przykładowe zadania dla uczniów:**

Zadanie 1

Dobierz środki ochrony indywidualnej i zbiorowej na stanowisku technika automatyka. Uzasadnij swój wybór.

Zadanie 2

Podaj obowiązki pracownika w zakresie bezpieczeństwa i higieny pracy wynikające z Kodeksu Pracy.

Zadanie 3

Podaj obowiązki pracodawcy w zakresie bezpieczeństwa i higieny pracy wynikające z Kodeksu Pracy.

Zadanie 4

Podaj zasady udzielania pierwszej pomocy w przypadku porażenia prądem elektrycznym.

Zadanie 5

Wymień zasady ergonomii pracy przy stanowisku komputerowym.

Zadanie 6

Zapisz swoją datę urodzenia czyli dzień, miesiąc, rok w binarnym systemie liczbowym.

Zadanie 7

Podaj przykłady komunikatorów sieciowych. Z podanych przykładów wskaż ten który najlepiej nadaje się do komunikacji audio.

Zadanie 8

Scharakteryzuj topologię sieci zastosowaną w pracowni szkolnej.

**EWALUACJA PRZEDMIOTU**

Strategia przeprowadzanej ewaluacji będzie polegać na tzw. twardej analizie danych, którymi będą oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych. Zebrane dane zostaną poddane analizie ilościowej i jakościowej przy użyciu narzędzia, którym jest statystyka matematyczna.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego. Spowoduje to podwyższenie jakości kształcenia i znacząco wpłynie na indywidualne wyniki uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. stosowania przepisów bezpieczeństwa i higieny pracy w wykonywaniu zadań zawodowych,
2. udzielania pierwszej pomocy poszkodowanemu,
3. stosowania środków ochrony indywidualnej i zbiorowej podczas wykonywania zadań zawodowych,
4. analizy zagrożeń występowania czynników szkodliwych w środowisku pracy,
5. zapisania liczb w wybranych pozycyjnych systemach liczbowych stosowanych w informatyce,
6. stosowania pojęć związanych z urządzeniami techniki komputerowej, systemami informatycznymi i sieciami komputerowymi,
7. stosować zasady bezpiecznego korzystania z komputera podłączonego do sieci lokalnej i Internetu.

**ZALECANA LITERATURA**

Proponowane podręczniki:

1. Krzysztof Szczęch, Wanda Bukała, Bezpieczeństwo i higiena pracy, Podręcznik do kształcenia zawodowego. WSiP. Warszawa 2016.
2. Marcin Czerwonka, Zenon Nowocień Kwalifikacja INF.02. Administracja i eksploatacja systemów komputerowych, urządzeń peryferyjnych i lokalnych sieci komputerowych. Część 1. Systemy komputerowe. Podręcznik do nauki zawodu technik informatyk, wyd. Helion,

## Bazy danych

**Cele ogólne przedmiotu**

* 1. Poznanie podstawowych pojęć dotyczących baz danych;
  2. Poznanie systemów zarządzania bazami danych;
  3. Poznanie typów danych oraz atrybutów kolumn stosowanych w bazach danych;
  4. Poznanie poleceń strukturalnego języka zapytań SQL;
  5. Nabycie umiejętności budowania diagramów E/R;
  6. Nabycie umiejętności analizowania struktury bazy danych;
  7. Nabycie umiejętności projektowania relacyjnych baz danych;
  8. Nabycie umiejętności tworzenia zapytań w języku SQL;
  9. Nabycie umiejętności stosowania funkcji w strukturalnym języku zapytań SQL;
  10. Nabycie umiejętności tworzenia i modyfikowania tabel w bazach danych;
  11. Kształtowanie postawy
* aktywnego słuchania i prowadzenia dyskusji;
* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
* odpowiedzialności za podejmowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;

**Cele operacyjne**

**Uczeń potrafi:**

1. stosować podstawowe pojęcia dotyczące baz danych: encja, atrybut, krotka, relacja, klucz główny, klucz obcy, tabela, baza danych, relacja,
2. rozróżniać systemy zarządzania bazami danych: MySQL, MS SQL Server, PostgreSQL, DB2, Oracle, Microsoft Access,
3. uzasadnić wybór systemu zarządzania bazami danych do konkretnego projektu bazy danych,
4. scharakteryzować typy danych stosowane w bazach danych: char, varchar, text, int, float, double, blob, date, datetime, time,
5. opisać bloki występujące w diagramie E/R,
6. wykonać diagram E/R na podstawie opisu relacyjnego modelu bazy danych,
7. omówić polecenia i operatory strukturalnego języka zapytań wykorzystywane przy pobieraniu danych z bazy: select, distinc, where, order by, and, or, in, between, not, =, !=, <>, >, >=, <, <=,
8. omówić polecenia służące do tworzenia i modyfikowania baz danych: create, insert, delete, update, drop, alter, not null, select into, default, unique,
9. omówić funkcje dostępne w strukturalnym języku zapytań: avg, count, max, min, sum, ucase, lcase, length, replace, round, substring, truncate,
10. uzasadnić składnię polecenia select do pobrania wskazanych danych z bazy danych,
11. dobrać polecenia do tworzenia i modyfikacji tabel w bazie danych,
12. uzasadnić wybór funkcji do wykonania operacji na danych pobranych z bazy danych,
13. wykonać projekt relacyjnej bazy danych.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Podstawy baz danych** | 1. Podstawowe pojęcia związane z bazami danych. | 12 | - stosować podstawowe pojęcia dotyczące baz danych: encja, atrybut, krotka, relacja, klucz główny, klucz obcy, tabela, baza danych, relacja,  - scharakteryzować typy danych stosowane w bazach danych: char, varchar, text, int, float, double, blob, date, datetime, time, | - podać przykłady encji na podstawie obiektów rzeczywistych  - dobrać odpowiednie atrybuty do encji,  - wskazać klucz podstawowy i klucz obcy w tabeli, | Klasa I |
| 2. Relacyjny model baz danych. | 9 | - opisać bloki występujące diagramie diagramie E/R, | - wykonać diagram E/R na podstawie opisu relacyjnego modelu bazy danych, |  |
| **II. Podstawy języka SQL** | 1. Systemy zarządzania bazami danych. | 3 | - rozróżniać systemy zarządzania bazami danych: MySQL, MS SQL Server, PostgreSQL, DB2, Oracle, Microsoft Access, | - uzasadnić wybór systemu zarządzania bazami danych do konkretnego projektu bazy danych, |  |
| 2. Polecenia SQL. | 21 | - omówić polecenia i operatory strukturalnego języka zapytań wykorzystywane przy pobieraniu danych z bazy: select, distinc, where, order by, and, or, in, between, not, =, !=, <>, >, >=, <, <=,  - omówić polecenia służące do tworzenia i modyfikowania baz danych: create, insert, delete, update, drop, alter, not null, select into, default, unique,  - omówić funkcje dostępne w strukturalnym języku zapytań: avg, count, max, min, sum, ucase, lcase, length, replace, round, substring, truncate, | - uzasadnić składnię polecenia select do pobrania wskazanych danych z bazy danych;  - dobrać atrybuty do wybranego rodzaju zapytania,  - wskazać funkcje do wykonania operacji np. sumowania podczas wykonywania zapytania do bazy danych | Klasa I |
| **III. Tworzenie baz danych** | 1. Tworzenie baz danych. | 6 | dobrać polecenia do tworzenia i modyfikacji tabel w bazie danych, | zdefiniować strukturę bazy danych, | Klasa I |
| 2. Tworzenie zapytań do bazy danych. | 3 | stosować podstawową strukturę zapytania języka SQL select-from-where, | uzasadnić wybór funkcji języka SQL do wykonania operacji na danych pobranych z bazy danych, |
| 3. Analiza struktury bazy danych | 3 | opisuje elementy występujące w przykładowej strukturze bazy danych, | podaje składnię poleceń SQL umożliwiających modyfikację przykładowej struktury bazy danych, |
| 4. Uprawnienia użytkowników | 3 | wymienia uprawnienia użytkowników w systemie zarządzania bazami danych, | dobiera uprawnienia użytkownika w systemie zarządzania bazami danych, |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągnięcia założonych celów kształcenia w zakresie przedmiotu jest opracowanie odpowiednich dla danego zawodu procesu, a w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych, jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

**METODY NAUCZANIA**

Dla przedmiotu Bazy danych, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych takich jak:

* wykład informacyjny,
* pokaz z objaśnieniem,
* wykład problemowy,
* dyskusja dydaktyczna,
* burza mózgów,
* ćwiczenia praktyczne,

Zajęcia mogą także odbywać się w grupach. Dominująca metodą kształcenia powinny być także ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku, polegająca na analizowaniu przypadku opisującego problem. W trakcie realizacji zajęć nauczyciel powinien:

– motywować uczniów do pracy,

– dostosowywać stopień trudności planowanych ćwiczeń do możliwości uczniów,

– uwzględniać zainteresowania uczniów,

– przygotowywać zadania o różnym stopniu trudności i złożoności,

– zachęcać uczniów do korzystania z różnych źródeł informacji zawodowej,

– stosować metody aktywizujące,

– stosować nowoczesne środki kształcenia, np. tablice multimedialne.

**ŚRODKI DYDAKTYCZNE**

W pracowni, w której prowadzone będą zajęcia powinny się znajdować: plansze z blokami diagramów E/R, przykładowe diagramy E/R, plansze z poleceniami SQL, plansze z typowymi zapytaniami w języku SQL, literatura dotycząca baz danych, komputer wyposażony w oprogramowanie do zarzadzania bazami danych, dostęp do Internetu, baza danych dostępna online, projektor, tablica interaktywna, drukarka.

**FORMY ORGANIZACYJNE**

Zajęcia powinny być prowadzone z wykorzystaniem różnych form organizacyjnych: indywidualnie i zespołowo. W przypadku realizacji ćwiczeń praktycznych powinny być stosowane formy organizacyjne indywidualne. Bardzo ważną kwestią w kształceniu zawodowym jest indywidualizacja pracy w kierunku potrzeb i możliwości ucznia w zakresie metod, środków oraz form kształcenia. Ponadto uczniowie powinni samodzielnie budować swoją wiedzę i kształtować umiejętności poprzez uczenie się we współpracy oraz korzystanie z różnych źródeł informacji.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Metody sprawdzania osiągnięć edukacyjnych ucznia: testy wielokrotnego wyboru, testy zawierające zadania otwarte, odpowiedzi ustne, prezentacje uczniów.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu, na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* ustnych sprawdzianów, sprawdzających poziom wiedzy i umiejętności,
* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń praktycznych,
* projektu bazy danych i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest, aby uczniowie dokonywali także własnej samooceny pracy i kolegów z zespołu, wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

Zadanie 1

Zapisz składnię polecenia SELECT pobierającego z bazy danych szkoła z tabeli uczniowie wszystkich uczniów (id, imie, nazwisko, data\_ur, id\_k). Wiedząc, że id\_k jest kluczem obcym i łączy tabelę uczniowie z tabelą klasy relacją wiele do jednego zmodyfikuj zapytanie SELECT tak aby pobrało z bazy uczniów wybranej klasy.

Zadanie 2

Twoim zadaniem jest utworzenie bazy danych warsztat, która będzie zawierać następujące tabele: pracownicy, części, narzędzia, usługi oraz klienci. Wypisz polecenia w odpowiedniej kolejności umożliwiające utworzenie wyżej wymienionej struktury bazy danych.

Zadanie 3

Scharakteryzuj uprawnienia użytkowników w systemie zarządzania baz danych.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Strategia przeprowadzanej ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych. Zebrane dane zostaną poddane analizie ilościowej i jakościowej przy użyciu narzędzia, którym jest statystyka matematyczna. Przydatnym narzędziem w tej analizie może być na przykład korzystanie z platformy office365.com lub podobnej, która daje możliwość analizy, które z pytań testowych sprawiają trudność.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego. Spowoduje to podwyższenie jakości kształcenia i znacząco wpłynie na indywidualne wyniki uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. stosowania podstawowych pojęć z zakresu baz danych,
2. znajomość podstawowych poleceń języka SQL
3. formułowania zapytań do bazy danych SELECT – FROM - WHERE,
4. tworzenie bazy danych zgodnie z projektem.

**ZALECANA LITERATURA**

Jolanta Pokorska, Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych. Część 2. Projektowanie i administrowanie bazami danych. Podręcznik do nauki zawodu technik informatyk i technik programista, wyd. Helion,

## Tworzenie i zarządzanie bazami danych

**Cele ogólne przedmiotu**

1. Nabycie umiejętności instalacji, konfiguracji i zarządzania systemem baz danych;
2. Nabycie umiejętności tworzenia i modyfikowania struktury bazy danych przy użyciu języka zapytań;
3. Nabycie umiejętności tworzenia, modyfikowania i usuwania tabel w bazie danych przy użyciu języka zapytań;
4. Nabycie umiejętności dodawania, modyfikowania i usuwania rekordów w bazie przy użyciu języka zapytań;
5. Nabycie umiejętności programowania skryptów automatyzujących proces tworzenia bazy danych;
6. Nabycie umiejętności wyszukiwania informacji w bazie danych przy użyciu języka zapytań;
7. Nabycie umiejętności tworzenia formularzy i raportów do przetwarzania danych;
8. Kształtowanie postawy
   * aktywnego słuchania i prowadzenia dyskusji;
   * stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
   * odpowiedzialności za podejmowane działania;
   * kreatywności w rozwiązywaniu problemów;
   * stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
   * doskonalenia jakości wykonywanych działań;
   * analizowania i oceny podejmowanych działań;

**Cele operacyjne**

**Uczeń potrafi:**

1. zainstalować, skonfigurować i zaktualizować system zarządzania bazą danych,
2. utworzyć użytkowników w systemie zarządzania bazami danych,
3. utworzyć strukturę bazy danych zgodnie z projektem,
4. modyfikować strukturę istniejącej bazy danych,
5. zweryfikować poprawność struktury po rozbudowie lub modyfikacji bazy danych,
6. programować skrypty w języku zapytań automatyzujące tworzenie struktury bazy danych,
7. tworzyć, modyfikować i usuwać tabele przy użyciu języka zapytań,
8. dodawać, modyfikować i usuwać rekordy w bazie danych przy użyciu języka zapytań,
9. pobrać dane z bazy danych przy użyciu języka zapytań,
10. wyszukać dane w bazie o zadanych kryteriach przy użyciu języka zapytań,
11. utworzyć formularz do wprowadzania, modyfikowania i usuwania danych,
12. tworzyć raporty w bazie danych,
13. importować i importować tabele bazy danych,
14. tworzyć i weryfikować kopię zapasową bazy danych,
15. przywrócić bazę danych z kopii zapasowej,
16. zdiagnozować i naprawić bazę danych.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Programowanie baz danych.** | 1. Programowanie w języku SQL. | 39 | - tworzyć bazy danych i tabele poleceniem CREATE,  - usuwać bazy danych i tabele poleceniem DROP,  - wstawiać rekordy do tabeli poleceniem INSERT,  - usuwać rekordy z tabeli poleceniem DELETE,  - modyfikować rekordy w tabeli poleceniem UPDATE,  - tworzyć strukturę bazy danych przy użyciu języka zapytań SQL zgodnie z otrzymanym projektem,  - pobierać dane z bazy za pomocą zapytania SELECT – FROM – WHERE | - sortować pobrane dane za pomocą polecenia ORDER BY według wybranej kolumny,  - pobierać dane z tabeli bez powtórzeń z pomocą polecenia DISTINCT,  - pobierać dane z tabel z wykorzystaniem operatorów AND i OR  - pobierać dane z tabel z użyciem funkcji np.: AVG, MAX, MIN, SUM,  - tworzyć skrypty w języku zapytań SQL  - pobierać dane z kilku tabel z użyciem poleceń INNER JOIN, OUTER JOIN  - łączyć wyniki zapytań za pomocą polecenia UNION | Klasa II |
| 2. Tworzenie bazy danych na podstawie projektu. | 15 | - definiować tabele ustalając atrybuty kolumn i typy przechowywanych w nich danych,  wprowadzać dane do bazy z poziomu systemu zarządzania bazą danych,  - importować dane do bazy z pliku,  - eksportować strukturę bazy do pliku,  -eksportować dane z bazy do pliku, | - programować skrypty tworzące strukturę bazy danych,  - eksportować i importować bazę danych online, |  |
| **II. Praca z bazą danych.** | 1. Praca z danymi w bazie danych. | 18 | - tworzyć formularze do wprowadzania danych do bazy,  - tworzyć zapytania do pojedynczych tabel w bazie danych,  - generować raporty w bazie danych, | - tworzyć formularze do wprowadzania danych z poziomu przeglądarki internetowej,  - stosować podzapytania w zapytaniach do bazy danych,  - tworzyć zapytania łączące wyniki z wielu tabel w bazie, | Klasa II |
| 2. Modyfikacja bazy danych | 6 | - dodawać tabele, rekordy i relacje do istniejącej bazy danych,  - usuwać elementy struktury bazy danych,  - usuwać dane w bazie danych,  - weryfikować poprawność struktury bazy danych, | - dodawać tabele, rekordy i relacje do istniejącej bazy danych za pomocą poleceń języka zapytań,  - modyfikować istniejące tabele, rekordy i zapytania w bazie danych, | Klasa II |
| **III. Administrowanie bazą danych.** | 1. Instalacja i konfiguracja SZBD. | 6 | - zainstalować system baz danych,  - konfigurować system baz danych,  - aktualizować system baz danych, | - zainstalować system baz danych online,  - tworzyć użytkowników z odpowiednimi uprawnieniami przy użyciu języka zapytań, | Klasa II |
| 2. Zarządzanie bazą danych | 6 | - tworzyć konta użytkowników,  - nadawać uprawnienia uzytkownikom  - tworzyć kopię zapasową,  - przywracać dane z kopii zapasowej,  - eksportować i importować tabele bazy danych, | - diagnozować bazę ,  - naprawiać błędy w bazie danych, | Klasa II |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągania założonych celów kształcenia w zakresie przedmiotu jest opracowanie odpowiednich dla danego zawodu procedur, a w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

ZALECANE METODY DYDAKTYCZNE

Dla przedmiotu Tworzenie i zarządzanie bazami danych, który jest przedmiotem praktycznym, dominujące powinny być metody praktyczne np.: pokaz z objaśnieniem, ćwiczenia praktyczne, metoda projektów, metoda przewodniego tekstu. W kształceniu zawodowym dobrze sprawdzają się też metody problemowe, szczególnie aktywizujące, np.: metoda przypadków, dyskusja dydaktyczna.

FORMY ORGANIZACYJNE

Zajęcia z przedmiotu Tworzenie i zarządzanie bazami danych powinny być prowadzone indywidualnie lub zespołowo. Należą one do grupy zajęć praktycznych. O liczebności grup ogólnie mówi §6 Rozporządzenia Ministra Edukacji Narodowej z dnia 22 lutego 2019 r. w sprawie praktycznej nauki zawodu. W celu spełnienia wymienionych w nim warunków realizacji kształcenia praktycznego, zajęcia edukacyjne powinny się odbywać w pracowni z podziałem na grupy o liczebności maksymalnie do 12 osób.

**ŚRODKI DYDAKTYCZNE**

W pracowni, w której prowadzone będą zajęcia powinny się znajdować: plansze z blokami diagramów E/R, przykładowe diagramy E/R, plansze z poleceniami SQL, plansze z typowymi zapytaniami w języku SQL, literatura dotycząca baz danych, stanowiska komputerowe dla ucznia i nauczyciela wyposażone w komputer z oprogramowaniem do zarzadzania bazami danych, dostępem do Internetu, oraz bazą danych dostępna online. W pracowni powinien znajdować się projektor, tablica interaktywna, drukarka oraz miejsca do pracy zespołowej wyposażone w flipchart i stoły. Uczniowie powinni posiadać dostęp do Team lub Slack umożliwiający pracę zespołową i komunikację online.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Do głównych metod sprawdzania osiągnięć edukacyjnych ucznia należą ćwiczenia praktyczne oraz metoda projektów. Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu na podstawie kryteriów, przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* realizowanych zadań praktycznych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania zadań praktycznych,
* produktu projektu i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest, aby uczniowie dokonywali także własnej samooceny pracy i kolegów z zespołu wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

**Przykładowe zadania**

Zadanie 1

Utwórz strukturę bazy danych zgodnie z poniższym modelem relacyjnym przedstawiającym strukturę tabel.

Oznaczenia: **klub podstawowy**, *klucz obcy*  
- Kluby (**Id\_k**, Kraj, Miasto, Nazwa)  
- Piłkarze (**Id\_p**, Imie, Nazwisko, Rok\_urodzenia, *Id\_k*)  
- Wypozyczenia (**Id\_w**, *id\_k, id\_p,* Do\_kiedy)  
- Trener (**Id\_t**, Imie, Nazwisko, Rok\_urodzenia, *Id\_k*)

Zadanie 2

Na podstawie utworzonej struktury w zadaniu 1 utwórz następujące relacje dla poszczególnych tabel:

- relacja jeden do wielu pomiędzy tabelami Kluby i Piłkarze,

- relacja jeden do jednego pomiędzy tabelami Kluby i Trener,

- relacja jeden do wielu pomiędzy tabelami Kluby i Wypozyczenia oraz Piłkarze i Wypozyczenia,

- stwórz formularze pozwalające na wprowadzenie danych do poszczególnych tabel,

- uzupełnij poszczególne tabele Kluby, Trener i Piłkarze na podstawie 4 klubów piłkarskich polskiej ligi, informacje pobierz z Internetu,

Zadanie 3

Na podstawie bazy danych z zadania 2 stwórz raport który wyświetli nazwę klubu, trenera danego klubu i zawodników znajdujących grających w danym klubie. Jeśli raport udało Ci się stworzyć poprawnie to uzupełnij tabelę Wypozyczenia wprowadzając 10 zawodników jako wypożyczonych do innych klubów.

Zmodyfikuj raport z początku zadania tak aby nie wyświetlał zawodników wypożyczonych do innego klubu. Po wykonaniu zadania zastanów się, czy tak utworzona tabela Wypozyczenia nie będzie powodowała błędów tzn. czy umożliwi wypożyczenie zawodnika do 2 klubów jednocześnie? Zastanów się nad modyfikację tabeli Wypozyczenia oraz formularza wprowadzającego do niej dane aby zabezpieczyć się przed błędami. Propozycje przedyskutuj w grupie.

**PROPONOWANE METODY EWALUACJI**

Strategia przeprowadzanej ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych. Zebrane dane zostaną poddane analizie ilościowej i jakościowej przy użyciu narzędzia, którym jest statystyka matematyczna. Przydatnym narzędziem w tej analizie może być na przykład korzystanie z platformy testowej office 365.com lub podobnej, która daje możliwość analizy, które z pytań testowych sprawiają trudność.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego. Spowoduje to podwyższenie jakości kształcenia i znacząco wpłynie na indywidualne wyniki uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. stosowania zasad programowania baz danych w języku zapytań SQL,

2. tworzenie bazy danych zgodnie z projektem,

3. tworzenie baz danych za pomocą języka zapytań SQL,

4. praca z danymi z a pomocą języka SQL,

5. tworzenie formularzy do pracy z danymi,

6. administrowanie systemem baz danych.

**ZALECANA LITERATURA**

Jolanta Pokorska, Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych. Część 2. Projektowanie i administrowanie bazami danych. Podręcznik do nauki zawodu technik informatyk i technik programista, wyd. Helion,

## Strony i aplikacje internetowe

**Cele ogólne przedmiotu**

1. Poznanie

* przeznaczenia , składni i poleceń języka HTML
* właściwości i reguł CSS;
* rodzajów stylów;
* systemów zarządzania treścią CMS;
* właściwości i funkcji edytorów grafiki rastrowej i wektorowej;
* właściwości plików graficznych i multimedialnych;
* zasad komputerowego przetwarzania dźwięku i obrazu;
* reguł testowania stron internetowych, walidacji i optymalizacji ich kodu;
* zasad publikowania stron i aplikacji internetowych
* usług hostingu;
* zasad programowania strukturalnego;
* skryptowych języków programowania;
* składni języka JavaScript;
* bibliotek i frameworków języka JavaScript;
* składni jednego z języków skryptowych działających po stronie serwera: PHP, Python, JSP, ASP.NET;
* funkcji środowiska programistycznego;
* funkcji debugera w przeglądarce internetowej ;
* zasad walidacji kodu programu;
* zasad dokumentowania aplikacji;

1. Nabycie umiejętności

* rozróżniania znaczników HTML i ich atrybutów
* posługiwania się językiem HTML do tworzenia elementów strony;
* rozróżniania selektorów CSS;
* wykorzystania stylów do formatowania elementów strony,
* analizowania kodu strony;
* rozróżniania funkcji systemów CMS;
* charakteryzowania funkcji edytorów graficznych do tworzenia komputerowej grafiki rastrowej i wektorowej;
* osadzania multimediów na stronie internetowej;
* opisywania zasad testowania stron internetowych, dokonywania walidacji i optymalizacji ich kodu;
* scharakteryzowania zasad publikowania stron i aplikacji internetowych;
* analizowania i rozwiązywania problemów programistycznych;
* analizowania i tworzenia skryptów JavaScript;
* analizowania i tworzenia skryptów wykonywanych po stronie serwera w jednym z języków: PHP, Python, JSP ASP.NET;
* dobierania środowiska programistycznego do realizowanych zadań i języka programowania;
* omawiania środowisk programistycznych;
* opisywania zasad walidacji kodu programu;
* opisywania zasad dokumentowania tworzonej aplikacji;

1. Kształtowanie postawy, świadomości

* aktywnego słuchania i prowadzenia dyskusji;
* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
* wrażliwości na potrzeby osób niepełnosprawnych;
* odpowiedzialności za podejmowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;

**Cele operacyjne**

**Uczeń potrafi:**

1. wymienić polecenia (znaczniki) języka HTML, grupując je zależnie od przeznaczenia,
2. rozróżnić polecenia języka HTML i ich atrybuty,
3. scharakteryzować znaczniki języka stron internetowych i ich atrybuty,
4. zastosować polecenia języka HTML do tworzenia elementów strony (akapity, nagłówki, obrazy, hiperłącza, mapy odsyłaczy, listy, tabele, formularze, pływające ramki),
5. zastosować elementy strukturalne do tworzenia układu strony,
6. wymienić właściwości CSS,
7. skonstruować regułę CSS,
8. rozróżnić selektory CSS (elementów, atrybutów, klas, identyfikatorów, pseudoklas, pseudoelementów, kontekstowe)
9. scharakteryzować rodzaje stylów (lokalny, osadzony, zewnętrzny),
10. zdefiniować strukturę arkusza CSS,
11. objaśnić kaskadowość stylów,
12. zastosować reguły CSS do formatowania strony internetowej,
13. wykonać projekt strony internetowej,
14. scharakteryzować responsywny układ strony,
15. scharakteryzować systemy zarządzania treścią,
16. dobrać system zarządzania treścią do określonego zastosowania,
17. omówić konfigurację systemu zarządzania treścią (Joomla i Wordpress),
18. opisać dostępne funkcje administracyjne systemów CMS
19. scharakteryzować grafikę rastrową,
20. scharakteryzować grafikę wektorową,
21. opisać różnice między edytorami grafiki rastrowej i wektorowej,
22. scharakteryzować typy plików graficznych,
23. objaśnić zasady cyfrowego zapisu obrazu,
24. scharakteryzować modele barw,
25. wykorzystywać kodowanie kolorów na stronach www,
26. objaśnić funkcje edytorów grafiki rastrowej i edytorów grafiki wektorowej,
27. dobrać oprogramowanie do edycji dźwięku i obrazu ruchomego,
28. określić zasady komputerowego przetwarzania dźwięku i wideo na potrzeby stron internetowych,
29. wyjaśnić zasady tworzenia animacji,
30. osadzić elementy multimedialne na stronie internetowej,
31. określić cechy i zastosowanie plików wideo, dźwiękowych i animacji,
32. zaprojektować układ sekcji na stronie internetowej,
33. analizować projekt strony internetowej,
34. stworzyć strukturę strony wg projektu,
35. uwzględniać potrzeby niepełnosprawnych użytkowników strony (np. kontrast, powiększenie)
36. opisać zasady i znaczenie wytycznych dotyczących ułatwień w dostępie do treści publikowanych w internecie,
37. scharakteryzować etapy procesu testowania strony internetowej,
38. opisać zasady testowania responsywności strony internetowej,
39. opisać proces walidacji strony internetowej,
40. dobrać narzędzia walidacji strony internetowej,
41. opisać proces pozycjonowania strony internetowej,
42. określić zasady dostępności (wytyczne WCAG) i pozycjonowania strony internetowej,
43. opisać usługę hostingową,
44. wyjaśnić pojęcia: domena, subdomena, serwer wirtualny, serwer dedykowany,
45. dobrać usługi hostingu do potrzeb użytkownika,
46. opisać operacje na domenach internetowych,
47. określić etapy publikacji witryn i aplikacji internetowych,
48. opisać funkcje programów służących do przesyłania danych na serwer,
49. dobrać program do przesyłania danych na serwer,
50. analizować problemy programistyczne,
51. tworzyć algorytmy w postaci listy kroków, pseudokodu, schematu blokowego,
52. analizować algorytmy,
53. określić zasady programowania strukturalnego,
54. identyfikować skryptowe języki programowania,
55. implementować algorytmy w języku interpretowanym ,
56. rozróżniać polecenia języka Javascript,
57. wyjaśnić pojęcie zmiennej,
58. omówić ogólne zasady nazywania zmiennych w JavaScript,
59. rozróżniać typy proste zmiennych w JavaScript i w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET (np.: liczbowe, tekstowe, logiczne),
60. dokonać konwersji typów prostych,
61. rozróżniać operatory, ich priorytety i typy (przypisania, porównania, arytmetyczne, logiczne, bitowe, [inkrementacji / dekrementacji]),
62. zastosować instrukcje warunkowe if...else, [switch],
63. realizować powtórzenia w skryptach za pomocą pętli for, [for... in], while i [do...while],
64. tworzyć skrypty w języku JavaScript i w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET z wykorzystaniem instrukcji warunkowych i pętli,
65. wykonywać operacje na ciągach tekstowych,
66. definiować własne funkcje bezargumentowe, funkcje z argumentami, przekazywać argumenty,
67. określić zasięg zmiennych (globalny, lokalny, [blokowy]),
68. zdefiniować funkcje rekurencyjne,
69. zdefiniować złożone typy danych (tablice, obiekty),
70. wykonywać operacje na elementach tablicy (sumowanie, wyszukiwanie, zliczanie wg kryterium, zamiana elementów, sortowanie),
71. rozróżniać właściwości i metody obiektów wbudowanych w JavaScript (np. document, window, style, navigator, location, string, Math, Date, Number),
72. tworzyć funkcje anonimowe,
73. objaśnić obiektowy model dokumentu (DOM),
74. opisać zasady dynamicznej zmiany stylu i zawartości strony,
75. opisać zasady dynamicznego tworzenia elementów drzewa dokumentu,
76. opisać sposoby rejestracji funkcji obsługujących zdarzenie (inline, przypisanie funkcji [anonimowej], słuchacz zdarzeń),
77. zastosować obsługę zdarzeń związanych z myszą i klawiaturą,
78. pobierać i przetwarzać dane z okien dialogowych i pól formularzy,
79. opisywać mechanizmy walidacji formularzy HTML za pomocą mechanizmów HTMLS,
80. definiować wyrażenia regularne,
81. opisać zasady manipulowania elementami strony za pomocą skryptów JavaScript,
82. opisać zastosowania bibliotek i frameworków języka JavaScript, w tym JQuery, Angular, React,
83. definiować złożone typy danych (tablice, obiekty),
84. pobierać i przesyłać dane z formularza,
85. opisać sposób współpracy aplikacji z bazą danych (biblioteki),
86. opisać sposób definiowania zapytań do bazy danych,
87. objaśnić mechanizm ciasteczek (cookie) i sesji,
88. opisać funkcje obsługi plików,
89. zaimplementować algorytmy w językach skryptowych: JavaScript i w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,
90. opisać funkcje środowiska programistycznego,
91. dobrać środowisko programistyczne do określonych zadań i języka programowania,
92. scharakteryzować gotowe pakiety dla aplikacji internetowych, np. phpMyAdmin
93. analizować błędy w kodzie źródłowym programu,
94. opisać sposoby testowania tworzonych programów,
95. poprawiać błędy w tworzonych programach,
96. zastosować debugger w przeglądarce internetowej,
97. opisać sposoby umieszczania komentarzy w kodzie źródłowym programu,
98. opisać sposoby dokumentowania programu,
99. opisać zasady tworzenia instrukcji użytkownika programu.
100. aktywnie słuchać włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,
101. stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;
102. zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;
103. wykazywać się kreatywnością w rozwiązywaniu problemów,
104. ponosić odpowiedzialność za podejmowane działania,
105. dokonać analizy i oceny podejmowanych działań,
106. doskonalić jakość wykonywanych działań;
107. ułatwiać dostęp do informacji osobom niepełnosprawnym;
108. opisać sposoby dokumentowania programu,
109. opisać zasady tworzenia instrukcji użytkownika programu.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Tworzenie stron interneto-wych** | 1. Język HTML |  | wymienić polecenia (znaczniki) języka HTML (standard HTML5),  rozróżniać polecenia języka HTML i ich atrybuty,  scharakteryzować znaczniki języka stron internetowych i ich atrybuty,  zastosować polecenia języka HTML do tworzenia elementów strony (akapity, nagłówki, obrazy, hiperłącza wewnętrzne i zewnętrzne, mapy odsyłaczy, listy, tabele, formularze, pływające ramki),  zastosować elementy strukturalne do tworzenia układu strony | grupować polecenia języka HTML zależnie od przeznaczenia,  zanalizować projekt strony internetowej, | Klasa I |
| 2.Kaskadowe arkusze stylów (CSS) |  | wymienić podstawowe właściwości CSS elementów strony (np.: czcionki, tekstu, tła, obramowania, odsyłaczy, list, kursora, marginesów wewnętrznych i zewnętrznych, rozmiarów elementów),  skonstruować regułę CSS,  rozróżnić selektory CSS (elementów, atrybutów, klas, identyfikatorów, pseudoklas, pseudoelementów, kontekstowe)  scharakteryzować rodzaje stylów,  zdefiniować strukturę arkusza stylów,  dołączyć styl do strony internetowej,  objaśnić kaskadowość stylów,  zastosować reguły CSS do formatowania strony internetowej | wykonać projekt strony internetowej z zastosowaniem CSS,  scharakteryzować responsywny układ strony | Klasa I |
| 3.Systemy zarządzania treścią (CMS) |  | scharakteryzować systemy zarządzania treścią (CMS),  dobrać system zarządzania treścią do określonego zastosowania | omówić konfigurację systemu zarządzania treścią (Joomla i Wordpress),  opisać dostępne funkcje administracyjne systemów CMS |  |
| **II. Grafika i multimedia** | 4. Grafika komputerowa |  | scharakteryzować grafikę rastrową,  scharakteryzować grafikę wektorową,  scharakteryzować typy plików graficznych,  scharakteryzować modele barw,  dobierać oprogramowanie do obróbki grafiki komputerowej,  wykorzystać kodowanie kolorów na stronach www,  objaśnić funkcje edytorów grafiki rastrowej i edytorów grafiki wektorowej, | opisać różnice między edytorami grafiki rastrowej i wektorowej,  objaśnić zasady cyfrowego zapisu obrazu,  scharakteryzować metody kompresji obrazu | Klasa I |
| 1. Multimedia na stronie www |  | dobrać oprogramowanie do edycji dźwięku i obrazu ruchomego na potrzeby stron internetowych,  wyjaśnić zasady tworzenia animacji,  osadzić elementy multimedialne na stronie internetowej | określić zasady komputerowego przetwarzania dźwięku i wideo na potrzeby stron internetowych,  określić cechy i zastosowanie plików wideo, dźwiękowych i animacji | Klasa I |
| **III. Projektowa-nie, testowanie i publikowa-nie stron interneto-wych** | 1. Projektowanie stron internetowych |  | zaprojektować układ sekcji na stronie internetowej,  analizować projekt strony internetowej,  stworzyć strukturę strony wg projektu,  doskonalić jakość wykonywanych działań;  ułatwiać dostęp do informacji osobom niepełnosprawnym; | uwzględniać potrzeby niepełnosprawnych użytkowników strony (np. kontrast, powiększenie)  opisać zasady i znaczenie wytycznych dotyczących ułatwień w dostępie do treści publikowanych w internecie, | Klasa I |
| 1. Testowanie i publikowanie stron |  | scharakteryzować etapy procesu testowania strony internetowej,  opisać zasady testowania responsywności strony internetowej,  opisać proces walidacji strony internetowej,  dobrać narzędzia walidacji strony internetowej,  opisać usługę hostingowa,  wyjaśnić pojęcia: domena, subdomena, serwer wirtualny, serwer dedykowany,  opisać operacje na domenach internetowych,  określić etapy publikacji witryn i aplikacji internetowych,  opisać funkcje programów służących do przesyłania danych na serwer,  dobrać program do przesyłania danych na serwer,  ponosić odpowiedzialność za podejmowane działania,  dokonać analizy i oceny podejmowanych działań, | opisać proces pozycjonowania strony internetowej,  określić zasady dostępności (wytyczne WCAG) i pozycjonowania strony internetowej,  dobrać usługę hostingową do potrzeb użytkownika, | Klasa I |
| **IV. Algorytmy i zasady programo-wania** | 1. Algorytmy i zasady programowania strukturalnego |  | analizować proste problemy programistyczne,  opisać reprezentacje algorytmów,  analizować proste algorytmy,  określić zasady programowania strukturalnego,  aktywnie słuchać włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,  stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;  zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;  wykazywać się kreatywnością w rozwiązywaniu problemów, | analizować złożone problemy programistyczne,  tworzyć algorytmy w postaci listy kroków, pseudokodu, schematu blokowego,  analizować złożone algorytmy | Klasa I |
| **V. Skryptowe języki programo-wania** | 1. Skryptowe języki programowania |  | identyfikować skryptowe języki programowania  implementować proste algorytmy w języku interpretowanym  rozróżniać polecenia języka JavaScript i jednego z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  wyjaśnić pojęcie zmiennej,  omówić ogólne zasady nazywania zmiennych w JavaScript i w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  rozróżniać typy proste zmiennych (liczbowe, tekstowe, logiczne),  dokonać konwersji typów prostych,  rozróżniać operatory, ich priorytety i typy (przypisania, porównania, arytmetyczne, logiczne, bitowe, [inkrementacji / dekrementacji]),  stosować instrukcje warunkowe if...else, [switch],  realizować powtórzenia w skryptach za pomocą pętli for, while i [do...while],  tworzyć skrypty w języku JavaScript i w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET z wykorzystaniem instrukcji warunkowych i pętli,  definiować własne funkcje bezargumentowe,  wyjaśnić pojęcie rekurencji,  definiować złożone typy danych (tablice, obiekty) | implementować złożone algorytmy w języku interpretowanym ,  stosować pętle zagnieżdżone,  definiować własne funkcje z argumentami,  przekazywać argumenty funkcji,  określać zasięg zmiennych w (np. w JavaScript - globalny, lokalny, blokowy),  zdefiniować funkcje rekurencyjne,  wykonywać operacje na elementach tablicy (np. wyszukiwanie, zliczanie wg kryterium, zamiana elementów, sortowanie) | Klasa I |
| **VI.Tworze-nie, testowanie i dokumento-wanie aplikacji interneto-wych** | 1. Skrypty wykonywane po stronie klienta |  | rozróżniać właściwości i funkcje obiektów wbudowanych w JavaScript (np. document, window, navigator, location, style, string, Math, Date, Number)  scharakteryzować obiektowy model dokumentu (DOM),  opisać zasady dynamicznej zmiany stylu i zawartości strony,  opisać zasady dynamicznego tworzenia elementów drzewa dokumentu,  opisać sposoby rejestracji funkcji obsługujących zdarzenie (inline, przypisanie funkcji [anonimowej], słuchacz zdarzeń),  manipulować elementami strony za pomocą skryptów JavaScript, np. animacje, efekt rollover, menu rozwijane,  pobierać i przetwarzać dane z okien dialogowych i kontrolek formularzy,  wykonywać operacje na ciągach tekstowych,  wyjaśnić pojęcie frameworku,  wymienić rodzaje frameworków | tworzyć funkcje anonimowe,  weryfikować dane z formularza,  opisywać mechanizmy walidacji formularzy HTML za pomocą mechanizmów HTMLS,  definiować wyrażenia regularne,  opisać zastosowania bibliotek i frameworków języka JavaScript, w tym JQuery, Angular, React, | Klasa II |
| 1. Skrypty wykonywane po stronie serwera |  | tworzyć proste programy w jednym z języków wykonywanych po stronie serwera: Python, PHP, JSP, ASP.NET,  stosować wbudowane instrukcje i funkcje, np. działające na tekstach,  pobierać i przesyłać dane z formularza,  opisać sposób współpracy aplikacji z bazą danych (biblioteki),  objaśnić mechanizm ciasteczek (cookie) | tworzyć zaawansowane programy w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  objaśnić mechanizm sesji,  opisać funkcje obsługi plików,  opisać sposób definiowania zapytań do bazy danych  opisuje bibliotekę PDO | Klasa II |
| 1. Środowisko programistyczne i uruchomieniowe aplikacji internetowych |  | opisać funkcje środowiska programistycznego,  tworzyć programy w wybranym środowisku programistycznym  scharakteryzować gotowe pakiety dla aplikacji internetowych, np. phpMyAdmin | dobrać środowisko programistyczne do określonych zadań i języka programowania, | Klasa II |
| 1. Walidacja kodu programu |  | opisać sposoby testowania tworzonych programów,  poprawiać błędy w tworzonych programach, | analizować błędy w kodzie źródłowym programu,  stosować debugger w przeglądarce internetowej, | Klasa II |
| 1. Dokumentowanie aplikacji |  | opisuje sposoby umieszczania komentarzy w kodzie źródłowym programu, | opisać sposoby dokumentowania programu,  opisać zasady tworzenia instrukcji użytkownika programu. | Klasa II |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Do osiągania założonych celów kształcenia w zakresie przedmiotu Strony i aplikacje internetowe konieczne jest

* zaplanowanie lekcji poprzez wskazanie celów szczegółowych,
* wykorzystanie różnorodnych metod nauczania, w szczególności aktywizujących,
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany, kartkówki, testy i projekty,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

Środki dydaktyczne

W sali lekcyjnej lub pracowni, w której prowadzone będą zajęcia powinno znajdować się stanowisko komputerowe dla nauczyciela, z odpowiednim oprogramowaniem (edytor stron internetowych z możliwością edycji hipertekstowego języka znaczników, kaskadowych arkuszy stylów, skryptów wykonywanych po stronie klienta i po stronie serwera, z możliwością walidacji strony; oprogramowanie do tworzenia grafiki i animacji, obróbki materiałów audio i wideo; podłączenie do sieci lokalnej z dostępem do Internetu; dostęp do serwera umożliwiającego publikację stron www i aplikacji internetowych, z obsługą baz danych; dostęp do portalu wspierającego pracę grupową), podłączone do sieci lokalnej z dostępem do internetu z urządzeniem wielofunkcyjnym oraz z projektorem multimedialnym lub tablicą interaktywną, ekran, głośniki, wskazane są także stanowiska komputerowe dla uczniów (jedno stanowisko dla maksimum dwóch uczniów) z odpowiednim oprogramowaniem (takim jak nauczyciela). W pracowni, w której prowadzone będą zajęcia powinny się znajdować: plansze poglądowe, filmy dydaktyczne i prezentacje multimedialne związane z treściami kształcenia, wzorce stron i aplikacji internetowych, dokumentacje języków programowania, czasopisma branżowe.

Zalecane metody dydaktyczne

Dla przedmiotu Strony i aplikacje internetowe, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych, opartych na słowie i obserwacji, takich jak:

* wykład informacyjny,
* pogadanka,
* pokaz z objaśnieniem,
* wykład problemowy,
* dyskusja dydaktyczna,
* burza mózgów,
* ćwiczenia praktyczne,

Zajęcia mogą odbywać się w grupach. Istotną metodą kształcenia powinny być także ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku.

Formy organizacyjne

Zajęcia z przedmiotu Tworzenie stron i aplikacji internetowych powinny być prowadzone indywidualnie, zespołowo lub zbiorowo. Należą one do grupy zajęć teoretycznych. W przypadku realizacji ćwiczeń praktycznych powinny być stosowane formy organizacyjne indywidualne. W kształceniu zawodowym bardzo ważną kwestią jest dostosowanie wymagań edukacyjnych do zróżnicowanych potrzeb i możliwości uczniów. Ponadto uczniowie powinni samodzielnie zdobywać wiedzę i kształtować umiejętności poprzez uczenie się we współpracy oraz korzystanie z różnych źródeł informacji.

Formy indywidualizacji pracy uczniów

W celu dostosowania warunków, środków, metod i form kształcenia do potrzeb i możliwości ucznia można w zakresie organizacji pracy zastosować instrukcje do zadań, podawanie dodatkowych zaleceń, objaśnień, instrukcji do pracy indywidualnej, udzielanie konsultacji indywidualnych. Nauczyciel powinien dostosować stopień trudności materiału do możliwości i potrzeb uczniów. W pracy grupowej należy zwracać uwagę na taki podział zadań między członków zespołu, by każdy wykonywał tę część zadania, której podoła. Uczniom szczególnie zdolnym i posiadającym określone zainteresowania zawodowe należy zaplanować zadania o większym stopniu trudności, proponować samodzielne poszerzanie wiedzy, korzystanie z dodatkowej literatury.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

W celu sprawdzenia osiągnięć edukacyjnych ucznia proponuje się: testy wielokrotnego wyboru, testy zawierające zadania otwarte, kartkówki, prezentacje uczniów, projekty.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu, na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* realizowanych ćwiczeń praktycznych w formie zadań,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń,
* produktu projektu i jego prezentacji,

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (tutoriali, dokumentacji danego języka – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest też, by uczniowie dokonywali samooceny i oceny kolegów z zespołu według zaproponowanych przez nauczyciela kryteriów.

Stosowane przez nauczyciela ocenianie powinno korzystać z zasad występujących w ocenianiu kształtującym, ma być dla ucznia informacją zwrotną, która pomaga mu się uczyć. Nauczyciel wskazuje silne strony ucznia i doradza, jak je rozwijać; wyraźnie i konstruktywnie informuje o stronach słabych i o tym, jak można je eliminować; stwarza uczniom możliwość poprawienia własnej pracy. Ocenianie skupiające się na postępach i osiągnięciach, a nie na podkreślaniu niepowodzeń, zachęca uczniów do uczenia się. Porównywanie osiągnięć poszczególnych uczniów z osiągnięciami ich kolegów, tworzenie rankingów, nie motywuje, a często zniechęca do uczenia się. Nauczyciel korzysta też z oceniania sumującego, nie rezygnuje ze stopni.

**Przykładowe zadania**

1. Scharakteryzuj znaczniki służące do tworzenia tabel. Wymień właściwości CSS, które można ustawić dla tabeli i jej komórek. Podaj ich przykładowe wartości.
2. Napisz skrypt zliczający częstość wystąpienia określonego znaku w tekście. \*) Zmodyfikuj skrypt tak, by możliwe było wczytanie znaku do zliczania oraz wybranie tekstu do analizy.
3. Napisz skrypt wykonywany po stronie serwera, który nawiązuje połączenie z bazą danych *ksiazki*, pobiera z niej wszystkie rekordy i wyświetla je na stronie na liście numerowanej.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Koncepcja ewaluacji przedmiotu polegać będzie na tzw. twardej analizie danych, czyli ocen zdobytych przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych oraz z przygotowanych przez uczniów projektów. Zebrane wyniki zostaną poddane analizie ilościowej i jakościowej z wykorzystaniem statystyki matematycznej.

Uzyskane informacje pozwolą na wyodrębnienie zagadnień sprawiających uczniom problemy. W następstwie, będzie można zwiększyć liczbę godzin dydaktycznych przypisanych do danego działu programowego. Korekta taka powinna się przyczynić do podwyższenia jakości kształcenia i poprawy indywidualnych wyników uczniów z egzaminu w danej kwalifikacji.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

* tworzenia stron internetowych,
* formatowania stron internetowych,
* programowania skryptów wykonywanych po stronie klienta,
* programowania skryptów wykonywanych po stronie serwera,
* analizowania poprawności działania aplikacji

**LITERATURA**

1. Jolanta Pokorska, Podręcznik do zawodu technik informatyk, technik programista, część 1, Tworzenie stron internetowych, Kwalifikacja INF.03.Programwanie, tworzenie i administrowanie stronami internetowymi i bazami danych, wyd. Helion –podręcznik w przygotowaniu.
2. Adam Freeman, HTML5. Przewodnik encyklopedyczny, wyd. Helion
3. David Sawayer McFarland, CSS3 nieoficjalny podręcznik, wyd. Helion
4. David Sawayer McFarland, JavaScript i jQuery. Nieoficjalny podręcznik, wyd. Helion,
5. Eric T. Freeman, Elisabeth Robson, Programowanie w Javascript. Rusz głową!, wyd.Helion
6. [Luke Welling](https://helion.pl/autorzy/luke-welling), [Laura Thomson](https://helion.pl/autorzy/laura-thomson), PHP i MySQL. Tworzenie stron WWW. Vademecum profesjonalisty, wyd. Helion
7. http://codecademy.com
8. http://w3schools.com

## Tworzenie stron i aplikacji internetowych

**Cele ogólne przedmiotu**

1. Poznanie

* przeznaczenia, składni i poleceń języka HTML
* właściwości i reguł CSS;
* rodzajów stylów;
* systemów zarządzania treścią CMS;
* właściwości i funkcji edytorów grafiki rastrowej i wektorowej;
* właściwości plików graficznych i multimedialnych;
* zasad komputerowego przetwarzania dźwięku i obrazu;
* reguł testowania stron internetowych, walidacji i optymalizacji ich kodu;
* zasad publikowania stron i aplikacji internetowych
* usług hostingu;
* zasad programowania strukturalnego;
* skryptowych języków programowania;
* składni języka JavaScript;
* bibliotek i frameworków języka JavaScript;
* składni jednego z języków skryptowych wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET;
* funkcji środowiska programistycznego;
* funkcji debugera w przeglądarce internetowej ;
* zasad walidacji kodu programu;
* zasad dokumentowania aplikacji;

1. Nabycie umiejętności

* rozróżniania znaczników HTML i ich atrybutów
* posługiwania się językiem HTML do tworzenia elementów strony;
* rozróżniania selektorów CSS;
* wykorzystania stylów do formatowania elementów strony,
* analizowania kodu strony;
* korzystania z systemów CMS do tworzenia serwisów WWW;
* korzystania z edytorów graficznych do tworzenia grafiki komputerowej rastrowej i wektorowej;
* osadzania multimediów na stronie internetowej;
* testowania stron internetowych, dokonywania walidacji i optymalizacji ich kodu;
* publikowania stron i aplikacji internetowych;
* analizowania i rozwiązywania problemów programistycznych;
* analizowania i tworzenia skryptów JavaScript;
* analizowania i tworzenia skryptów wykonywanych po stronie serwera w jednym z języków: PHP, Python, JSP ASP.NET;
* dobierania środowiska programistycznego do realizowanych zadań i języka programowania;
* tworzenia programów w wybranych środowiskach programistycznych;
* dokonywania walidacji kodu programu;
* dokumentowania tworzonej aplikacji;

1. Kształtowanie postawy, świadomości

* aktywnego słuchania i prowadzenia dyskusji;
* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
* wrażliwości na potrzeby osób niepełnosprawnych;
* odpowiedzialności za podejmowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;
* pracy w zespole;

**Cele operacyjne**

**Uczeń potrafi:**

1. rozróżniać polecenia języka HTML i ich atrybuty,
2. stworzyć stronę internetową zawierającą akapity, nagłówki, obrazy, hiperłącza wewnętrzne i zewnętrzne, mapy odsyłaczy, listy, tabele, pływające ramki,
3. wykonać formularz na stronie internetowej,
4. zastosować elementy strukturalne do tworzenia układu strony,
5. tworzyć regułę CSS,
6. rozróżnić selektory CSS (elementów, atrybutów, klas, identyfikatorów, pseudoklas, pseudoelementów, kontekstowe)
7. stosować różne rodzaje stylów (lokalny, osadzony, zewnętrzny),
8. stworzyć arkusz stylów,
9. zastosować reguły CSS do formatowania strony internetowej,
10. wykonać projekt strony internetowej,
11. wykonać animację z zastosowaniem CSS,
12. wykonać responsywną stronę internetową z zastosowaniem CSS,
13. zastosować systemy zarządzania treścią,
14. dobrać system zarządzania treścią do określonego zastosowania,
15. zainstalować system CMS (Joomla i Wordpress),
16. skonfigurować system zarządzania treścią (Joomla i Wordpress),
17. administrować systemami zarządzania treścią,
18. aktualizować system CMS,
19. tworzyć stronę z wykorzystaniem gotowego szablonu dla systemów CMS,
20. projektować strony internetowe, wykorzystując systemy CMS,
21. opisać różnice między grafiką rastrową i wektorową,
22. przestrzegać zasad cyfrowego zapisu obrazu,
23. stosować modele barw,
24. wykorzystywać kodowanie kolorów na stronach www,
25. tworzyć grafikę w edytorze grafiki rastrowej i edytorze grafiki wektorowej,
26. osadzić tekst na grafice i dobrać jego krój i styl,
27. edytować grafikę na potrzeby stron internetowych – np. skalowanie, przycinanie, fotomontaż, retusz,
28. projektować elementy graficzne dla strony internetowej, np. przyciski, tekstury, loga,
29. dobrać oprogramowanie do edycji dźwięku i obrazu ruchomego,
30. stworzyć animację na potrzeby strony internetowej,
31. wykonać materiały wideo na potrzeby strony internetowej,
32. edytować wideo i dźwięk na potrzeby strony internetowej,
33. osadzić elementy multimedialne na stronie internetowej,
34. zaimportować pliki z multimediami do systemów zarządzania treścią (CMS),
35. zaprojektować układ sekcji na stronie internetowej,
36. analizować projekt strony internetowej,
37. stworzyć strukturę strony wg projektu,
38. dobrać paletę barw dla strony internetowej,
39. dobrać czcionki dla strony internetowej,
40. uwzględniać potrzeby niepełnosprawnych użytkowników strony (np. kontrast, powiększenie)
41. stworzyć stronę zgodną z wytycznymi dotyczącymi ułatwień w dostępie do treści publikowanych w internecie,
42. testować stronę internetową w różnych przeglądarkach,
43. testować responsywność strony internetowej,
44. dobrać narzędzia walidacji strony internetowej,
45. dokonać walidacji strony internetowej,
46. zoptymalizować stronę internetową,
47. stosować zasady dostępności (WCAG) i pozycjonowania strony internetowej,
48. dobrać usługi hostingu do potrzeb użytkownika,
49. wykonać operacje na domenach internetowych,
50. dobrać program do przesyłania danych na serwer,
51. przesłać dane na serwer,
52. dobrać pakiety serwerowe www,
53. sprawdzić poprawność publikowanych stron www,
54. publikować witryny internetowe,
55. analizować problemy programistyczne,
56. analizować algorytmy,
57. stosować zasady programowania strukturalnego,
58. identyfikować skryptowe języki programowania,
59. implementować algorytmy w języku interpretowanym ,
60. stosować polecenia języka JavaScript i jednego z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,
61. stosować zmienne,
62. rozróżniać typy proste zmiennych (np.: liczbowe, tekstowe, logiczne),
63. dokonać konwersji typów prostych,
64. stosować operatory zgodnie z ich priorytetami (przypisania, porównania, arytmetyczne, logiczne, bitowe, [inkrementacji / dekrementacji]),
65. zastosować instrukcje warunkowe if...else, [switch],
66. realizować powtórzenia w skryptach za pomocą pętli for, [for... in], while, [do...while],
67. tworzyć skrypty w w/w językach z wykorzystaniem instrukcji warunkowych i pętli,
68. wykonywać operacje na ciągach tekstowych,
69. definiować własne funkcje bezargumentowe, funkcje z argumentami, przekazywać argumenty,
70. określić zasięg zmiennych (globalny, lokalny, [blokowy]),
71. zastosować funkcje strzałkowe w JavaScript (arrow function),
72. zastosować funkcje rekurencyjne,
73. zastosować złożone typy danych (tablice, obiekty), np. wykonywać operacje na elementach tablicy (sumowanie, wyszukiwanie, zliczanie wg kryterium, zamiana elementów, sortowanie),
74. zastosować wybrane biblioteki skryptowych języków programowania,
75. tworzyć skrypty w języku JavaScript,
76. zastosować właściwości i metody obiektów wbudowanych (np. document, window, style, navigator, location, string, Math, Date, Number),
77. wykorzystać funkcje i właściwości obiektowego modelu dokumentu (DOM), np. dynamiczna zmiana stylu i zawartości strony, dynamiczne tworzenie drzewa dokumentu,
78. zastosować obsługę zdarzeń związanych z myszą i klawiaturą metodą klasyczną i ze słuchaczem zdarzeń, np. galeria zdjęć,
79. pobierać dane z pól formularzy i przetwarzać je w skryptach,
80. zastosować mechanizmy walidacji formularzy HTML za pomocą mechanizmów HTMLS,
81. manipulować elementami strony za pomocą skryptów JavaScript, np. rozwijane menu, animacja,
82. zastosować biblioteki i frameworki języka JavaScript, w tym JQuery, Angular, React,
83. tworzyć strony internetowe wykorzystujące skryptowe języki programowania,
84. tworzyć skrypty w jednym z języków programowania wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,
85. zastosować wbudowane funkcje, np. działające na tekście,
86. pobierać i przesyłać dane z formularza,
87. realizować dostęp do bazy danych z poziomu jednego z języków: PHP, Python , JSP, ASP.NET,
88. zastosować biblioteki do obsługi bazy danych odpowiednie dla języka i frameworka,
89. przesyłać zapytania do bazy,
90. pobierać dane z bazy,
91. dodawać, usuwać, aktualizować dane w bazie,
92. korzystać z funkcji do obsługi ciasteczek (cookies) oraz sesji,
93. korzystać z funkcji do obsługi plików,
94. implementować algorytmy w językach skryptowych: JavaScript i w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,
95. dobrać środowisko programistyczne do określonych zadań i języka programowania,
96. tworzyć programy w wybranym środowisku programistycznym,
97. zainstalować i skonfigurować serwer www,
98. zainstalować i skonfigurować serwer baz danych,
99. korzystać z gotowych pakietów dla aplikacji internetowych, np. phpMyAdmin
100. analizować błędy w kodzie źródłowym programu,
101. wykonywać testy tworzonych programów,
102. poprawiać błędy w tworzonych programach,
103. zastosować debugger w przeglądarce internetowej,
104. zastosować komentarze w kodzie źródłowym programu,
105. tworzyć dokumentację programu,
106. tworzyć instrukcję użytkownika programu.
107. aktywnie słuchać włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,
108. stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;
109. zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;
110. wykazywać się kreatywnością w rozwiązywaniu problemów,
111. ponosić odpowiedzialność za podejmowane działania,
112. dokonać analizy i oceny podejmowanych działań,
113. doskonalić jakość wykonywanych działań;
114. pracować w zespole,
115. oceniać przydatność poszczególnych członków zespołu do wykonania zadania,
116. rozdzielać zadania według umiejętności i kompetencji członków zespołu,
117. ustalać kolejność wykonywania zadań,
118. monitorować proces wykonywania zadań,
119. kontrolować efekty pracy zespołu,
120. oceniać pracę poszczególnych członków zespołu pod względem zgodności z warunkami technicznymi odbioru prac,
121. udzielić wskazówek w celu prawidłowego wykonania przydzielonych zadań
122. ułatwiać dostęp do informacji osobom niepełnosprawnym;

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Tworze-nie stron interne-towych** | 1. Język HTML |  | rozróżniać polecenia języka HTML i ich atrybuty (standard HTML5),  posługiwać się znacznikami i ich atrybutami,  stworzyć stronę internetową zawierającą akapity, nagłówki, obrazy, hiperłącza wewnętrzne i zewnętrzne, listy, tabele, zastosować elementy strukturalne języka HTML do tworzenia układu strony, | wykonać formularz na stronie internetowej,  zastosować na stronie internetowej mapę odsyłaczy, pływające ramki | Klasa I |
| 2.Kaskadowe arkusze stylów (CSS) |  | tworzyć regułę CSS,  rozróżnić selektory CSS (elementów, atrybutów, klas, identyfikatorów, pseudoklas, pseudoelementów, kontekstowe)  stosować różne rodzaje stylów (lokalny, osadzony, zewnętrzny),  stworzyć arkusz stylów,  podpiąć arkusz stylów do strony internetowej  zastosować reguły CSS do formatowania strony internetowej, | wykonać projekt strony internetowej z zastosowaniem CSS,  stosować kaskadowosć stylów,  zastosować CSS do uzyskania responsywności strony internetowej,  wykonać animację z zastosowaniem CSS,  wykonać rozwijane menu z zastosowaniem CSS | Klasa I |
| 3.Systemy zarządzania treścią (CMS) |  | stosować systemy zarządzania treścią,  dobrać system zarządzania treścią do określonego zastosowania,  zainstalować system CMS (Joomla i Wordpress)  aktualizować system CMS,  tworzyć stronę z wykorzystaniem gotowego szablonu dla systemów CMS | skonfigurować system zarządzania treścią (Joomla i Wordpress)  administrować systemami zarządzania treścią,  instalować motywy i wtyczki w systemach CMS,  projektować strony internetowe, wykorzystując systemy CMS, |  |
| **II. Grafika i multimedia** | 4. Grafika komputerowa |  | opisać różnice między grafiką rastrową i wektorową,  przestrzegać zasad cyfrowego zapisu obrazu,  stosować modele barw,  wykorzystywać kodowanie kolorów na stronach www (szestnastkowy, dziesiętny),  przeliczać zdefiniowany kolor z zapisu szestnastkowego na dziesiętny i odwrotnie,  tworzyć grafikę w edytorze grafiki rastrowej i edytorze grafiki wektorowej,  osadzić tekst na grafice i dobrać jego krój i styl,  edytować grafikę na potrzeby stron internetowych – np. skalowanie, przycinanie  eksportować i zapisywać grafikę w różnych formatach | edytować grafikę na potrzeby stron internetowych – np. fotomontaż, retusz,  projektować elementy graficzne dla strony internetowej, np. przyciski, tekstury, loga | Klasa I |
| 1. Multimedia na stronie www |  | dobrać oprogramowanie do edycji dźwięku i obrazu ruchomego,  stosować narzędzia i funkcje programu do obróbki dźwięku, wideo i animacji,  stworzyć animację na potrzeby strony internetowej,  osadzić elementy multimedialne na stronie internetowej,  eksportować i zapisywać wideo, dźwięk i animację w różnych formatach  zaimportować pliki z multimediami do systemów zarządzania treścią (CMS) | wykonać materiały wideo na potrzeby strony internetowej,  edytować wideo i dźwięk na potrzeby strony internetowej | Klasa I |
| **III. Projektowa-nie, testowanie i publikowa-nie stron interneto-wych** | 1. Projektowanie stron internetowych |  | zaprojektować układ sekcji na stronie internetowej,  analizować projekt strony internetowej,  stworzyć strukturę strony wg projektu,  dobrać czcionki dla strony internetowej,  dobrać paletę barw dla strony internetowej | uwzględniać potrzeby niepełnosprawnych użytkowników strony (np. kontrast, powiększenie)  stworzyć stronę zgodną z wytycznymi dotyczącymi ułatwień w dostępie do treści publikowanych w internecie,  ułatwiać dostęp do informacji osobom niepełnosprawnym; | Klasa I |
| 1. Testowanie, walidacja i publikowanie stron |  | testować w różnych przeglądarkach wygląd strony internetowej, funkcjonalność i szybkość wczytywania strony,  testować responsywność strony internetowej,  dokonać walidacji strony internetowej,  dobrać program do przesyłania danych na serwer,  przesłać dane na serwer,  sprawdzić poprawność wyświetlania się i działania opublikowanych stron www,  publikować witryny internetowe | dobrać narzędzia walidacji strony internetowej,  zoptymalizować stronę internetową,  wykonać operacje na domenach internetowych,  dobrać usługi hostingu do potrzeb użytkownika,  dobrać pakiety serwerowe www stosować zasady dostępności (WCAG) i pozycjonowania strony internetowej | Klasa I |
| **IV. Algorytmy i zasady programowania** | 1. Algorytmy i zasady programowania strukturalnego |  | analizować proste problemy programistyczne,  analizować proste algorytmy,  stosować zasady programowania strukturalnego  aktywnie słuchać włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,  stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;  zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;  wykazywać się kreatywnością w rozwiązywaniu problemów, | analizować złożone problemy programistyczne,  analizować złożone algorytmy | Klasa II |
| **V. Skryptowe języki programowania** | 1. Skryptowe języki programowania |  | identyfikować skryptowe języki programowania,  implementować proste algorytmy w języku interpretowanym ,  stosować polecenia języka JavaScript i jednego z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  stosować zmienne,  rozróżniać typy proste (liczbowe, tekstowe, logiczne),  stosować operatory zgodnie z ich priorytetami (przypisania, porównania, arytmetyczne, logiczne, bitowe, [inkrementacji / dekrementacji]),  stosować instrukcje warunkowe if...else, [switch],  realizować powtórzenia w skryptach za pomocą pętli for, while, [do...while],  tworzyć skrypty w w/w językach z wykorzystaniem instrukcji warunkowych i pętli,  wykonywać operacje na ciągach tekstowych,  definiować własne funkcje bezargumentowe,  stosować złożone typy danych (tablice, listy, rekordy, obiekty, pliki), np. wykonywać operacje na elementach tablicy (sumowanie, wyszukiwanie, sortowanie),  stosować wybrane biblioteki skryptowych języków programowania | implementować złożone algorytmy w języku interpretowanym ,  stosować pętle zagnieżdżone,  definiować własne funkcje z argumentami,  przekazywać argumenty funkcji,  określać zasięg zmiennych w (np. w JavaScript - globalny, lokalny, blokowy),  stosować funkcje rekurencyjne,  stosować funkcje strzałkowe w JavaScript (arrow function),  stosować złożone typy danych (tablice, obiekty), np. wykonywać operacje na elementach tablicy (zliczanie wg kryterium, zamiana elementów)  tworzyć strony internetowe wykorzystujące skryptowe języki programowania, | Klasa II |
| **VI.Tworze-nie, testowanie i dokumento-wanie aplikacji interneto-wych** | 1. Skrypty wykonywane po stronie klienta |  | tworzyć skrypty w języku JavaScript,  stosować właściwości i metody obiektów wbudowanych (np. document, window, navigator, location, style, Math, Date, Number),  wykorzystać funkcje i właściwości obiektowego modelu dokumentu (DOM),  odwołać się do elementu strony (getElement..., querySelector...) lub wykorzystując relacje między węzłami drzewa (np. parentNode, firstChild, lastChild, nextSibling, previousSibling, children, childNodes),  wstawić zawartość (np. innerHTML, innerText),  ustawić i pobierać atrybuty HTML danego znacznika (getAttribute, setAttribute lub poprzez właściwości obiektów),  tworzyć elementy strony (np. createElement, appendChild, insertBefore, createTextNode),  zastosować funkcje i właściwości obiektowego modelu dokumentu (DOM) do tworzenia strony, np. z dynamiczną zmianą stylu i zawartości, dynamicznym tworzeniem drzewa dokumentu,  zastosować obsługę zdarzeń związanych z myszą i klawiaturą (metodą klasyczną i ze słuchaczem zdarzeń), np. efekt rollover,  pobrać dane z pól formularzy i przetwarzać je w skryptach,  ponosić odpowiedzialność za podejmowane działania,  dokonać analizy i oceny podejmowanych działań,  doskonalić jakość wykonywanych działań; | zastosować wyrażenia regularne do weryfikacji danych z formularza,  stosować mechanizmy walidacji formularzy HTML za pomocą mechanizmów HTMLS,  manipulować elementami strony za pomocą skryptów JavaScript, np. rozwijane menu, animacja, galeria zdjęć, slajder,  stosować biblioteki (np. Animejs, number-flip) i frameworki języka JavaScript, w tym JQuery, Angular, React | Klasa II |
| 1. Skrypty wykonywane po stronie serwera |  | tworzyć proste skrypty w jednym z języków programowania wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  rozróżniać polecenia zapisane w jednym z języków programowania wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  stosować instrukcje jednego z języków programowania wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  stosować złożone typy danych (tablice, obiekty) w jednym z języków programowania wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  stosować wbudowane funkcje, np. działające na tekście,  pobierać i przesyłać dane z formularza,  realizować dostęp do bazy danych z poziomu jednego z języków: PHP, Python, JSP, ASP.NET,  stosować biblioteki do obsługi bazy danych odpowiednie dla języka i frameworka,  korzystać z funkcji do obsługi ciasteczek (cookies)  pracować w zespole,  oceniać przydatność poszczególnych członków zespołu do wykonania zadania,  rozdzielać zadania według umiejętności i kompetencji członków zespołu,  ustalać kolejność wykonywania zadań,  monitorować proces wykonywania zadań,  kontrolować efekty pracy zespołu,  oceniać pracę poszczególnych członków zespołu pod względem zgodności z warunkami technicznymi odbioru prac,  udzielać wskazówek w celu prawidłowego wykonania przydzielonych zadań | tworzyć zaawansowane programy w jednym z języków wykonywanych po stronie serwera: PHP, Python, JSP, ASP.NET,  przesyłać zapytania do bazy,  pobierać dane z bazy,  aktualizować dane w bazie,  dodawać i usuwać dane z bazy,  korzystać z mechanizmu sesji,  korzystać z funkcji do obsługi plików, | Klasa II |
| 1. Środowisko programistyczne i uruchomieniowe aplikacji internetowych |  | tworzyć programy w wybranym środowisku programistycznym,  instalować i konfigurować serwer www,  instalować i konfigurować serwer baz danych,  korzystać z gotowych pakietów dla aplikacji internetowych, np. phpMyAdmin | dobrać środowisko programistyczne do określonych zadań i języka programowania, | Klasa II |
| 1. Walidacja kodu programu |  | wykonywać testy tworzonych programów,  poprawiać błędy w tworzonych programach | analizować błędy w kodzie źródłowym programu,  stosować debugger w przeglądarce internetowej, | Klasa II |
| 1. Dokumentowanie aplikacji |  | stosować komentarze w kodzie źródłowym programu | tworzyć dokumentację programu,  tworzyć instrukcję użytkownika programu | Klasa II |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Do osiągnięcia założonych celów kształcenia w zakresie przedmiotu Tworzenie stron i aplikacji internetowych konieczne jest:

* zaplanowanie lekcji poprzez wskazanie celów szczegółowych,
* wykorzystanie różnorodnych metod nauczania, w szczególności aktywizujących,
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez testy praktyczne i projekty,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

Środki dydaktyczne

W pracowni, w której prowadzone będą zajęcia powinno znajdować się stanowisko komputerowe dla nauczyciela, podłączone do sieci lokalnej z dostępem do internetu z urządzeniem wielofunkcyjnym oraz z projektorem multimedialnym lub tablicą interaktywną, ekran, głośniki, stanowiska komputerowe (jedno stanowisko dla jednego ucznia) z odpowiednim oprogramowaniem: edytor stron internetowych z możliwością edycji hipertekstowego języka znaczników, kaskadowych arkuszy stylów, skryptów wykonywanych po stronie klienta i po stronie serwera, z możliwością walidacji strony; oprogramowanie do tworzenia grafiki i animacji, obróbki materiałów audio i wideo; podłączenie do sieci lokalnej z dostępem do Internetu; dostęp do serwera umożliwiającego publikację stron www i aplikacji internetowych, z obsługą baz danych; dostęp do portalu wspierającego pracę grupową.

Zalecane metody dydaktyczne

Dla przedmiotu Tworzenie stron i aplikacji internetowych, który jest przedmiotem praktycznym, dominujące powinny być metody praktyczne np.: pokaz z objaśnieniem, ćwiczenia praktyczne, metoda projektów, metoda przewodniego tekstu. W kształceniu zawodowym dobrze sprawdzają się też metody problemowe, szczególnie aktywizujące, np.: metoda przypadków, dyskusja dydaktyczna.

Formy organizacyjne

Zajęcia z przedmiotu Tworzenie stron i aplikacji internetowych powinny być prowadzone indywidualnie lub zespołowo. Należą one do grupy zajęć praktycznych. O liczebności grup ogólnie mówi §6 Rozporządzenia Ministra Edukacji Narodowej z dnia 22 lutego 2019 r. w sprawie praktycznej nauki zawodu. W celu spełnienia wymienionych w nim warunków realizacji kształcenia praktycznego, zajęcia edukacyjne powinny się odbywać w pracowni z podziałem na grupy o liczebności maksymalnie do 12 osób.

Formy indywidualizacji pracy uczniów

W celu dostosowania warunków, środków, metod i form kształcenia do potrzeb i możliwości ucznia w zakresie organizacji pracy można zastosować instrukcje do zadań, podawanie dodatkowych zaleceń, objaśnień, instrukcji do pracy indywidualnej, udzielanie konsultacji indywidualnych. W pracy grupowej należy zwracać uwagę na taki podział zadań między członków zespołu, by każdy wykonywał tę część zadania, której podoła. Uczniom szczególnie zdolnym i posiadającym określone zainteresowania zawodowe należy zaplanować zadania o większym stopniu złożoności, proponować samodzielne poszerzanie wiedzy, studiowanie dodatkowej literatury.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Do głównych metod sprawdzania osiągnięć edukacyjnych ucznia należą ćwiczenia praktyczne oraz metoda projektów. Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu na podstawie kryteriów, przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* realizowanych zadań praktycznych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania zadań praktycznych,
* efektu końcowego projektu i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie z sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (tutoriali, dokumentacji technicznej – w tym w języku obcym)

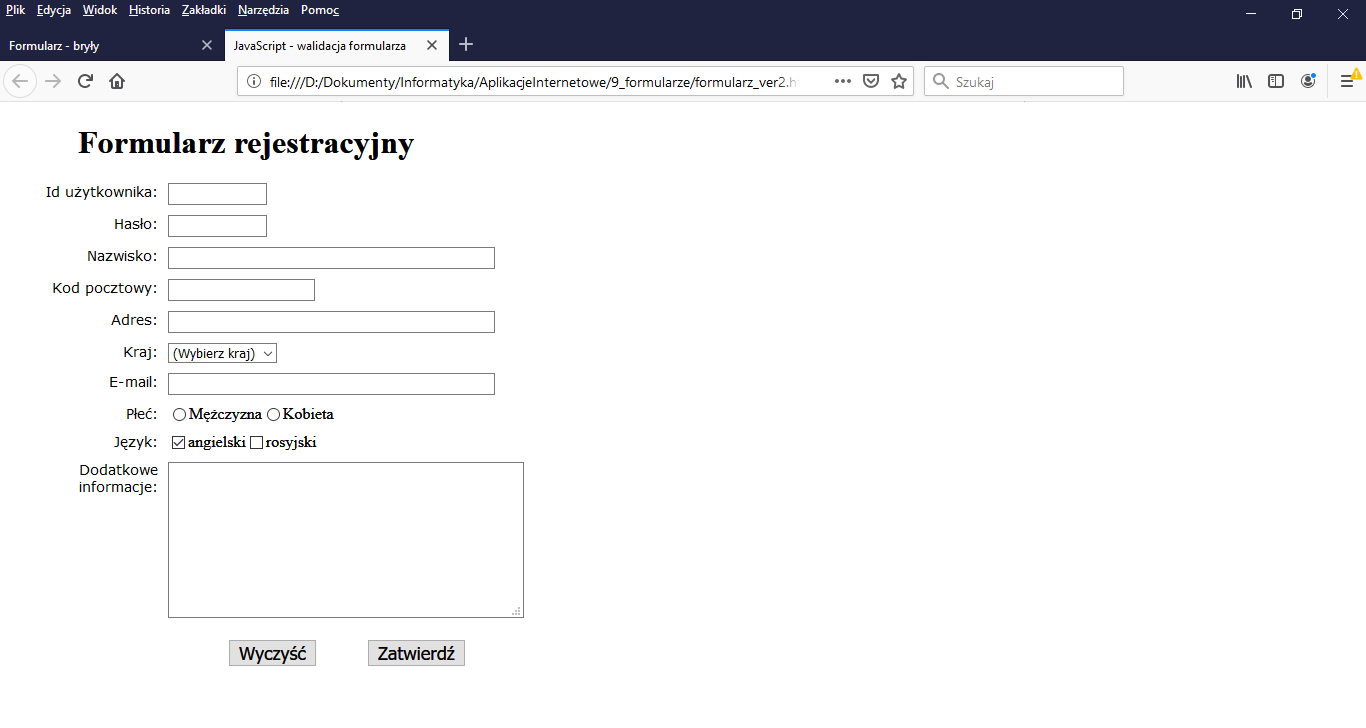
Wskazane jest też, by uczniowie dokonywali samooceny i oceny kolegów z zespołu według zaproponowanych przez nauczyciela kryteriów.

Stosowane przez nauczyciela ocenianie powinno korzystać z zasad występujących w ocenianiu kształtującym, ma być dla ucznia informacją zwrotną, która pomaga mu się uczyć. Nauczyciel wskazuje silne strony ucznia i doradza, jak je rozwijać; wyraźnie i konstruktywnie informuje o stronach słabych i o tym, jak można je eliminować; stwarza uczniom możliwość poprawienia własnej pracy. Ocenianie skupiające się na postępach i osiągnięciach, a nie na podkreślaniu niepowodzeń, zachęca uczniów do uczenia się. Porównywanie osiągnięć poszczególnych uczniów z osiągnięciami ich kolegów, tworzenie rankingów, nie motywuje, a często zniechęca do uczenia się. Nauczyciel korzysta też z oceniania sumującego, nie rezygnuje ze stopni.

**Przykładowe zadania**

1. Zaprojektuj i umieść na stronie formularz z etykietami, polami, listą i przyciskami jak na rysunku. Zrealizuj za pomocą CSS jego formatowanie – rozmieszczenie etykiet, pól, listy, przycisków, wyrównanie pól formularza i tekstów etykiet.

Zadanie ma być wykonywane indywidualnie. Ocenie podlegać będzie zgodność formularza z założeniami, zastosowanie odpowiednich znaczników, poprawne formatowanie za pomocą arkusza stylów CSS, estetyka.



1. Przygotuj stronę internetową, demonstrującą działanie skryptów JavaScript, o układzie jak na rysunku 1, składającą się z następujących bloków:

* baner z tekstem „Przykładowe skrypty JavaScript” w nagłówku stopnia drugiego,
* umieszczone obok siebie
  + panel lewy o szerokości 30%, wysokości 400px, zawierający menu w postaci listy punktowanej,
  + panel prawy o szerokości 70%, wysokości 400px, z zawartością będącą wynikiem działania skryptów,
* na dole stopka z wyrównanym do prawej nazwiskiem autora oraz wyśrodkowanymi, wyświetlonymi drobnym drukiem informacjami o
  + dacie ostatniej modyfikacji dokumentu
  + lokalizacji pliku strony,
  + informacjami o przeglądarce.

Stronę sformatuj

* dobierając paletę barw i ustawiając różne kolory tła w poszczególnych blokach,
* dobierając czcionkę – rodzaj, styl, rozmiar,
* realizując zróżnicowane wyrównanie zawartości bloków
  + w nagłówku i panelu prawym – wyśrodkowana,
  + w panelu lewym – do lewej,
  + w stopce – autor do prawej, pozostałe informacje, wyświetlane skryptem, wyśrodkowane.

Rozmieść i sformatuj elementy strony dbając o jej estetykę i czytelność. Każda podstrona zawiera tytuł informujący o rodzaju przedstawianego skryptu i opisaną niżej zawartość.

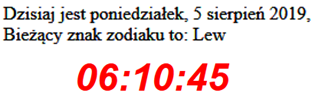
|  |  |
| --- | --- |
| **Przykładowe skrypty JavaScript** | |
| * Zegar i data * Animacja * Efekt rollover * Kula | W tym panelu mają być wyświetlane efekty działania skryptów |
| Autor: Jan Kowalski  Ostatnia modyfikacja: 12/08/2019 10:05:54  Adres URL: file:///D:/Dokumenty/Informatyka/skrypty.html  Przeglądarka: Mozilla Netscape 5.0 (Windows) | |

Rysunek 1.

**Wytyczne dotyczące skryptów działających na stronie**

Skrypt **Zegar i data**

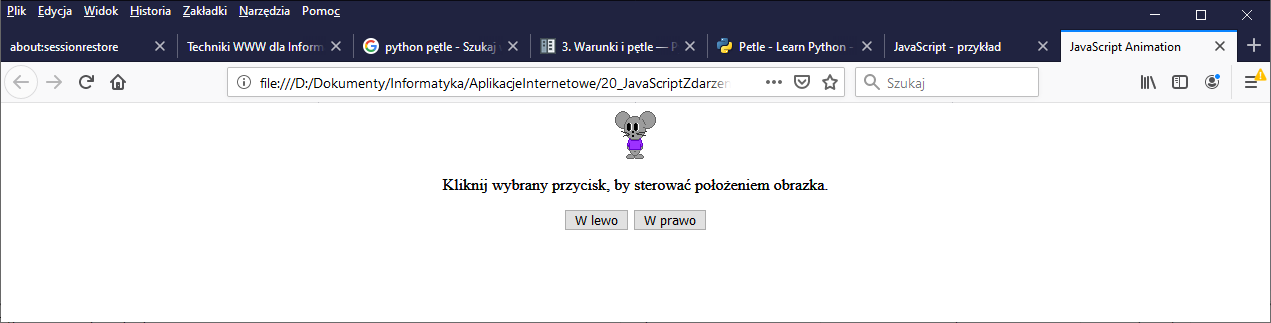
* wyświetla informacje o
  + bieżącej dacie z polskimi nazwami dni tygodnia i miesiąca (rysunek 2),
  + bieżącym znaku zodiaku,
  + aktualnym czasie,
* sformatuj zegar czcionką Arial, dobierając jej wielkość, styl i kolor,
* \*) dodatkowo skrypt oblicza i wyświetla liczbę dni, które pozostały do końca roku lub innego, wybranego wydarzenia.



Rysunek 2

Skrypt **Animacja**

zmienia położenie, tj. przesuwa w lewo lub prawo, dowolny, samodzielnie zaprojektowany i wykonany obrazek. Położenie ma być sterowane dwoma przyciskami. Każde kliknięcie, to przesunięcie obrazka o dowolnie wybraną liczbę pikseli w odpowiednią stronę.



Rysunek 3

Skrypt **Efekt rollover**

Podmienia dowolne obrazki po najechaniu na nie myszą.

Skrypt **Kula**

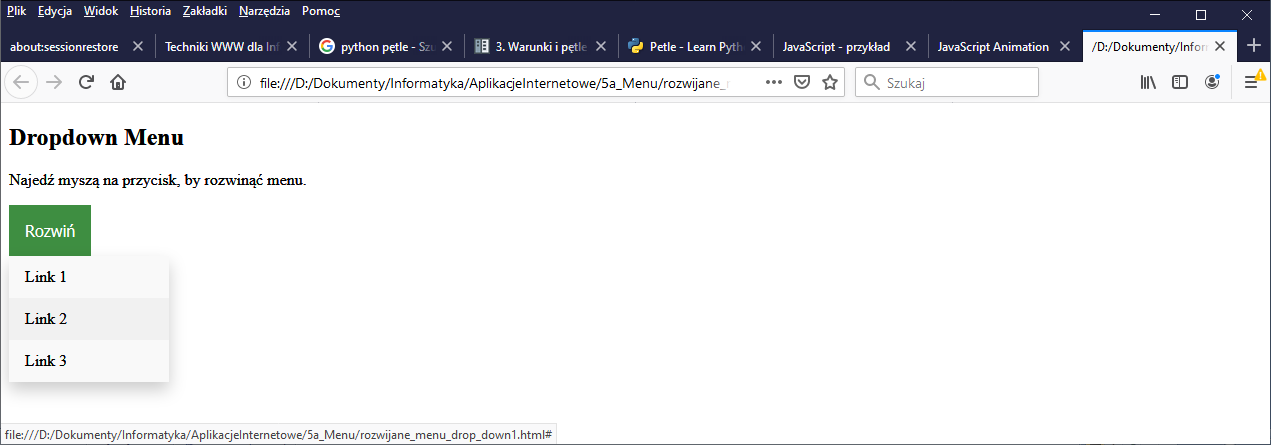
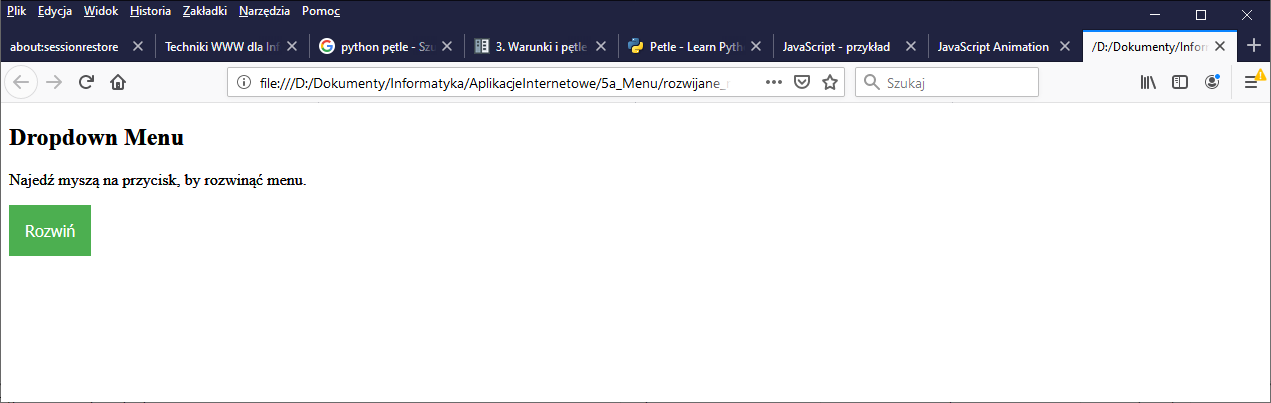
Przygotuj prosty formularz umożliwiający wczytanie promienia kuli. Formularz zawiera pole tekstowe i przycisk.

Skrypt

* wczytuje z pola tekstowego promień kuli jako liczbę całkowitą
* sprawdza, czy wczytana dana jest liczbą (funkcja isNaN)
* jeśli podano liczbę – oblicza ze wzoru i wyświetla pod formularzem objętość kuli (Math.PI, Math.pow), w przeciwnym przypadku wyświetla komunikat o błędzie.

Zadanie może być wykonywane zespołowo lub indywidualnie. Ocenie podlegać będzie zgodność strony internetowej z założeniami, zastosowanie odpowiednich znaczników, poprawne formatowanie strony za pomocą arkusza stylów CSS, estetyka strony i wykonanej przez ucznia grafiki, działanie skryptów.

1. Zaprojektuj menu rozwijane w pionie po najechaniu myszą. Rysunek 1 reprezentuje przykładowe menu Rozwiń, rysunek 2 rozwinięte menu (po najechaniu na nie myszą).



Rysunek 1 Rysunek 2

Zadanie powinno być wykonywane zespołowo lub indywidualnie. Ocenie podlegać będzie zgodność z założeniami, zastosowanie odpowiednich znaczników, formatowanie menu za pomocą arkusza stylów CSS, funkcjonalność i estetyka.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Koncepcja ewaluacji przedmiotu polegać będzie na tzw. twardej analizie danych, czyli ocen zdobytych przez uczniów z testów praktycznych z poszczególnych działów programowych oraz wykonanych przez uczniów projektów. Zebrane wyniki zostaną poddane analizie ilościowej i jakościowej z wykorzystaniem statystyki matematycznej.

Uzyskane informacje pozwolą na wyodrębnienie zagadnień sprawiających uczniom problemy. W następstwie, będzie można zwiększyć liczbę godzin dydaktycznych przypisanych do danego działu programowego. Korekta taka powinna się przyczynić do podwyższenia jakości kształcenia i indywidualnych wyników uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

* tworzenia stron internetowych,
* formatowania stron internetowych,
* programowania skryptów wykonywanych po stronie klienta,
* programowania skryptów wykonywanych po stronie serwera,
* analizowania poprawności działania aplikacji

**LITERATURA**

1. Jolanta Pokorska, Podręcznik do zawodu technik informatyk, technik programista, część 1, Tworzenie stron internetowych, Kwalifikacja INF.03.Programwanie, tworzenie i administrowanie stronami internetowymi i bazami danych, wyd. Helion –podręcznik w przygotowaniu,
2. Adam Freeman, HTML5. Przewodnik encyklopedyczny, wyd. Helion,
3. David Sawayer McFarland, CSS3 nieoficjalny podręcznik, wyd. Helion,
4. David Sawayer McFarland, JavaScript i jQuery. Nieoficjalny podręcznik, wyd. Helion,
5. Eric T. Freeman, Elisabeth Robson, Programowanie w Javascript. Rusz głową!, wyd.Helion,
6. [Luke Welling](https://helion.pl/autorzy/luke-welling), [Laura Thomson](https://helion.pl/autorzy/laura-thomson), PHP i MySQL. Tworzenie stron WWW. Vademecum profesjonalisty, wyd. Helion,
7. https://pl.khanacademy.org/computing/computer-programming,
8. http://codecademy.com,
9. http://w3schools.com

## Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych – klasa II

## Kwalifikacja INF.04. Projektowanie, programowanie i testowanie aplikacji – klasa III i IV

Język angielski zawodowy

**Cele ogólne**

* Posługiwanie się podstawowym zasobem środków językowych.
* Rozumienie prostych wypowiedzi ustnych.
* Samodzielne tworzenie krótkich, prostych, spójnych i logicznych wypowiedzi ustnych związanych z wykonywaniem zadań zawodowych.
* Przeprowadzenie rozmowy w sytuacjach związanych z realizacją zadań zawodowych związanych z wykonywaniem zadań zawodowych.
* Korzystanie z dokumentacji technicznej.

**Cele operacyjne**

Uczeń potrafi:

1. rozwijać sprawność językową (mówienie, rozumienie ze słuchu, czytanie i rozumienie różnych typów tekstów, pisanie różnych form) w zakresie tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, projektowania, programowania i testowania aplikacji,
2. rozwijać sprawność funkcjonalnego użycia języka obcego w zakresie tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, projektowania, programowania i testowania aplikacji,
3. rozwijać umiejętność pozyskiwania informacji niezbędnych w zakresie realizowanych zadań zawodowych z różnych źródeł w zakresie tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, projektowania, programowania i testowania aplikacji,
4. doskonalić rozumienie sensu wypowiedzi osób posługujących się językiem jako macierzystym w różnych sytuacjach,
5. posługiwać się zasobem środków językowych (leksykalnych, gramatycznych, ortograficznych oraz fonetycznych) umożliwiających realizację zadań zawodowych w zakresie tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, projektowania, programowania i testowania aplikacji,
6. analizować i interpretować krótkie teksty pisemne dotyczące wykonywania typowych czynności zawodowych w zakresie tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, projektowania, programowania i testowania aplikacji.

**MATERIAŁ NAUCZANIA JĘZYK ANGIELSKI ZAWODOWY**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dział programowy | Tematy jednostek metodycznych | **Liczba godz.** | | Wymagania programowe | | | Uwagi o realizacji |
| Podstawowe  **Uczeń potrafi:** | | Ponadpodstawowe  **Uczeń potrafi:** | Etap realizacji |
| Posługiwanie się językiem obcym zawodowym na stanowisku pracy związanym z tworzeniem i zarzadzaniem bazami danych, tworzeniem stron i aplikacji internetowych, | Charakterystyka stanowiska pracy |  | | * posługiwać się słownictwem związanym z czynnościami tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, * posługiwać się słownictwem związanym z narzędziami i materiałami wykorzystywanymi na stanowisku pracy, * posługiwać się słownictwem związanym z urządzeniami wykorzystywanymi na stanowisku pracy. | | * sformułować wypowiedź w języku obcym zawodowym związanym w czynnościami zawodowymi, * sformułować wypowiedź w języku obcym zawodowym związanym z narzędziami i materiałami wykorzystywanymi na stanowisku pracy, * sformułować wypowiedź w języku obcym zawodowym związanym z urządzeniami wykorzystywanymi na stanowisku pracy. | Klasa II |
| Tworzenie instrukcji, opisów związanych z wykonywaniem zadań zawodowych |  | | * znajdować w tekście określone informacje związane z tworzeniem i zarzadzaniem bazami danych, tworzeniem stron i aplikacji internetowych, * układać informacje w określonym porządku. | | * stworzyć instrukcję w języku obcym zawodowym dotyczącym stanowiska pracy, * stworzyć opis w języku obcym zawodowym dotyczącym stanowiska pracy. | Klasa II |
| Prowadzenie rozmowy kwalifikacyjnej | CV oraz list motywacyjny, dokumenty związane z wykonywanym zawodem |  | | * dobrać słownictwo w języku obcym zawodowym podczas tworzenia listu motywacyjnego, * dobrać słownictwo w języku obcym zawodowym podczas tworzenia CV. | | * napisać w języku obcym zawodowym list motywacyjny, * napisać w języku obcym zawodowym CV. | Klasa II |
| Rozmowa kwalifikacyjna |  | | * dobrać słownictwo w języku obcym zawodowym podczas prowadzenia rozmowy kwalifikacyjnej, * dobrać słownictwo określające zadania zawodowe. | | * sformułować wypowiedzi w języku obcym zawodowym podczas rozmowy kwalifikacyjnej, * sformułować wypowiedź opisującą zadania zawodowe technika programisty. | Klasa II |
| Formułowanie wypowiedzi w języku obcym zawodowym związane z tworzeniem i zarzadzaniem bazami danych, tworzeniem stron i aplikacji internetowych | Polecenie, komunikat, instrukcja |  | | * dobrać słownictwo w języku obcym zawodowym dotyczące poleceń związanych z tworzeniem i zarzadzaniem bazami danych, tworzeniem stron i aplikacji internetowych, * dobrać słownictwo w języku obcym zawodowym dotyczące komunikatów i instrukcji związanych z wykonywaniem zadań tworzenia i zarzadzania bazami danych, tworzenia stron i aplikacji internetowych, | | * sformułować polecenie w języku obcym zawodowym, * sformułować komunikaty w języku obcym zawodowym, * sformułować instrukcję w języku obcym zawodowym. | Klasa II |
| Tworzenie wiadomości e-mail i innych wiadomości tekstowych związanych z czynnościami zawodowym |  | | * dobrać słownictwo w języku obcym zawodowym przy tworzeniu wiadomości e-mail, * dobrać słownictwo w języku obcym zawodowym przy tworzeniu dokumentów typu wiadomość, formularz. | | * sformułować wiadomość  e-mail w języku obcym zawodowym, * sformułować formularz w języku obcym zawodowym, * sformułować wiadomość w języku obcym zawodowym. | Klasa II |
| Rozmowa z klientem. |  | | * dobrać słownictwo w języku obcym zawodowym dotyczące rozmowy z pracownikiem, * dobrać słownictwo w języku obcym zawodowym dotyczące rozmowy z kontrahentem. | | * posługiwać się słownictwem w języku obcym zawodowym w trakcie rozmowy z pracownikiem, * posługiwać się słownictwem w języku obcym zawodowym podczas rozmowy z kontrahentem. | Klasa II |
| Posługiwanie się językiem obcym zawodowym na stanowisku pracy związanym z projektowaniem, programowaniem i testowaniem aplikacji | Charakterystyka stanowiska pracy |  | | * posługiwać się słownictwem związanym w czynnościami projektowania, programowania i testowania aplikacji, * posługiwać się słownictwem związanym z narzędziami i materiałami wykorzystywanymi na stanowisku pracy, * posługiwać się słownictwem związanym z maszynami i urządzeniami wykorzystywanymi na stanowisku pracy. | | * sformułować wypowiedź w języku obcym zawodowym związanym w czynnościami zawodowymi, * sformułować wypowiedź w języku obcym zawodowym związanym z narzędziami programistycznymi na stanowisku pracy, * sformułować wypowiedź w języku obcym zawodowym związanym z urządzeniami wykorzystywanymi na stanowisku pracy. | Klasa III |
| Tworzenie instrukcji, opisów związanych z wykonywaniem zadań zawodowych |  | | * znajdować w tekście określone informacje związane z projektowaniem, programowaniem i testowaniem aplikacji, * układać informacje w określonym porządku. | | * stworzyć instrukcję w języku obcym zawodowym dotyczącym stanowiska pracy, urządzenia, * stworzyć opis w języku obcym zawodowym dotyczącym stanowiska pracy, urządzenia. | Klasa III |
| Dokumentacja w języku obcym | Formularze, specyfikacje i normy w języku obcym |  |  | | * określić główną myśl wypowiedzi/tekstu lub fragmentu wypowiedzi/tekstu, * znajdować w wypowiedzi/tekście określone informacje. | | Klasa III |
| Obcojęzyczna dokumentacja specjalistyczna |  |  | | * rozpoznać związki między poszczególnymi częściami tekstu. | | Klasa IV |
|  | Tworzenie wiadomości e-mail i innych wiadomości tekstowych związanych z czynnościami zawodowym |  |  | | * sformułować wiadomość  e-mail w języku obcym zawodowym, * sformułować formularz w języku obcym zawodowym, * sformułować wiadomość w języku obcym zawodowym. | | Klasa IV |
|  | Rozmowa z klientem. |  |  | | * posługiwać się słownictwem w języku obcym zawodowym w trakcie rozmowy z pracownikiem, * posługiwać się słownictwem w języku obcym zawodowym podczas rozmowy z kontrahentem. | | Klasa IV |
| **Razem** | |  |  | |  | |  |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągania założonych celów kształcenia w zakresie przedmiotu jest opracowanie odpowiednich dla danego zawodu procesu, a w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych, jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia,

**METODY NAUCZANIA**

Dla przedmiotu Język angielski zawodowy, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych, takich jak:

* wykład informacyjny,
* pokaz z objaśnieniem,
* wykład problemowy,
* metoda przypadku,
* dyskusja dydaktyczna,
* ćwiczenia przedmiotowe,
* burza mózgów.

Zajęcia powinny odbywać się w grupach maksymalnie 16-osobowych.

W zakresie kształcenia zawodowego bardzo dobrze sprawdza się również nauczanie problemowe ze szczególnym uwzględnieniem metod aktywizujących, np.

* metoda przypadków,
* metoda sytuacyjna,
* dyskusja dydaktyczna,
* gry dydaktyczne,

Nauczyciel powinien:

– motywować uczniów do pracy,

– dostosowywać stopień trudności planowanych ćwiczeń do możliwości uczniów,

– uwzględniać zainteresowania uczniów,

– przygotowywać zadania o różnym stopniu trudności i złożoności,

– zachęcać uczniów do korzystania z różnych źródeł informacji zawodowej,

– nauczyciel powinien stosować metody aktywizujące,

– nauczyciel powinien stosować nowoczesne środki kształcenia, np. tablice multimedialne.

**FORMY ORGANIZACYJNE**

Zajęcia powinny odbywać się w grupach, maksymalnie 16-osobowych. Zajęcia powinny być prowadzone z wykorzystaniem różnych form organizacyjnych: indywidualnie i zespołowo. Bardzo ważną kwestią w kształceniu zawodowym jest indywidualizacja pracy w kierunku potrzeb i możliwości ucznia w zakresie metod, środków oraz form kształcenia. Nauczyciel realizujący program powinien:

**ŚRODKI DYDAKTYCZNE**

* scenariusz dialogu (po jednym na grupę 3 os.) z usuniętymi interesującymi nas zdaniami,
* paski papieru ze zdaniami usuniętymi uprzednio z tekstu – po zestawie na grupę,
* CD lub filmy z nagraniem dialogu,
* zdjęcie przedstawiające bohaterów dialogu pogrążonych w rozmowie,
* słowniki.

Dla prawidłowej realizacji zajęć niezbędna jest pracownia językowa wyposażona m.in. w komputer stacjonarny z oprogramowaniem biurowym z dostępem do internetu, telewizor, tablicę flipchart, słuchawki z mikrofonem, system do nauczania języków obcych, podręczniki do nauczania języków obcych, słowniki, fiszki językowe, filmy i nagrania dydaktyczne, plansze dydaktyczne etc.

W nauczaniu należy odwołać się do e-zasobów do nauczania języka obcego ukierunkowanego zawodowo zaplanowanych wg. koncepcji programu nauczania funkcjonalno-sytuacyjnego. Osią tak pomyślanego programu są typowe sytuacje komunikacyjne, w których znajduje się osoba w swoim środowisku pracy.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Testy wielokrotnego wyboru, testy zawierające zadania otwarte, odpowiedzi ustne, prezentacje uczniów.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* ustnych sprawdzianów poziomu wiedzy i umiejętności,
* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest, aby uczniowie dokonywali także własnej samooceny pracy i kolegów z zespołu wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

**Przykładowe zadania:**

**Zadanie 1.**

Sformułuj wiadomość e-mail do kontrahenta.

**Zadanie 2**

Opracuj instrukcję użytkowania aplikacji wskazanej przez nauczyciela.

**Zadanie 3**

Opracuj dialog dotyczący rozmowy przełożonego z pracownikiem na temat zadań zawodowych.

**EWALUACJA PRZEDMIOTU**

Zaleca się stosowanie zarówno metod ilościowych, jak i jakościowych. Metody ilościowe mają w głównej mierze postać ankiet audytoryjnych. Główną zaletą tego typu rozwiązania jest możliwość dotarcia do dużej liczby osób, wadą natomiast brak pogłębionej refleksji. W przypadku zastosowania metod jakościowych (wywiad, obserwacja, analiza dokumentów) można dogłębnie poznać i zinterpretować problem. W przypadku ewaluacji programu typową metodą jest ankieta ewaluacyjna, natomiast narzędziem kwestionariusz ankiety, który zawiera pytania zadawane uczniom. Samo zbieranie danych możemy powierzyć praktycznie dowolnej osobie pod warunkiem, że wcześniej zostanie do tego przygotowana. Podczas realizacji badań ewaluacyjnych powinno się stosować wiele różnych metod badawczych. Daje to możliwość na uzupełnienie oraz pogłębianie danych i informacji zdobytych jedna metodą, a także sprzyja zachowaniu obiektywizmu. Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. posługiwania się językiem obcym zawodowym podczas formułowania wypowiedzi związanych z wykonywaniem zadań zawodowych,
2. posługiwania się językiem obcym zawodowym w trakcie opracowywania instrukcji, formularzy i innych dokumentów związanych z wykonywaniem zadań zawodowych,
3. posługiwania się słownictwem z zakresu baz danych, witryn internetowych i programowania,
4. posługiwania się językiem obcym zawodowym podczas rozmów w kontrahentami i pracownikami.

**ZALECANA LITERATURA**

Proponowane podręczniki:

1. R. Sama, K. Sama, Język angielski zawodowy w branży informatycznej, wyd. WSiP, Warszawa 2016.

# Kwalifikacja INF.04. Projektowanie, programowanie i testowanie aplikacji

## Podstawy programowania

**Cele ogólne przedmiotu**

* 1. Poznanie
* pojęć związanych ze środowiskiem programistycznym: kompilator, interpreter, debuger,
* metod i zasad tworzenia algorytmów,
* algorytmicznego analizowania problemu,
* prostych i złożonych typów danych,
* paradygmatów programowania strukturalnego i obiektowego,
* pojęć klasa, pole, metoda, obiekt, konstruktor, destruktor, dziedziczenie, składowe statyczne, funkcje i klasy zaprzyjaźnione, metody wirtualne, klasy abstrakcyjne, wyjątek,
  1. Nabycie umiejętności
* tworzenia algorytmów,
* zapisywania gotowych algorytmów w języku programowania,
* doboru algorytmu do problemu programistycznego,
* rozpoznawania i definiowania typów danych,
* określania paradygmatów programowania strukturalnego i obiektowego,
* definiowania klas i obiektów
  1. Kształtowanie postawy, świadomości
* aktywnego słuchania i prowadzenia dyskusji;
* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
* wrażliwości na potrzeby osób niepełnosprawnych;
* odpowiedzialności za podejmowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;

**Cele operacyjne**

**Uczeń potrafi:**

1. rozróżniać kompilatory i interpretery,
2. scharakteryzować zadania kompilatora, interpretera, debuggera,
3. scharakteryzować pojęcie biblioteki,
4. scharakteryzować etapy kompilacji i interpretacji kodu,
5. zaprojektować algorytmy za pomocą różnych metod: schematów blokowych, listy kroków, drzew decyzyjnych, pseudokodu,
6. scharakteryzować algorytmy iteracyjne,
7. scharakteryzować metody sortowania i ich złożoność obliczeniową
8. opisać rodzaje prostych typów danych,
9. zidentyfikować zmienną typu prostego i określić jej rodzaj (numeryczny stałoprzecinkowy i zmiennoprzecinkowy, logiczny, znakowy, łańcuchowy),
10. podać przykłady wykorzystania danych typu prostego,
11. deklarować zmienne prostych typów danych,
12. podać przykłady operacji na zmiennych: wejścia i wyjścia, arytmetyczne, logiczne,
13. podać przykłady wyrażeń z operatorami arytmetycznymi, przypisania, porównania, logicznymi, operatorami do obsługi łańcuchów, bitowymi oraz wykorzystujących priorytety operatorów,
14. omówić rodzaje złożonych typów danych,
15. identyfikować dane typu złożonego i określić ich rodzaj (tablice jednowymiarowe i dwuwymiarowe, plik, rekord, np. struktura, unia),
16. opisać własności złożonych typów danych
17. identyfikować tablice dynamiczne, asocjacyjne,
18. identyfikować typ wskaźnikowy,
19. określić własne typy danych dla problemu programistycznego,
20. podać przykłady wykorzystania złożonych typów danych (tablice, rekordy, pliki, typ wskaźnikowy),
21. określić składnię instrukcji warunkowej i wyboru,
22. określić składnię instrukcji pętli,
23. podać definicję funkcji i opisać jej prototyp,
24. objaśnić sposoby przekazywania argumentów funkcji,
25. zapisywać program w postaci zestawu funkcji,
26. identyfikować wybrane biblioteki języka C++, C#, Python lub innego języka programowania: standardowej, z funkcjami matematycznymi, z podstawowymi algorytmami,
27. zapisać algorytmy w języku programowania,
28. objaśnić pojęcie rekurencji,
29. zdefiniować funkcję rekurencyjną,
30. zastosować paradygmaty programowania obiektowego,
31. wyjaśnić pojęcia klasa, obiekt, metoda, pole, konstruktor, destruktor, dziedziczenie, hermetyzacja, polimorfizm,
32. tworzyć obiekty jako instancje klasy,
33. odwołać się do pól i metod obiektu,
34. definiować pola (właściwości) klasy,
35. definiować metody klasy,
36. określić zakres widoczności składowych klasy za pomocą specyfikatorów dostępu,
37. zdefiniować konstruktor, w tym konstruktor kopiujący, i destruktor klasy,
38. deklarować obiekty,
39. objaśnić składniki statyczne klasy i je zdefiniować,
40. definiować klasy zaprzyjaźnione,
41. tworzyć funkcje zaprzyjaźnione z klasą,
42. podać przykład wykorzystania składowych statycznych klasy i metod do ich obsługi,
43. definiować klasy bazowe i pochodne,
44. określić przynależność metod i pól do odpowiednich klas w hierarchii dziedziczenia,
45. rozróżnić klasy dziedziczone,
46. projektować programy wykorzystujące hierarchię dziedziczenia klas,
47. scharakteryzować metody wirtualne, zdefiniować klasy abstrakcyjne,
48. zaprojektować aplikację korzystającą z hermetyzacji, dziedziczenia i polimorfizmu,
49. wyjaśnić rolę szablonów klas w programowaniu obiektowym,
50. określić szablony klas dla prostych typów liczbowych,
51. określić mechanizm obsługi wyjątków z instrukcjami try i catch,
52. zgłaszać wyjątki do obsłużenia (instrukcja throw),
53. sklasyfikować możliwe błędy wykonania aplikacji,
54. określić obsługę błędów wykonania aplikacji,
55. scharakteryzować algorytmy tekstowe i szyfrowania, tablicowe,
56. scharakteryzować algorytmy rekurencyjne,
57. zapisać w języku programowania różne algorytmy sortowania, np. bąbelkowe, przez wstawianie, zachłanne, szybkie, metodą dziel i zwyciężaj
58. scharakteryzować algorytmy np. heurystyczne, problem komiwojażera,
59. określić złożoność obliczeniową algorytmów,
60. ocenić efektywność różnych algorytmów sortowania,
61. omówić algorytmy wyszukiwania dla różnych typów danych

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Algorytm, typy danych i instrukcje** | 1. Środowisko programistyczne |  | rozróżniać kompilatory i interpretery,  charakteryzować zadania kompilatora, interpretera, debuggera,  charakteryzować pojęcie biblioteki | charakteryzować etapy kompilacji i interpretacji kodu | Klasa I |
| 1. Algorytmy sortowania |  | zaprojektować proste algorytmy za pomocą różnych metod: schematów blokowych, listy kroków, drzew decyzyjnych, pseudokodu,  scharakteryzować algorytmy iteracyjne,  scharakteryzować typy sortowania i ich złożoność obliczeniową | zaprojektować złożone algorytmy za pomocą różnych metod: schematów blokowych, listy kroków, drzew decyzyjnych, pseudokodu | Klasa I |
| 1. Proste typy danych |  | opisać rodzaje prostych typów danych,  zidentyfikować zmienną typu prostego i określić jej rodzaj (numeryczny stałoprzecinkowy i zmiennoprzecinkowy,  logiczny, znakowy, łańcuchowy,  podać przykłady wykorzystania danych typu prostego |  | Klasa I |
| 1. Operatory i instrukcja przypisania |  | deklarować zmienne prostych typów danych,  podać przykłady operacji na zmiennych: wejścia i wyjścia, arytmetyczne, logiczne,  podać przykłady wyrażeń z operatorami arytmetycznymi, przypisania, porównania, logicznymi, operatorami do obsługi łańcuchów, bitowymi | podać przykłady wyrażeń wykorzystujących priorytety operatorów | Klasa I |
| 1. Złożone typy danych |  | omówić rodzaje złożonych typów danych,  identyfikować dane typu złożonego i określić ich rodzaj (tablice jednowymiarowe i dwuwymiarowe, rekord, np. struktura, unia, plik),  opisać własności złożonych typów danych | identyfikować tablice dynamiczne, asocjacyjne,  identyfikować typ wskaźnikowy,  określać własne typy danych dla problemu programistycznego,  podać przykłady wykorzystania złożonych typów danych (tablice, rekordy, pliki, typ wskaźnikowy) | Klasa I |
| 1. Instrukcje sterujące |  | określić składnię instrukcji warunkowej i wyboru,  określić składnię instrukcji pętli |  | Klasa I |
| 1. Funkcje. Funkcje biblioteczne |  | podać definicję funkcji i opisać prototyp funkcji,  zapisywać program w postaci zestawu funkcji,  identyfikować wybrane biblioteki języka C++, C#, Python lub innego języka programowania: standardowej, z funkcjami matematycznymi, z podstawowymi algorytmami | objaśnić sposoby przekazywania argumentów funkcji | Klasa II |
| 1. Zasady programowania |  | zapisywać proste algorytmy w języku programowania | objaśnić pojęcie rekurencji,  definiować funkcję rekurencyjną,  zapisywać złożone algorytmy w języku programowania | Klasa II |
| **II. Obiekto-wość** | 1. Zasady programowania obiektowego |  | stosować paradygmaty programowania obiektowego,  wyjaśnić pojęcia klasa, obiekt, metoda, pole, dziedziczenie, hermetyzacja, polimorfizm,  zdefiniować klasę jako reprezentację obiektu ze świata rzeczywistego,  tworzyć obiekty jako instancje klasy,  odwołać się do pól i metod obiektu |  | Klasa II |
| 1. Klasy i obiekty |  | definiować pola(właściwości) klasy,  definiować metody klasy,  określać zakres widoczności składowych klasy za pomocą specyfikatorów dostępu,  zdefiniować konstruktor z instrukcjami inicjującymi, w tym konstruktor kopiujący, i destruktor klasy,  deklarować obiekty,  objaśnić składniki statyczne klasy i je zdefiniować | definiować klasy zaprzyjaźnione,  tworzyć funkcje zaprzyjaźnione z klasą,  podać przykład wykorzystania składowych statycznych klasy i metod do ich obsługi | Klasa II |
| 1. Dziedziczenie klas |  | wyjaśnić pojęcia klasy bazowej, klasy pochodnej, dziedziczenia prostego i złożonego (wielokrotnego), polimorfizmu,  definiować klasy bazowe i pochodne  projektować programy wykorzystujące hierarchię dziedziczenia klas,  rozróżnić klasy dziedziczone, | określić przynależność metod i pól do odpowiednich klas w hierarchii dziedziczenia,  scharakteryzować metody wirtualne,  zdefiniować klasy abstrakcyjne,  zaprojektować aplikację korzystającą z hermetyzacji, dziedziczenia i polimorfizmu | Klasa III |
| 1. Szablony (wzorce) klas |  | wyjaśnić rolę szablonów klas w programowaniu obiektowym | określać szablony klas dla prostych typów liczbowych | Klasa III |
| 1. Obsługa wyjątków |  | określić mechanizm obsługi wyjątków z instrukcjami try i catch,  zgłaszać wyjątki do obsłużenia (instrukcja throw), | sklasyfikować możliwe błędy wykonania aplikacji,  określić obsługę błędów wykonania aplikacji | Klasa III |
| **III. Projektowa-nie algorytmów** | 1. Projektowanie algorytmów |  | scharakteryzować algorytmy tekstowe i szyfrowania, tablicowe,  scharakteryzować algorytmy rekurencyjne,  zapisać w języku programowania algorytmy sortowania, np. bąbelkowe, przez wstawianie | scharakteryzować algorytmy np. heurystyczne, problem komiwojażera,  określić złożoność obliczeniową algorytmów,  opisać problem sortowania różnymi metodami, np. zachłanne, szybkie, metodą dziel i zwyciężaj i zapisać go w języku programowania,  ocenić efektywność różnych algorytmów sortowania,  omówić algorytmy wyszukiwania dla różnych typów danych | Klasa III |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Do osiągania założonych celów kształcenia w zakresie przedmiotu Podstawy programowania konieczne jest

* zaplanowanie lekcji poprzez wskazanie celów szczegółowych,
* wykorzystanie różnorodnych metod nauczania, w szczególności aktywizujących,
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany, kartkówki, testy i projekty,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

Środki dydaktyczne

W sali lekcyjnej lub pracowni, w której prowadzone będą zajęcia powinno znajdować się stanowisko komputerowe dla nauczyciela, z odpowiednim oprogramowaniem (oprogramowanie biurowe, różne środowiska programistyczne – edytor, kompilator, biblioteki, frameworki), podłączone do sieci lokalnej z dostępem do internetu z urządzeniem wielofunkcyjnym oraz z projektorem multimedialnym lub tablicą interaktywną, ekran, głośniki, wskazane są także stanowiska komputerowe dla uczniów (jedno stanowisko dla maksimum dwóch uczniów) z odpowiednim oprogramowaniem (takim jak nauczyciela). W pracowni, w której prowadzone będą zajęcia powinny się znajdować: plansze poglądowe, filmy dydaktyczne i prezentacje multimedialne związane z treściami kształcenia, dokumentacje języków programowania, dokumentacja oprogramowania w wersji papierowej i elektronicznej.

Zalecane metody dydaktyczne

Dla przedmiotu Podstawy programowania, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych, opartych na słowie i obserwacji, takich jak:

* wykład informacyjny,
* pogadanka,
* pokaz z objaśnieniem,
* wykład problemowy,
* dyskusja dydaktyczna,
* burza mózgów,
* ćwiczenia praktyczne,

Zajęcia mogą odbywać się w grupach. Istotną metodą kształcenia powinny być także ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku.

Formy organizacyjne

Zajęcia z przedmiotu Podstawy programowania powinny być prowadzone indywidualnie, zespołowo lub zbiorowo. Należą one do grupy zajęć teoretycznych. W przypadku realizacji ćwiczeń praktycznych powinny być stosowane formy organizacyjne indywidualne. W kształceniu zawodowym bardzo ważną kwestią jest dostosowanie wymagań edukacyjnych do zróżnicowanych potrzeb i możliwości uczniów. Ponadto uczniowie powinni samodzielnie zdobywać wiedzę i kształtować umiejętności poprzez uczenie się we współpracy oraz korzystanie z różnych źródeł informacji.

Formy indywidualizacji pracy uczniów

W celu dostosowania warunków, środków, metod i form kształcenia do potrzeb i możliwości ucznia można w zakresie organizacji pracy zastosować instrukcje do zadań, podawanie dodatkowych zaleceń, objaśnień, instrukcji do pracy indywidualnej, udzielanie konsultacji indywidualnych. Nauczyciel powinien dostosować stopień trudności materiału do możliwości i potrzeb uczniów. W pracy grupowej należy zwracać uwagę na taki podział zadań między członków zespołu, by każdy wykonywał tę część zadania, której podoła. Uczniom szczególnie zdolnym i posiadającym określone zainteresowania zawodowe należy zaplanować zadania o większym stopniu trudności, proponować samodzielne poszerzanie wiedzy, korzystanie z dodatkowej literatury.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

W celu sprawdzenia osiągnięć edukacyjnych ucznia proponuje się: testy wielokrotnego wyboru, testy zawierające zadania otwarte, kartkówki, prezentacje uczniów, projekty.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu, na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* realizowanych ćwiczeń praktycznych w formie zadań,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń,
* efektu projektu i jego prezentacji,

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (tutoriali, dokumentacji danego języka – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest też, by uczniowie dokonywali samooceny i oceny kolegów z zespołu według zaproponowanych przez nauczyciela kryteriów.

Stosowane przez nauczyciela ocenianie powinno korzystać z zasad występujących w ocenianiu kształtującym, ma być dla ucznia informacją zwrotną, która pomaga mu się uczyć. Nauczyciel wskazuje silne strony ucznia i doradza, jak je rozwijać; wyraźnie i konstruktywnie informuje o stronach słabych i o tym, jak można je eliminować; stwarza uczniom możliwość poprawienia własnej pracy. Ocenianie skupiające się na postępach i osiągnięciach, a nie na podkreślaniu niepowodzeń, zachęca uczniów do uczenia się. Porównywanie osiągnięć poszczególnych uczniów z osiągnięciami ich kolegów, tworzenie rankingów, nie motywuje, a często zniechęca do uczenia się. Nauczyciel korzysta też z oceniania sumującego, nie rezygnuje ze stopni.

**Przykładowe zadania**

1. Zaprojektuj algorytm, który realizuje następujące zadania:
   1. wczytuje dwie liczby całkowite dol, gora, będące zakresem,
   2. wczytuje różnicę ciągu arytmetycznego r,
   3. oblicza i wyświetla kolejne wyrazy ciągu arytmetycznego mieszczące się w zakresie od wartości dol do gora,
   4. sumuje wyświetlane wyrazy ciągu arytmetycznego i obliczoną sumę wszystkich wyświetla.
2. Opisz paradygmat algorytmiczny dziel i zwyciężaj i przedstaw jego zastosowanie do sortowania elementów tablicy [listy].
3. Zdefiniuj klasę o nazwie Temperatura, która reprezentuje temperaturę otoczenia. Przyjmij znakową skalę temperatur:

c – w stopniach Celsjusza

k – w Kelwinach

f – w stopniach Fahrenheita

Diagram klasy:

|  |
| --- |
| Temperatura |
| - stopnieC: double |
| + setTemperatura(double temp, char skala): void  + getTemperatura(char skala): double |

* „-” to składowa prywatna, „+” to składowa publiczna,
* w polu klasy stopnieC temperatura przechowywana jest wyłącznie w stopniach Celsjusza,
* metody dostępowe set i get odpowiednio zapisują do i odczytują z pola wartość temperatury,
* argumentami metody dostępowej set są: temperatura w dowolnej skali i skala temperatur,
* argumentem metody dostępowej get jest skala temperatur,
* przeliczanie temperatury na stopnie Celsjusza odbywa się wg zależności:
  + ze stopni Fahrenheita na stopnie Celsjusza: stopnieC = (5.0 / 9.0) \* (stopnieF – 32)
  + ze stopni Kelwina na stopnie Celsjusza: stopnieC = stopnieK – 273.15,

gdzie stopnieC, stopnieF, stopnieK – zmienne przechowujące temperaturę w danej skali, odpowiednio Celsjusza, Fahrenheita, Kelwina,

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Koncepcja ewaluacji przedmiotu polegać będzie na tzw. twardej analizie danych, czyli ocen zdobytych przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych oraz z przygotowanych przez uczniów projektów. Zebrane wyniki zostaną poddane analizie ilościowej i jakościowej z wykorzystaniem statystyki matematycznej.

Uzyskane informacje pozwolą na wyodrębnienie zagadnień sprawiających uczniom problemy. W następstwie, będzie można zwiększyć liczbę godzin dydaktycznych przypisanych do danego działu programowego. Korekta taka powinna się przyczynić do podwyższenia jakości kształcenia i poprawy indywidualnych wyników uczniów z egzaminu w danej kwalifikacji.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

* tworzenia algorytmów,
* zapisywania algorytmów w języku programowania
* rozpoznawania i definiowania typów danych,
* określania paradygmatów programowania strukturalnego i obiektowego,
* definiowania klas i obiektów

**Literatura do przedmiotu**

1. Bjarne Stroustrup, Język C++. Kompendium wiedzy, wyd. Helion,
2. Stephen Prata, Język C++. Szkoła programowania. Wydanie VI, wyd. Helion,
3. Grębosz Jerzy, Opus magnum C++11, Programowanie w języku C++ (komplet), Wyd.: Grębosz Jerzy,

#### Python dla każdego. Podstawy programowania – Michael Dawson,wyd. Helion,

#### Eric Matthes, Python. Instrukcje dla programisty, wyd. Helion,

#### Al Sweigart, Automatyzacja nudnych zadań z Pythonem, wyd. Helion,

1. http://www.cplusplus.com/
2. https://docs.python.org/3/tutorial/
3. http://codecademy.com
4. http://w3schools.com
5. https://doc.qt.io/qtcreator/index.html
6. http://www-cs.ccny.cuny.edu/~wolberg/cs221/qt/books/C++-GUI-Programming-with-Qt-4-1st-ed.pdf
7. https://qmlbook.github.io/assets/qt5\_cadaques.pdf

## Tworzenie aplikacji desktopowych

**Cele ogólne przedmiotu**

1. Poznanie

* zagrożeń w środowisku pracy i wymogów w zakresie organizacji stanowiska pracy,
* zasad udzielania pierwszej pomocy,
* algorytmów,
* środowisk programistycznych,
* struktur danych,
* paradygmatów programowania strukturalnego i obiektowego,
* zasad tworzenia okienkowych aplikacji desktopowych,
* zasad testowania i dokumentowania aplikacji

1. Nabycie umiejętności

* bezpiecznego wykonywania zadań zawodowych,
* stosowania w programowaniu obiektowości,
* samodzielnego tworzenia poprawnych programów desktopowych,
* wyboru algorytmów do rozwiązania zadań,
* tworzenia programów według dostarczonego projektu,
* testowania i optymalizacji działania programu,
* dokumentowania programu,

1. Kształtowanie postawy, świadomości

* aktywnego słuchania i prowadzenia dyskusji;
* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
* wrażliwości na potrzeby osób niepełnosprawnych;
* odpowiedzialności za podejmowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;
* doskonalenia umiejętności zawodowych,
* pracy w zespole;

**Cele operacyjne**

**Uczeń potrafi:**

1. omówić zagrożenia występujące w środowisku pracy,
2. omówić skutki oddziaływania czynników fizycznych i psychofizycznych na organizm człowieka,
3. scharakteryzować czynniki niebezpieczne i uciążliwe i ich skutki na organizm człowieka,
4. zdefiniować pojęcie choroby zawodowej i wypadku przy pracy,
5. scharakteryzować środki ochrony zbiorowej,
6. wyliczyć środki ochrony zabezpieczające przed hałasem w pracy biurowej,
7. opisać wymagania w zakresie oświetlenia, temperatury i mikroklimatu pomieszczeń biurowych,
8. rozpoznać środki ochrony zapobiegające porażeniem prądem w pracy biurowej,
9. rozpoznać środki ochrony zapobiegające pogorszeniu wzroku i zniekształceniu kręgosłupa,
10. dobrać środki ochrony zbiorowej do rodzaju zagrożeń w pracy biurowej,
11. opisać podstawowe symptomy wskazujące na stany nagłego zagrożenia zdrowotnego,
12. opisać zasady oceniania sytuacji poszkodowanego na podstawie analizy obserwowanych u niego objawów,
13. opisać zasady zabezpieczenia siebie, poszkodowanego i miejsca wypadku,
14. zaprezentować ułożenie poszkodowanego w pozycji bezpiecznej,
15. wymienić służby, które powiadamia się w razie wypadku,
16. wykonać resuscytację krążeniowo-oddechową na fantomie zgodnie z wytycznymi Polskiej Rady Resuscytacji i Europejskiej Rady Resuscytacji,
17. określić działania wykonywane podczas udzielania pierwszej pomocy w urazowych stanach nagłego zagrożenia zdrowotnego, np. krwotok, zmiażdżenie, amputacja, złamanie, oparzenie,
18. określić działania wykonywane podczas udzielania pierwszej pomocy w nieurazowych stanach nagłego zagrożenia zdrowotnego, np. omdlenie, zawał, udar
19. scharakteryzować środowisko programistyczne(np. Visual Studio, CodeBlocks),
20. dobrać środowisko do programowania w wybranym języku (C++, C#, Python, Java, Visual Basic),
21. rozpoznać narzędzia wykorzystywane w procesie tworzenia aplikacji desktopowych
22. stosować typy proste: liczbowe stało- i zmiennoprzecinkowe,logiczny, znakowy i łańcuchowy (string),
23. deklarować zmienne,
24. wczytać i wyświetlić wartość zmiennej,
25. konstruować wyrażenia arytmetyczne i logiczne z uwzględnieniem hierarchii operatorów,
26. wykorzystywać w programach tablice jedno- i dwuwymiarowe,
27. wykonać operacje na zmiennych typu prostego,
28. definiować i stosować typ rekordowy (struktura, unia),
29. zastosować typ plikowy do obsługi plików,
30. wykorzystywać w programach tablice dynamiczne i asocjacyjne,
31. zastosować typ wskaźnikowy i zmienne dynamicznie,
32. zastosować kolekcje (stosy, kolejki, listy, wektory),
33. dobrać typ zmiennej do reprezentowania wartości w programie,
34. wykonać w programie operacje przypisania, arytmetyczne, porównania, logiczne, bitowe, obsługi łańcuchów,
35. stosować rozgałęzienia w programie za pomocą instrukcji warunkowych if, [switch],
36. realizować powtórzenia w programie za pomocą pętli for, while, [do while],
37. programować wykorzystując wybrane biblioteki języka C++, C#, Python lub innego języka programowania: biblioteka standardowa, biblioteka z funkcjami matematycznymi, biblioteka z podstawowymi algorytmami
38. kompilować i uruchamiać programy,
39. analizować błędy w kodzie za pomocą debuggera,
40. tworzyć program z podziałem na funkcje (metody),
41. implementować algorytmy w programie,
42. definiować funkcje rekurencyjne,
43. zastosować rekurencję do realizacji powtórzeń,
44. stosować algorytm wyszukiwania dla różnych zestawów danych (tablic, list, kolejek, stosów),
45. objaśnić pojęcie frameworka,
46. zastosować framework Qt do tworzenia aplikacji desktopowych w języku C++ (framework WPF – do tworzenia aplikacji desktopowych w języku C#),
47. projektować interfejs użytkownika i wygląd aplikacji wykorzystując odpowiednie elementy (widżety, np. okna dialogowe, przyciski, paski narzędziowe),
48. projektować funkcjonalność aplikacji ,
49. planować system zabezpieczeń aplikacji,
50. projektować struktury danych dla aplikacji,
51. odzwierciedlać rzeczywistość w formie zbioru obiektów,
52. definiować klasy i tworzyć obiekty,
53. definiować składowe klasy – pola i metody,
54. dobrać specyfikatory dostępu (public, private) dla składowych klasy,
55. definiować konstruktory, w tym kopiujące i destruktory,
56. odwoływać się poprzez obiekt do składowych klasy,
57. wykorzystywać w programach składowe statyczne klasy,
58. tworzyć funkcje zaprzyjaźnione z klasą,
59. tworzyć klasy zaprzyjaźnione,
60. tworzyć klasę pochodną do danej klasy,
61. projektować program z zastosowaniem hierarchii dziedziczenia klas,
62. określić przynależność składowych klas (pól i metod) do odpowiednich klas w hierarchii dziedziczenia,
63. definiować konstruktory klas pochodnych,
64. definiować klasy bazowe i pochodne,
65. wykorzystać mechanizm przeciążania (przeładowania) metod klasy,
66. definiować metody wirtualne,
67. tworzyć klasy abstrakcyjne,
68. zastosować w programie hermetyzację, dziedziczenie, polimorfizm,
69. wyjaśnić rolę szablonów klas w programowaniu obiektowym,
70. opisywać szablony klas dla prostych typów liczbowych,
71. stosować mechanizm obsługi wyjątków z instrukcjami try i catch,
72. zgłaszać wyjątki do obsłużenia (instrukcja throw),
73. sklasyfikować możliwe błędy wykonania aplikacji,
74. zrealizować obsługę błędów wykonania aplikacji,
75. stosować strategie organizowania okien w ramach pulpitu (SDI, MDI),
76. zastosować funkcje jednego z języków C++, C#, Java Python do tworzenia aplikacji desktopowych,
77. zainstalować oprogramowanie do tworzenia GUI, np. Qt Creator,
78. tworzyć okno aplikacji np. z wykorzystaniem Qt Quick,
79. zaprojektować interfejs użytkownika,
80. stworzyć projekt i strukturę aplikacji,
81. stworzyć okno aplikacji z ikoną programu, przyciskami maksymalizuj, minimalizuj,
82. wstawić podstawowe widżety (kontrolki), np.: pole edycji, suwak, przycisk do okna aplikacji,
83. rozmieszczć widżety w oknie aplikacji,
84. przypisać widżetom zasady funkcjonowania,
85. zaprojektować obsługę zdarzeń myszy i klawiatury,
86. zaprogramować obsługę zdarzeń myszy i klawiatury,
87. wstawić do aplikacji kod źródłowy,
88. stosować predefiniowane okna dialogowe (np. wybór pliku, czcionki, koloru),
89. zaprojektować i dodać menu do aplikacji,
90. implementować opcje menu np. Plik, Edycja,
91. utworzyć interfejs użytkownika,
92. stworzyć klasyczne okno aplikacji desktopowej (pasek menu, pasek narzędzi, obszar dokumentu, pasek statusu)
93. stosować język do projektowania interfejsu użytkownika, np. QML, w tym np.: stosować podstawowe elementy wizualne i niewizualne, wykonać transformację obiektu (np. przesunięcie, obrót, skalowanie), pozycjonować elementy, animować właściwości elementów, np.: kolor, obrót, przejście,
94. zaprojektować własne okna dialogowe,
95. utworzyć okienkowe aplikacje desktopowe, np. przeglądarka zdjęć, notatnik,
96. korzystać z elementów multimedialnych w aplikacji,
97. przechowywać dane aplikacji,
98. dynamiczne ładować elementy,
99. dobrać narzędzia i środowisko do testowania aplikacji,
100. dokonać statycznej analizy kodu w celu znalezienia nieefektywnych konstrukcji oraz fragmentów kodu,
101. przeprowadzić testy programów,
102. usunąć błędy i niedoskonałości ze swoich kodów źródłowych,
103. optymalizować kod źródłowy,
104. umiescić komentarze w kodzie źródłowym programu,
105. stosować w kodzie źródłowym komentarze typu DocBlocks,
106. tworzyć dokumentację techniczną aplikacji,
107. tworzyć pliki pomocy,
108. tworzyć instrukcję użytkownika programu,
109. opracować dokumentację wdrożenia projektu,
110. opracować dokumentację z przeprowadzonych testów aplikacji,
111. dokonać podziału testów (np. względu na weryfikowane obiekty, ze względu na metodę weryfikacji, bazujące na wymaganiach),
112. zaplanować fazy testowania,
113. przeprowadzać testy w kolejnych fazach projektu informatycznego, przeprowadzać testy funkcjonalne,
114. stosować narzędzia do automatyzacji procesu testowania,
115. korzystać z systemów raportowania błędów, np. BugZilla, JIRA,
116. przeprowadzać testy interfejsu,
117. testować prototyp projektu interfejsu,
118. przeprowadzać testy niefunkcjonalne: użyteczności, wydajnościowe, obciążeniowe, zgodności, bezpieczeństwa,
119. przygotować środowisko testowe,
120. tworzyć scenariusze testowania aplikacji,
121. aktywnie słuchać ,włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,
122. stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;
123. zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;
124. wykazywać się kreatywnością w rozwiązywaniu problemów,
125. ponosić odpowiedzialność za podejmowane działania,
126. dokonać analizy i oceny podejmowanych działań,
127. doskonalić jakość wykonywanych działań;
128. doskonalić umiejętności zawodowe,
129. pracować w zespole.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. BHP** | 1. Zagrożenia w środowisku pracy |  | omówić zagrożenia występujące w środowisku pracy,  omówić skutki oddziaływania czynników fizycznych (np. oświetlenie, hałas, mikroklimat, promieniowanie jonizujące i elektromagnetyczne) na organizm człowieka ,  omówić skutki oddziaływania czynników psychofizycznych (obciążenie fizyczne oraz obciążenie nerwowo–psychiczne) na organizm człowieka,  charakteryzować czynniki niebezpieczne i uciążliwe  zdefiniować pojęcie choroby zawodowej i wypadku przy pracy | opisać skutki oddziaływania czynników niebezpiecznych i uciążliwych na organizm człowieka | Klasa I |
| 1. Środki ochrony zbiorowej |  | scharakteryzować środki ochrony zbiorowej,  wyliczyć środki ochrony zabezpieczające przed hałasem w pracy biurowej,  opisać wymagania w zakresie oświetlenia, temperatury i mikroklimatu pomieszczeń biurowych,  rozpoznać środki ochrony zapobiegające porażeniem prądem w pracy biurowej,  rozpoznać środki ochrony zapobiegające pogorszeniu wzroku i zniekształceniu kręgosłupa | dobrać środki ochrony zbiorowej do rodzaju zagrożeń w pracy biurowej | Klasa I |
| 1. Pierwsza pomoc |  | opisać podstawowe symptomy wskazujące na stany nagłego zagrożenia zdrowotnego,  opisać zasady oceniania sytuacji poszkodowanego na podstawie analizy obserwowanych u niego objawów,  opisać zasady zabezpieczenia siebie, poszkodowanego i miejsca wypadku,  zaprezentować ułożenie poszkodowanego w pozycji bezpiecznej,  wymienić służby, które powiadamia się w razie wypadku,  wykonać resuscytację krążeniowo-oddechową na fantomie zgodnie z wytycznymi Polskiej Rady Resuscytacji i Europejskiej Rady Resuscytacji | określić działania wykonywane podczas udzielania pierwszej pomocy w urazowych stanach nagłego zagrożenia zdrowotnego, np. krwotok, zmiażdżenie, amputacja, złamanie, oparzenie,  określić działania wykonywane podczas udzielania pierwszej pomocy w nieurazowych stanach nagłego zagrożenia zdrowotnego, np. omdlenie, zawał, udar |  |
| **II. Wstep do programowania i algorytmika** | 1. Środowisko programistyczne dla aplikacji desktopowych |  | dobrać środowisko programistyczne do programowania w wybranym języku (np. Visual Studio, CodeBlocks),  stosować środowisko programistyczne,  rozpoznać narzędzia wykorzystywane w procesie tworzenia aplikacji desktopowych |  | Klasa I |
| 1. Typy danych |  | stosować typy liczbowe stało- i zmiennoprzecinkowe,  wykorzystać typ logiczny,  stosować typ znakowy i łańcuchowy (string),  deklarować zmienne,  wczytać i wyświetlić wartość zmiennej,  tworzyć wyrażenia arytmetyczne,  tworzyć warunki logiczne,  wykorzystywać w programach tablice jedno- i dwuwymiarowe,  wykonać operacje na zmiennych prostego typu  definiować i stosować typ rekordowy (struktura, unia),  zastosować typ plikowy do obsługi plików | wykorzystywać w programach tablice dynamiczne i asocjacyjne  stosować typ wskaźnikowy i zmienne dynamicznie,  stosować kolekcje (stosy, kolejki, listy, wektory),  dobrać typ zmiennej do reprezentowania wartości w programie,  wykonać operacje na zmiennych złożonego typu |  |
| 1. Operatory , instrukcje sterujące, funkcje biblioteczne |  | Wykonać w programie operacje przypisania, arytmetyczne, porównania, logiczne, bitowe, obsługi łańcuchów,  stosować rozgałęzienia w programie za pomocą instrukcji warunkowych if, [switch],  realizować powtórzenia w programie za pomocą pętli for, while, [do while],  programować wykorzystując wybrane biblioteki języka C++, C#, Python lub innego języka programowania: biblioteka standardowa, biblioteka z funkcjami matematycznymi, biblioteka z podstawowymi algorytmami | konstruować wyrażenia arytmetyczne i logiczne z uwzględnieniem hierarchii operatorów, | Klasa I |
| Środowisko programistyczne dla obiektowych aplikacji konsolowych |  | kompilować i uruchamiać programy | analizować błędy w kodzie za pomocą debuggera | Klasa I |
| Zasady programowania |  | tworzyć program z podziałem na funkcje (metody),  implementować proste algorytmy w programie | definiować funkcje rekurencyjne,  zastosować rekurencję do realizacji powtórzeń,  implementować złożone algorytmy w programie | Klasa II |
| Algorytmy wyszukiwania |  | stosować algorytm wyszukiwania dla różnych zestawów danych (tablic, list, kolejek, stosów) |  | Klasa II |
| Wykorzystanie frameworków do programowania aplikacji desktopowych |  | objaśnić pojęcie frameworka,  zastosować framework do tworzenia prostych aplikacji desktopowych np. Qt w języku C++ (lub frameworka WPF – w języku C#) | zastosować framework do tworzenia prostych aplikacji desktopowych np. Qt w języku C++ (lub frameworka WPF – w języku C#) | Klasa II |
| Projektowanie aplikacji |  | projektować interfejs użytkownika i wygląd aplikacji wykorzystując odpowiednie elementy (widżety, np. okna dialogowe, przyciski, paski narzędziowe) ,  projektować funkcjonalność aplikacji ,  planować system zabezpieczeń aplikacji | projektować struktury danych dla aplikacji | Klasa II |
| Zasady programowania obiektowego |  | odzwierciedlać rzeczywistość w formie zbioru obiektów,  definiować klasy i tworzyć obiekty,  stosować specyfikatory dostępu, | planować aplikację z zastosowaniem hermetyzacji | Klasa II |
| **III. Programo-wanie obiektowe** | Klasy i obiekty |  | definiować składowe klasy – pola i metody,  dobierać specyfikatory dostępu (public, private) dla składowych klasy,  definiować konstruktory z instrukcjami inicjującymi, w tym kopiujące i destruktory klasy  implementować funkcjonalność klasy,  tworzyć obiekty,  odwoływać się poprzez obiekt do składowych klasy | wykorzystywać w programach składowe statyczne klasy,  tworzyć funkcje zaprzyjaźnione z klasą,  tworzyć klasy zaprzyjaźnione,  używać specyfikatorów dziedziczenia,  tworzyć klasę pochodną do danej klasy | Klasa III |
| Dziedziczenie |  | projektować program z zastosowaniem hierarchii dziedziczenia klas,  określać przynależność składowych klas (pól i metod) do odpowiednich klas w hierarchii dziedziczenia,  definiować konstruktory klas pochodnych  definiować klasy bazowe i pochodne,  wykorzystać mechanizm przeciążania (przeładowania) metod klasy | definiować metody wirtualne,  tworzyć klasy abstrakcyjne,  zastosować w programie, hermetyzację, dziedziczenie, polimorfizm | Klasa III |
| Szablony (wzorce) klas |  | wyjaśnić rolę szablonów klas w programowaniu obiektowym, | opisywać szablony klas dla prostych typów liczbowych | Klasa III |
| Obsługa wyjątków |  | stosować mechanizm obsługi wyjątków z instrukcjami try i catch,  zgłaszać wyjątki do obsłużenia (instrukcja throw), | sklasyfikować możliwe błędy wykonania aplikacji,  zrealizować obsługę błędów wykonania aplikacji | Klasa III |
| **IV. Wprowa-dzenie do programo-wania desktopo-wych aplikacji okienko-wych** | Wprowadzenie do programowania desktopowych aplikacji okienkowych |  | stosować strategie organizowania okien w ramach pulpitu(SDI, MDI),  zainstalować oprogramowanie do tworzenia GUI, np. Qt Creator (C++),  tworzyć okna aplikacji np. z wykorzystaniem Qt Quick, np. stworzyć okno aplikacji z ikoną programu, przyciskami maksymalizuj, minimalizuj,  zaprojektować interfejs użytkownika,  utworzyć interfejs uzytkownika,  zaprojektować obsługę zdarzeń myszy i klawiatury,  wstawić do okna aplikacji podstawowe widżety (kontrolki), np.: pole edycji, suwak, przycisk,  ułożyć widżety w oknie aplikacji zaprogramować obsługę zdarzeń myszy i klawiatury, | stworzyć klasyczne okno aplikacji desktopowej (pasek menu, pasek narzędzi, obszar dokumentu, pasek statusu), np. przeglądarka zdjęć | Klasa III |
| **V.Testo-wanie i dokumen-towanie aplikacji** | Walidacja kodu programu |  | dobrać narzędzia i środowisko do testowania aplikacji,  dokonać statycznej analizy kodu w celu znalezienia nieefektywnych konstrukcji oraz fragmentów kodu,  przeprowadzić testy programów | usuwać błędy i niedoskonałości ze swoich kodów źródłowych,  optymalizować kod źródłowy | Klasa IV |
| Dokumentacja programu |  | umieszczać komentarze w kodzie źródłowym programu,  stosować w kodzie źródłowym komentarze typu DocBlocks,  tworzyć dokumentację techniczną aplikacji,  tworzyć pliki pomocy | tworzyć instrukcję użytkownika programu,  opracować dokumentację wdrożenia projektu,  opracować dokumentację z przeprowadzonych testów aplikacji | Klasa IV |
| Testowanie aplikacji |  | dokonać podziału testów (np. względu na weryfikowane obiekty,  ze względu na metodę weryfikacji, bazujące na wymaganiach),  zaplanować fazy testowania,  przeprowadzać testy w kolejnych fazach projektu informatycznego, przeprowadzać testy funkcjonalne,  stosować narzędzia do automatyzacji procesu testowania,  korzystać z systemów raportowania  błędów, np. BugZilla, JIRA,  przeprowadzać testy interfejsu,  testować prototyp projektu interfejsu | przeprowadzać testy niefunkcjonalne: użyteczności, wydajnościowe, obciążeniowe, zgodności, bezpieczeństwa,  przygotować środowisko testowe,  tworzyć scenariusze testowania aplikacji, | Klasa IV |
| **VI.Tworze-nie desktopo-wych aplikacji okienko-wych** | Programowanie desktopowych aplikacji okienkowych |  | zastosować funkcje jednego z języków C++, C#, Java Python do tworzenia aplikacji desktopowych,  stworzyć projekt i strukturę aplikacji,  zaprojektować interfejs użytkownika,  wstawić podstawowe widżety (kontrolki), np.: pole edycji, suwak, przycisk do okna aplikacji,  rozmieszczać widżety w oknie aplikacji,  przypisać widżetom zasady funkcjonowania,  zaprojektować obsługę zdarzeń myszy i klawiatury,  zaprogramować obsługę zdarzeń myszy i klawiatury,  wstawić do aplikacji kod źródłowy,  stosować predefiniowane okna dialogowe (np. wybór pliku, wybór czcionki, wybór koloru),  projektować i dodawać menu do aplikacji,  implementować opcje menu np. Plik, Edycja,  utworzyć interfejs użytkownika,  aktywnie słuchać ,włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,  stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;  zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;  wykazywać się kreatywnością w rozwiązywaniu problemów,  ponosić odpowiedzialność za podejmowane działania,  dokonać analizy i oceny podejmowanych działań,  doskonalić jakość wykonywanych działań;  doskonalić umiejętności zawodowe,  pracować w zespole. | Stosować język do projektowania interfejsu użytkownika, np. QML, w tym np.: stosować podstawowe elementy wizualne i niewizualne,  wykonać transformację obiektu (np. przesunięcie, obrót, skalowanie),  pozycjonować elementy,  animować właściwości elementów, np.: kolor, obrót, przejście,  projektować własne okna dialogowe,  tworzyć okienkowe aplikacje desktopowe, np. notatnik,  korzystać z elementów multimedialnych w aplikacji,  przechowywać dane aplikacji,  dynamiczne ładować elementy | Klasa IV |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Do osiągnięcia założonych celów kształcenia w zakresie przedmiotu Tworzenie aplikacji desktopowych konieczne jest:

* zaplanowanie lekcji poprzez wskazanie celów szczegółowych,
* wykorzystanie różnorodnych metod nauczania, w szczególności aktywizujących,
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez testy praktyczne i projekty,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

Środki dydaktyczne

W pracowni, w której prowadzone będą zajęcia powinno znajdować się stanowisko komputerowe dla nauczyciela, podłączone do sieci lokalnej z dostępem do internetu z urządzeniem wielofunkcyjnym oraz z projektorem multimedialnym lub tablicą interaktywną, ekran, głośniki, stanowiska komputerowe (jedno stanowisko dla jednego ucznia) z odpowiednim oprogramowaniem (oprogramowanie biurowe, różne środowiska programistyczne – edytor, kompilator, interpreter, biblioteki, frameworki), podłączenie do sieci lokalnej z dostępem do Internetu; dostęp do portalu wspierającego pracę grupową.

Zalecane metody dydaktyczne

Dla przedmiotu Tworzenie aplikacji desktopowych, który jest przedmiotem praktycznym, dominujące powinny być metody praktyczne np.: pokaz z objaśnieniem, ćwiczenia praktyczne, metoda projektów, metoda przewodniego tekstu. W kształceniu zawodowym dobrze sprawdzają się też metody problemowe, szczególnie aktywizujące, np.: metoda przypadków, dyskusja dydaktyczna.

Formy organizacyjne

Zajęcia z przedmiotu Tworzenie aplikacji desktopowych powinny być prowadzone indywidualnie lub zespołowo. Należą one do grupy zajęć praktycznych. O liczebności grup ogólnie mówi §6 Rozporządzenia Ministra Edukacji Narodowej z dnia 22 lutego 2019 r. w sprawie praktycznej nauki zawodu. W celu spełnienia wymienionych w nim warunków realizacji kształcenia praktycznego, zajęcia edukacyjne powinny się odbywać w pracowni z podziałem na grupy o liczebności maksymalnie do 12 osób.

Formy indywidualizacji pracy uczniów

W celu dostosowania warunków, środków, metod i form kształcenia do potrzeb i możliwości ucznia w zakresie organizacji pracy można zastosować instrukcje do zadań, podawanie dodatkowych zaleceń, objaśnień, instrukcji do pracy indywidualnej, udzielanie konsultacji indywidualnych. W pracy grupowej należy zwracać uwagę na taki podział zadań między członków zespołu, by każdy wykonywał tę część zadania, której podoła. Uczniom szczególnie zdolnym i posiadającym określone zainteresowania zawodowe należy zaplanować zadania o większym stopniu złożoności, proponować samodzielne poszerzanie wiedzy, studiowanie dodatkowej literatury.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Do głównych metod sprawdzania osiągnięć edukacyjnych ucznia należą ćwiczenia praktyczne oraz metoda projektów. Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu na podstawie kryteriów, przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* realizowanych zadań praktycznych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania zadań praktycznych,
* efektu końcowego projektu i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie z sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (tutoriali, dokumentacji technicznej – w tym w języku obcym)

Wskazane jest też, by uczniowie dokonywali samooceny i oceny kolegów z zespołu według zaproponowanych przez nauczyciela kryteriów.

Stosowane przez nauczyciela ocenianie powinno korzystać z zasad występujących w ocenianiu kształtującym, ma być dla ucznia informacją zwrotną, która pomaga mu się uczyć. Nauczyciel wskazuje silne strony ucznia i doradza, jak je rozwijać; wyraźnie i konstruktywnie informuje o stronach słabych i o tym, jak można je eliminować; stwarza uczniom możliwość poprawienia własnej pracy. Ocenianie skupiające się na postępach i osiągnięciach, a nie na podkreślaniu niepowodzeń, zachęca uczniów do uczenia się. Porównywanie osiągnięć poszczególnych uczniów z osiągnięciami ich kolegów, tworzenie rankingów, nie motywuje, a często zniechęca do uczenia się. Nauczyciel korzysta też z oceniania sumującego, nie rezygnuje ze stopni.

**Przykładowe zadania**

1. Zdefiniuj klasę Ocena oraz dziedziczącą z niej klasę OcenaTestu wg poniższego schematu jak na rysunku.

‘-‘ oznacza składową prywatną

‘+’ oznacza składową publiczną

|  |
| --- |
| Ocena |
| - wynik: double |
| + Ocena()  + Ocena(double)  + getWynik()  + wystawOcene() |

|  |
| --- |
| OcenaTestu |
| - iloscPytan: int  - punktacja: double  - iloscDobrychOdp: int |
| + OcenaTestu()  + OcenaTestu(int, int)  + setPunktacja(int)  + getPunktacja()  + obliczWynik() |

W klasie Ocena:

* konstruktor domyślny – zeruje pole wynik,
* konstruktor parametryczny – ustawia wartość pola wynik równą wartości swego argumentu,
* getWynik() – zwraca wartość pola wynik,
* ocenę wystawiamy (funkcja wystawOcene) w zależności od wyniku punktowego wg następujących zasad:

100 – 6

>= 90 i <100 – 5

>=75 i <90 – 4

>=50 i <75 – 3

>=30 i <50 – 2

<30 – 1

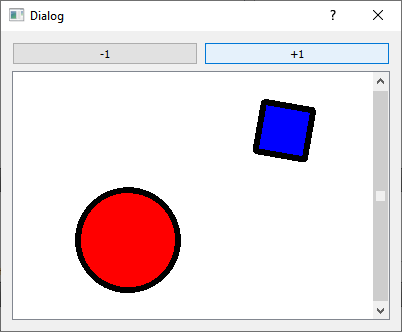
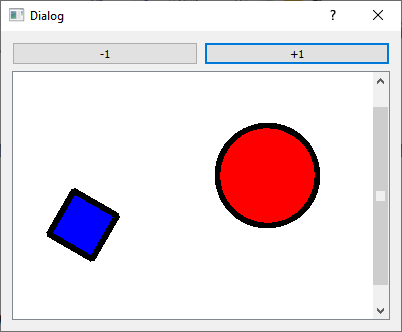
W klasie OcenaTestu odwołujemy się do konstruktorów klasy bazowej:

* konstruktor domyślny – zeruje wszystkie pola,
* konstruktor parametryczny – wypełnia pola wartościami argumentów konstruktora,
* punktację za każde pytanie w teście (setPunktacja(int)) ustalamy wg zasady, że zawsze maksymalny wynik punktowy testu to 100 pkt., tak więc np. przy 20 pytaniach, punktacja wynosi 5pkt.,
* funkcja obliczWynik() zwraca wartość równą iloczynowi: iloscDobrychOdp\*punktacja

W funkcji głównej main() przetestuj działanie metod klas

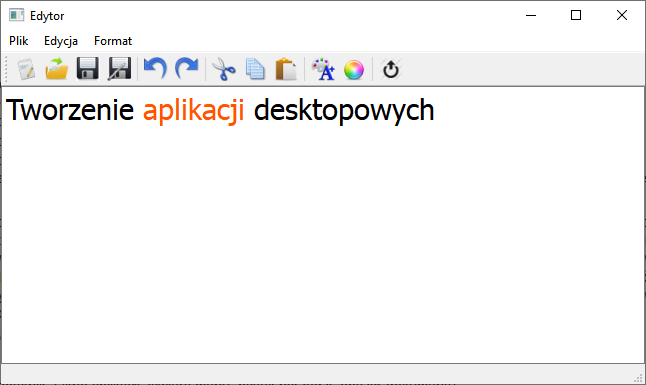
* wczytaj ilość pytań w teście i ilość udzielonych dobrych odpowiedzi,
* utwórz obiekt klasy OcenaTestu z parametrami równymi wczytanym wartościom,
* wyświetl: punktację, wynik punktowy testu i ocenę za test.

1. Napisz program prezentujący interakcję z użytkownikiem. Interfejs użytkownika pokazany jest na rysunku. Klikanie przycisków powoduje obrót figur geometrycznych w prawo lub lewo.

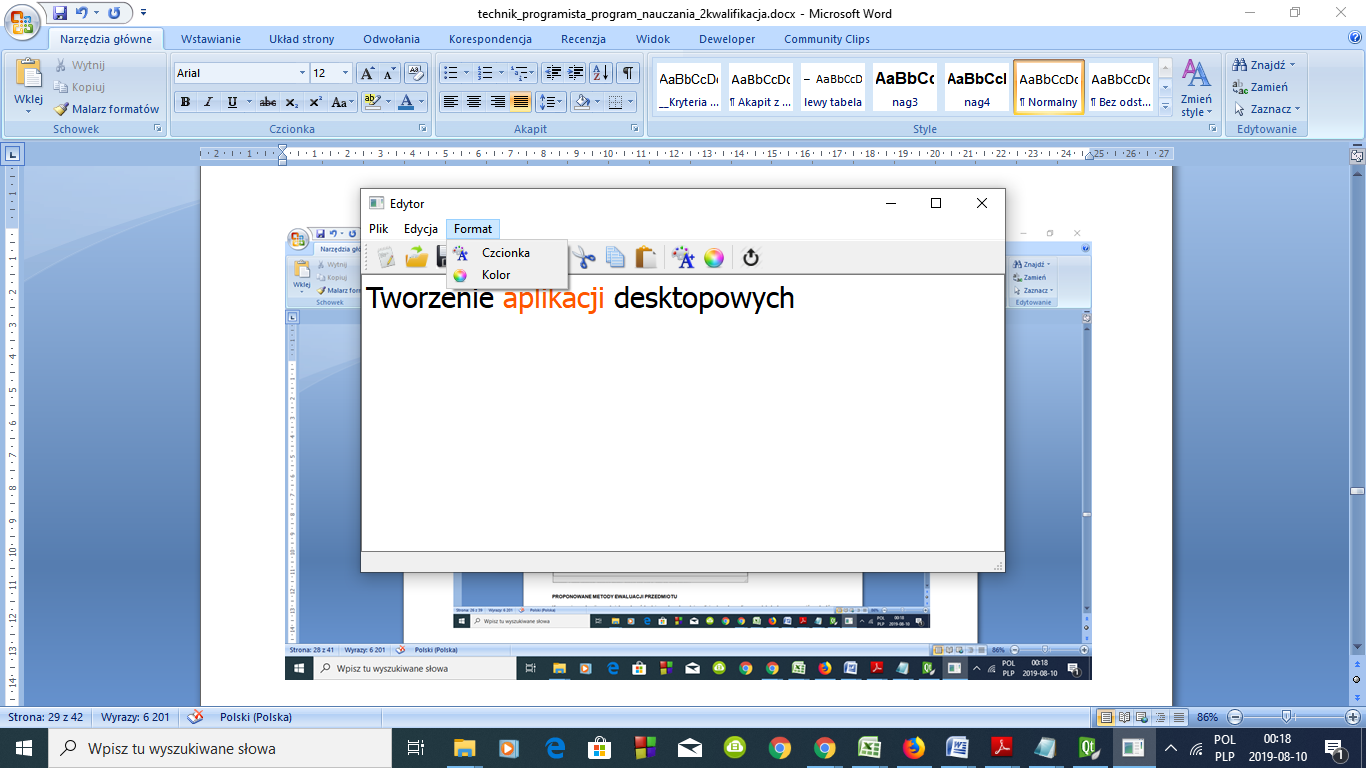
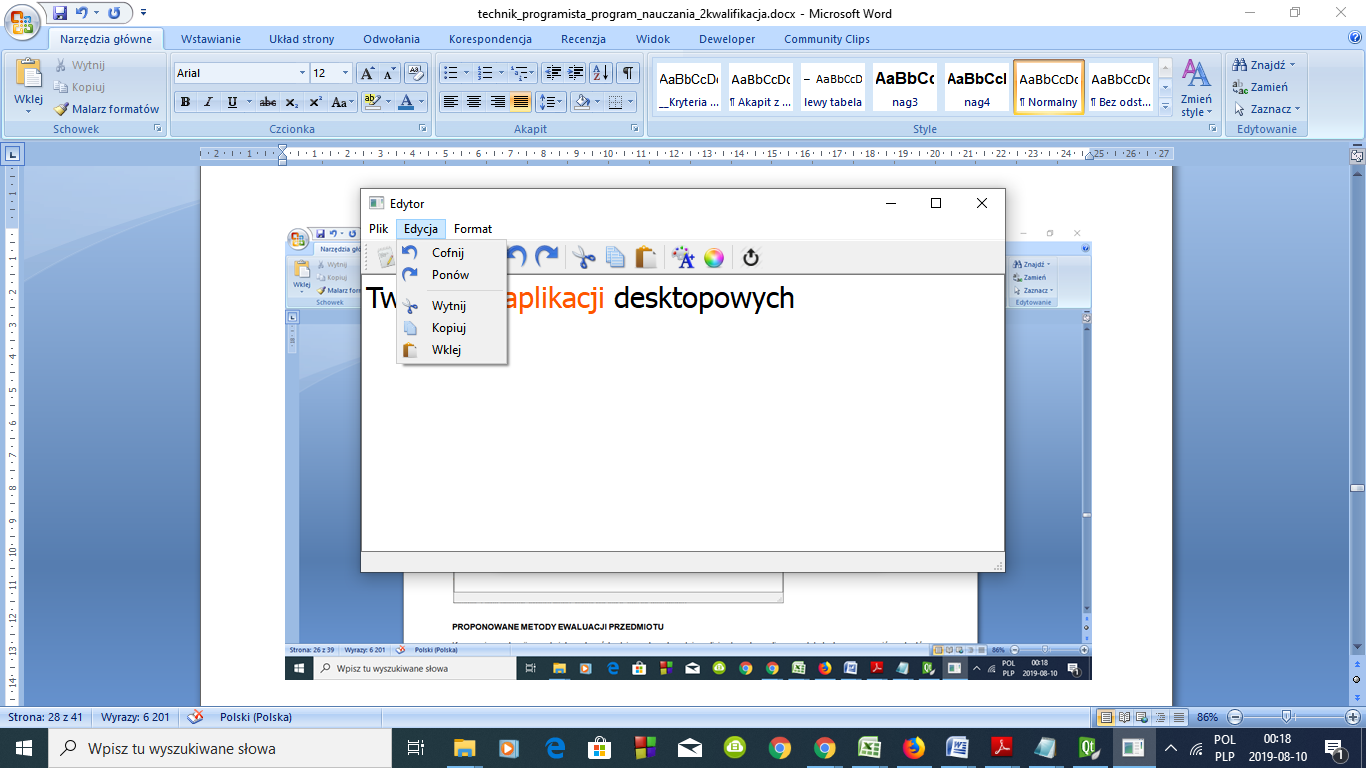
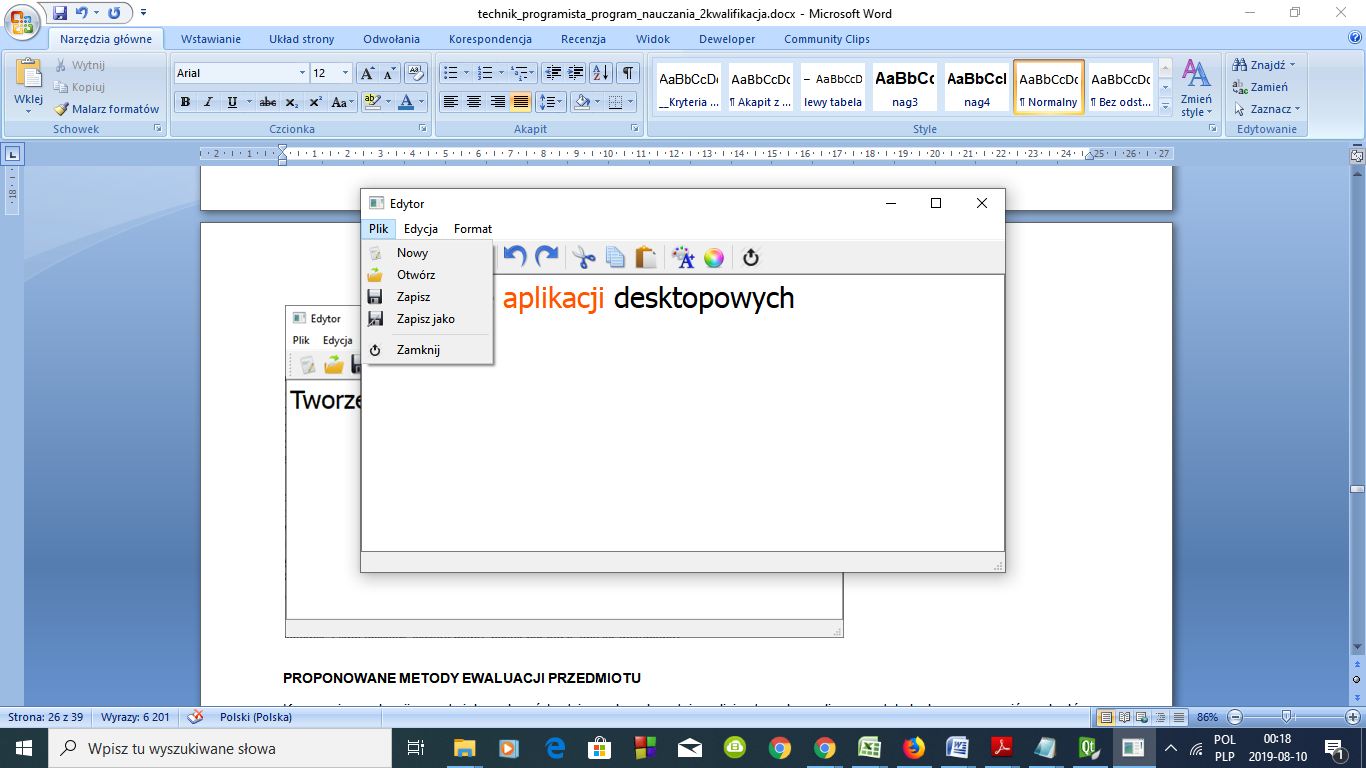


Rysunek.

1. Napisz program tworzący notatnik. Interfejs użytkownika pokazany jest na rysunku 1. Okno aplikacji zawiera menu, pasek narzędzi, obszar dokumentu. Funkcjonalność menu aplikacji pokazana jest na rysunku 2 i ma zostać powtórzona na pasku narzędzi.



Rysunek 1. Okno aplikacji



Rysunek 2. Polecenia poszczególnych opcji menu

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Koncepcja ewaluacji przedmiotu polegać będzie na tzw. twardej analizie danych, czyli ocen zdobytych przez uczniów z testów praktycznych z poszczególnych działów programowych oraz wykonanych przez uczniów projektów. Zebrane wyniki zostaną poddane analizie ilościowej i jakościowej z wykorzystaniem statystyki matematycznej.

Uzyskane informacje pozwolą na wyodrębnienie zagadnień sprawiających uczniom problemy. W następstwie, będzie można zwiększyć liczbę godzin dydaktycznych przypisanych do danego działu programowego. Korekta taka powinna się przyczynić do podwyższenia jakości kształcenia i indywidualnych wyników uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. samodzielnego tworzenia poprawnych programów desktopowych,
2. wyboru algorytmów do rozwiązania zadań,
3. tworzenia programów według dostarczonego projektu,
4. testowania i optymalizacji działania programu,
5. dokumentowania programu

**LITERATURA**

1. Bjarne Stroustrup, Język C++. Kompendium wiedzy, wyd. Helion,
2. Stephen Prata, Język C++. Szkoła programowania. Wydanie VI, wyd. Helion,
3. Grębosz Jerzy, Opus magnum C++11, Programowanie w języku C++ (komplet), Wyd.: Grębosz Jerzy,

#### Python dla każdego. Podstawy programowania – Michael Dawson,wyd. Helion,

#### Eric Matthes, Python. Instrukcje dla programisty, wyd. Helion,

#### Al Sweigart, Automatyzacja nudnych zadań z Pythonem, wyd. Helion,

1. http://www.cplusplus.com/
2. https://docs.python.org/3/tutorial/
3. http://codecademy.com
4. http://w3schools.com
5. https://doc.qt.io/qtcreator/index.html
6. http://www-cs.ccny.cuny.edu/~wolberg/cs221/qt/books/C++-GUI-Programming-with-Qt-4-1st-ed.pdf
7. https://qmlbook.github.io/assets/qt5\_cadaques.pdf

## Aplikacje desktopowe i mobilne

**Cele ogólne przedmiotu**

1. Poznanie

* oprogramowania do tworzenia aplikacji desktopowych,
* funkcji frameworków do tworzenia aplikacji okienkowych,
* zasad projektowania aplikacji okienkowych,
* narzędzi i metodologii planowania i zarządzania projektem,
* zasad projektowania aplikacji,
* prawnych aspektów praw autorskich,
* pojęć związanych ze środowiskiem programistycznym aplikacji mobilnych,
* pojęć związanych z interfejsem użytkownika aplikacji mobilnej,
* sposobów dokumentowania aplikacji,
* metod testowania aplikacji;
* podstawowych poleceń języka programowania aplikacji mobilnych;

1. Nabycie umiejętności
   * tworzenia aplikacji desktopowych,
   * projektowania aplikacji okienkowych,
   * tworzenia za pomocą frameworków desktopowych aplikacji okienkowych,
   * planowania i zarządzania projektem,
   * projektowania aplikacji,
   * przestrzegania praw autorskich
   * tworzenia interfejsu aplikacji w języku XAML;
   * tworzenia interfejsy aplikacji z elementów UI;
   * implementacji instrukcji warunkowych w języku programowania aplikacji mobilnych;
   * implementacji pętli w języku programowania aplikacji mobilnych;
   * definiowania klas i metod w języku programowania aplikacji mobilnych;
   * stosowania hermetyzacji w programowaniu obiektowym w języku programowania aplikacji mobilnych;
   * obsługi plików w języku programowania aplikacji mobilnych;
   * tworzenia dokumentacji dotyczącej projektu informatycznego;
2. Kształtowanie postawy, świadomości

* aktywnego słuchania i prowadzenia dyskusji;
* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w swoim środowisku;
* wrażliwości na potrzeby osób niepełnosprawnych;
* odpowiedzialności za podejmowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;
* doskonalenia umiejętności zawodowych,
* przestrzegania prawa

**Cele operacyjne**

**Uczeń potrafi:**

1. dobrać środowisko programistyczne do określonych zadań i języka programowania (np. Visual Studio) ,
2. opisać narzędzia wykorzystywane w procesie tworzenia aplikacji desktopowych,
3. objaśnić pojęcie frameworka,
4. opisać funkcje frameworka do tworzenia aplikacji desktopowych np. Qt w języku C++ (lub frameworka WPF – w języku C#)
5. opisać strategie organizowania okien w ramach pulpitu (SDI, MDI)
6. opisać rodzaje widżetów (kontrolek),
7. zaprojektować interfejs użytkownika,
8. stworzyć projekt i strukturę aplikacji,
9. zaprojektować okna aplikacji,
10. opisać zasady dodawania widżetów d okna aplikacji i sposoby ich rozmieszczania,
11. zaprojektować obsługę zdarzeń myszy i klawiatury,
12. zaprojektować menu aplikacji,
13. zaprojektować okna dialogowe aplikacji,
14. scharakteryzować język do projektowania interfejsu użytkownika, np. QML,
15. określić funkcje narzędzi do zarządzania projektem ,
16. dobrać narzędzia do zarządzania projektem,
17. zastosować narzędzia wizualizacji w zarządzaniu etapami projektu, zadaniami i czasem, np. diagram Gantta
18. korzystać z programów wspierających zarządzanie projektami, np. Jira, Trello,
19. korzystać z systemu kontroli wersji, np. Git,
20. analizować wymagania klienta i tworzyć aplikację zgodnie z nimi,
21. zidentyfikować elementy interfejsu użytkownika, np. formularze, paski narzędziowe, widżety,
22. zaprojektować interfejs użytkownika i wygląd aplikacji,
23. zaprojektować struktury danych dla aplikacji,
24. zaprojektować funkcjonalność aplikacji
25. utworzyć specyfikację techniczną dla zespołu programistów na podstawie wymagań klienta,
26. dostosować interfejs aplikacji do różnych platform,
27. zaprojektować aplikacje zgodnie z paradygmatami programowania: strukturalnym, obiektowym,
28. zaprojektować aplikację opartą na architekturze klient-serwer
29. zaplanować system zabezpieczeń aplikacji,
30. określić cel projektu ,
31. określić fazy realizacji projektu,
32. scharakteryzować cykl życia projektu informatycznego i jego poszczególne etapy,
33. określić zasoby ludzkie, ramy czasowe i koszt wykonania projektu,
34. planować etapy tworzenia aplikacji,
35. zastosować metodologię zarządzania projektem: model kaskadowy (waterfall), model przyrostowy, model prototypowy, metodyki zwinne (Agile oraz przynajmniej jedną z Scrum, Lean, Kanban) ,
36. dobrać optymalną metodologię zarządzania projektem,
37. organizować prace projektowe (wyznaczyć role, ustalić etapy prac),
38. stosować harmonogram czynności w celu efektywnego osiągnięcia celów,
39. dobrać wzorzec projektowy do zadania programistycznego,
40. stosować wzorce projektowe w programowaniu obiektowym, np. Metoda szablonowa (Template method), Fasada (Facade), Kompozyt (Composite)
41. rozróżniać autorskie prawa osobiste i majątkowe,
42. określić czas trwania praw autorskich,
43. określić konsekwencje naruszenia prawa autorskiego,
44. rozróżniać typy licencji oprogramowania,
45. scharakteryzować elementy własności intelektualnej (dobra niematerialne, własności przemysłowe),
46. rozróżnić dedykowane języki do programowania aplikacji mobilnych tj, Objective-C, Swift, Java, C#,
47. scharakteryzować typy danych w wybranym języku programowania,
48. deklarować zmienne w wybranym języku programowania,
49. deklarować tablice jedno i wielowymiarowe w wybranym języku programowania,
50. deklarować struktury danych w wybranym języku programowania,
51. wykonywać operacje arytmetyczne, bitowe, logiczne, przypisania, porównania na zmiennych w wybranym języku programowania,
52. obsługiwać operacje wejścia/wyjścia w wybranym języku programowania,
53. przetwarzać ciągi znaków i formatować dane tekstowe w wybranym języku programowania,
54. dobrać operatory warunkowe w wybranym języku programowania,
55. podać implementację instrukcji jeżeli w wybranym języku programowania,
56. podać implementację instrukcji przełączającej w wybranym języku programowania,
57. podać implementację poszczególnych pętli w wybranym języku programowania,
58. podać implementację instrukcji sterujących programem w wybranym języku programowania,
59. deklarować klasy i metody w wybranym języku programowania,
60. dobrać typy argumentów w metodach w wybranym języku programowania,
61. przeciążać metody w wybranym języku programowania,
62. definiować własne konstruktory i destruktory w klasie w wybranym języku programowania,
63. uzasadnić potrzebę stosowania hermetyzacji w programowaniu obiektowym,
64. uzasadnić potrzebę zastosowania dziedziczenia w wybranym języku programowania,
65. implementować klasy potomne w wybranym języku programowania,
66. implementować obsługę plików w wybranym języku programowania,
67. obsługiwać wyjątki i błędy w wybranym języku programowania,
68. dobrać środowiska programistyczne aplikacji mobilnych,
69. scharakteryzować narzędzia programistycznego środowiska aplikacji mobilnych,
70. określić parametry konfiguracji środowiska programistycznego aplikacji mobilnych,
71. scharakteryzować interakcje pomiędzy użytkownikiem i aplikacją mobilną,
72. rozróżnić gesty użytkownika dostępne w wybranym systemie iOS lub Android,
73. zidentyfikować elementy UI takie jak: przyciski, nawigacja, okna dialogowe, listy, formularze, paski narzędziowe, grafika, animacje, dźwięk w aplikacji mobilnej w systemie iOS,
74. identyfikować elementy UI takie jak: przyciski, nawigacja, okna dialogowe, listy, formularze, paski narzędziowe, grafika, animacje, dźwięk w aplikacji w systemie Android,
75. zastosować składnię języka XAML,
76. rozróżniać poszczególne elementy interfejsu użytkownika w kodzie XAML,
77. dobrać atrybuty elementów interfejsu użytkownika w języku XAML,
78. programować interfejs użytkownika w języku XAML,
79. zidentyfikować komentarze jedno i wielowierszowe,
80. stosować zastosowanie komentarzy w kodzie programu,
81. pisać dokumentację kodu,
82. pisać dokumentację pomocy do programu,
83. pisać instrukcję pomocy dla użytkownika,
84. pisać dokumentację wdrożenia projektu programistycznego,
85. rozróżniać narzędzia umożliwiające testowanie kodu programu w poszczególnych środowiskach programistycznych,
86. rozróżniać systemy raportowania błędów,
87. identyfikować błędy w kodzie programu zgłaszane przez interpreter,
88. scharakteryzować metody testowania programu,
89. przygotować testy funkcjonalne i niefunkcjonalne programu,
90. planować i organizować pracę zespołu w celu wykonania przydzielonych zadań,
91. dobrać osoby do wykonania przydzielonych zadań,
92. kierować wykonaniem przydzielonych zadań,
93. oceniać jakość wykonania przydzielonych zadań, aktywnie słuchać ,włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,
94. stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;
95. zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;
96. wykazywać się kreatywnością w rozwiązywaniu problemów,
97. ponosić odpowiedzialność za podejmowane działania,
98. dokonać analizy i oceny podejmowanych działań,
99. doskonalić jakość wykonywanych działań;

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** | |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji | |
| **I. Progra-mowanie desktopo-wych aplikacji okienko-wych** | Środowisko programistyczne dla aplikacji desktopowych |  | dobrać środowisko programistyczne do określonych zadań i języka programowania (np. Visual Studio) ,  opisać narzędzia wykorzystywane w procesie tworzenia aplikacji desktopowych |  | Klasa III | |
| Frameworki do programowania aplikacji desktopowych |  | objaśnić pojęcie frameworka,  opisać funkcje frameworka do tworzenia aplikacji desktopowych np. Qt w języku C++ (lub frameworka WPF – w języku C#) |  | Klasa III | |
| Programowanie desktopowych aplikacji okienkowych |  | opisać strategie organizowania okien w ramach pulpitu (SDI, MDI)  opisać rodzaje widżetów (kontrolek),  zaprojektować interfejs użytkownika,  stworzyć projekt i strukturę aplikacji,  zaprojektować okna aplikacji,  opisać zasady dodawania widżetów do okna aplikacji i sposoby ich rozmieszczania,  zaprojektować obsługę zdarzeń myszy i klawiatury,  projektować menu aplikacji,  zaprojektować okna dialogowe aplikacji,  aktywnie słuchać ,włączając się do dyskusji podczas szukania sposobu rozwiązania problemu,  stosować zasady kultury osobistej i ogólnie przyjęte normy zachowania w swoim środowisku;  zastosować właściwą technikę twórczego myślenia przy rozwiązaniu problemu;  wykazywać się kreatywnością w rozwiązywaniu problemów,  ponosić odpowiedzialność za podejmowane działania,  dokonać analizy i oceny podejmowanych działań,  doskonalić jakość wykonywanych działań; | scharakteryzować język do projektowania interfejsu użytkownika, np. QML, | Klasa III | |
| **II. Planowanie i zarządzanie projektem** | Narzędzia i metodologie planowania i zarządzania projektem |  | określić funkcje narzędzi do zarządzania projektem ,  dobrać narzędzia do zarządzania projektem,  zastosować narzędzia wizualizacji w zarządzaniu etapami projektu, zadaniami i czasem, np. diagram Gantta | korzystać z programów wspierających zarządzanie projektami, np. Jira, Trello,  korzystać z systemu kontroli wersji, np. Git | Klasa III | |
| Projektowanie aplikacji |  | analizować wymagania klienta i tworzyć aplikację zgodnie z nimi,  identyfikować elementy interfejsu użytkownika, np. formularze, paski narzędziowe, widżety,  projektować interfejs użytkownika i wygląd aplikacji,  projektować struktury danych dla aplikacji,  projektować funkcjonalność aplikacji | tworzyć specyfikację techniczną dla zespołu programistów na podstawie wymagań klienta,  dostosować interfejs aplikacji do różnych platform,  projektować aplikacje zgodnie z paradygmatami programowania: strukturalnym, obiektowym,  projektować aplikację opartą na architekturze klient-serwer  planować system zabezpieczeń aplikacji | Klasa III | |
| Planowanie przedsięwzięć programistycznych |  | określić cel projektu ,  określić fazy realizacji projektu,  charakteryzować cykl życia projektu informatycznego i jego poszczególne etapy,  określić zasoby ludzkie, ramy czasowe i koszt wykonania projektu,  planować etapy tworzenia aplikacji;  planować i organizować pracę zespołu w celu wykonania przydzielonych zadań,  dobierać osoby do wykonania przydzielonych zadań,  kierować wykonaniem przydzielonych zadań,  oceniać jakość wykonania przydzielonych zadań | stosować metodologię zarządzania projektem: model kaskadowy (waterfall), model przyrostowy, model prototypowy, metodyki zwinne (Agile oraz przynajmniej jedną z Scrum, Lean, Kanban) ,  dobierać optymalną metodologię zarządzania projektem,  organizować prace projektowe (wyznaczyć role, ustalić etapy prac),  stosować harmonogram czynności w celu efektywnego osiągnięcia celów | Klasa III | |
| **III. Wzorce projektowe** | Wzorce projektowe |  |  | dobrać wzorzec projektowy do zadania programistycznego,  stosować wzorce projektowe w programowaniu obiektowym, np. Metoda szablonowa (Template method), Fasada (Facade), Kompozyt (Composite) | Klasa III | |
| **IV. Aspekty prawne** | Prawo autorskie w dziedzinie programowania |  | rozróżniać autorskie prawa osobiste i majątkowe,  określać czas trwania praw autorskich,  określać konsekwencje naruszenia prawa autorskiego,  rozróżniać typy licencji oprogramowania | charakteryzować elementy własności intelektualnej (dobra niematerialne, własności przemysłowe) | Klasa III | |
| **V. Język programowania aplikacji mobilnych** | 1. Elementy języka programowania. |  | - rozróżnić dedykowane języki do programowania aplikacji mobilnych tj, Objective-C, Swift, Java, C#,  - charakteryzować typy danych w wybranym języku programowania,  - deklarować zmienne w wybranym języku programowania,  - wykonywać operacje arytmetyczne, bitowe, logiczne, przypisania, porównania na zmiennych w wybranym języku programowania,  - obsługiwać operacje wejścia/wyjścia w wybranym języku programowania, | - deklarować tablice jedno i wielowymiarowe w wybranym języku programowania,  - deklarować struktury danych w wybranym języku programowania,  - przetwarzać ciągi znaków i formatować dane tekstowe w wybranym języku programowania, | | Klasa III |
| 2.Pętle i instrukcje warunkowe. |  | - dobrać operatory warunkowe w wybranym języku programowania,  - podać implementację instrukcji jeżeli w wybranym języku programowania,  - podać implementację instrukcji sterujących programem w wybranym języku programowania, | - podać implementację instrukcji *jeżeli … w przeciwnym przypadku* w wybranym języku programowania,  - podać implementację instrukcji przełączającej w wybranym języku programowania,  - podać implementację poszczególnych pętli w wybranym języku programowania, | | Klasa III |
| 3. Programowanie obiektowe. |  | - deklarować klasy i metody w wybranym języku programowania,  - dobierać typy argumentów w metodach w wybranym języku programowania,,  - uzasadnić potrzebę zastosowania dziedziczenia w wybranym języku programowania, | - przeciążać metody w wybranym języku programowania,  - definiować własne konstruktory i destruktory w klasie w wybranym języku programowania,  - implementować klasy potomne w wybranym języku programowania, | | Klasa III |
| Obsługa plików oraz wyjątków. |  |  | - implementować obsługę plików w wybranym języku programowania,  - obsługiwać wyjątki i błędy w wybranym języku programowania, | | Klasa III |
| **VI. Specyfika programo-wania aplikacji mobilnych** | 1. Środowiska programistyczne aplikacji mobilnych |  | - dobierać środowiska programistyczne aplikacji mobilnych,  - charakteryzować narzędzia programistycznego środowiska aplikacji mobilnych,  - określać parametry konfiguracji środowiska programistycznego aplikacji mobilnych, |  | | Klasa III |
| 2. Interfejs użytkownika aplikacji mobilnej. |  | - charakteryzować interakcje pomiędzy użytkownikiem i aplikacją mobilną,  - rozróżniać gesty użytkownika dostępne w wybranym systemie iOS lub Android,  - identyfikować elementy UI takie jak: przyciski, nawigacja, okna dialogowe, listy, formularze, paski narzędziowe, grafika, animacje, dźwięk w aplikacji mobilnej w systemie iOS,  - identyfikować elementy UI takie jak: przyciski, nawigacja, okna dialogowe, listy, formularze, paski narzędziowe, grafika, animacje, dźwięk w aplikacji w systemie Android, | stosować składnię języka XAML,  rozróżniać poszczególne elementy interfejsu użytkownika w kodzie XAML,  dobierać atrybuty elementów interfejsu użytkownika w języku XAML,  programować interfejs użytkownika w języku XAML, | | Klasa III |
| **III. Testowanie i dokumentowanie aplikacji** | Testowanie aplikacji |  | - rozróżniać narzędzia umożliwiające testowanie kodu programu w poszczególnych środowiskach programistycznych,  - rozróżniać systemy raportowania błędów,  - charakteryzować metody testowania programu, | - identyfikować błędy w kodzie programu zgłaszane przez interpreter,  - przygotować testy funkcjonalne i niefunkcjonalne programu, | | Klasa III |
| Dokumentowanie aplikacji |  | - zidentyfikować komentarze jedno i wielowierszowe,  - stosować komentarze w kodzie programu,  - pisać instrukcję pomocy dla użytkownika, | - pisać dokumentację kodu,  - pisać dokumentację pomocy do programu,  - pisać dokumentację wdrożenia projektu programistycznego, | | Klasa III |
| Razem | |  |  |  | |  |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Do osiągania założonych celów kształcenia w zakresie przedmiotu Aplikacje desktopowe i mobilne konieczne jest

* zaplanowanie lekcji poprzez wskazanie celów szczegółowych,
* wykorzystanie różnorodnych metod nauczania, w szczególności aktywizujących,
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany, kartkówki, testy i projekty,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

Środki dydaktyczne

W sali lekcyjnej lub pracowni, w której prowadzone będą zajęcia powinno znajdować się stanowisko komputerowe dla nauczyciela, z odpowiednim oprogramowaniem (oprogramowanie biurowe, różne środowiska programistyczne – edytor, kompilator, biblioteki, frameworki , oprogramowanie wspomagające zarządzanie projektem, dostęp do portalu wspierającego pracę grupową), podłączone do sieci lokalnej z dostępem do internetu z urządzeniem wielofunkcyjnym oraz z projektorem multimedialnym lub tablicą interaktywną, ekran, głośniki, W pracowni, w której prowadzone będą zajęcia powinny się znajdować: plansze z elementami interfejsu UI, plansze dotyczące języka XAML, plansze z poleceniami w wybranym języku programowania, plansze z typowymi konstrukcjami programistycznymi w wybranym języku programowania, literatura dotycząca tworzenia interfejsu aplikacji mobilnej, języków programowania oraz środowisk programistycznych aplikacji mobilnych, komputer wyposażony w oprogramowanie służące do programowania aplikacji mobilnych, dostęp do Internetu, projektor, tablica interaktywna, drukarka.

Zalecane metody dydaktyczne

Dla przedmiotu Aplikacje desktopowe i mobilne, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych, opartych na słowie i obserwacji, takich jak:

* wykład informacyjny,
* pogadanka,
* pokaz z objaśnieniem,
* wykład problemowy,
* dyskusja dydaktyczna,
* burza mózgów,
* ćwiczenia praktyczne,

Zajęcia mogą odbywać się w grupach. Istotną metodą kształcenia powinny być także ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku.

Formy organizacyjne

Zajęcia z przedmiotu Aplikacje desktopowe i mobilne powinny być prowadzone indywidualnie, zespołowo lub zbiorowo. Należą one do grupy zajęć teoretycznych. W przypadku realizacji ćwiczeń praktycznych powinny być stosowane formy organizacyjne indywidualne. W kształceniu zawodowym bardzo ważną kwestią jest dostosowanie wymagań edukacyjnych do zróżnicowanych potrzeb i możliwości uczniów. Ponadto uczniowie powinni samodzielnie zdobywać wiedzę i kształtować umiejętności poprzez uczenie się we współpracy oraz korzystanie z różnych źródeł informacji.

Formy indywidualizacji pracy uczniów

W celu dostosowania warunków, środków, metod i form kształcenia do potrzeb i możliwości ucznia można w zakresie organizacji pracy zastosować instrukcje do zadań, podawanie dodatkowych zaleceń, objaśnień, instrukcji do pracy indywidualnej, udzielanie konsultacji indywidualnych. Nauczyciel powinien dostosować stopień trudności materiału do możliwości i potrzeb uczniów. W pracy grupowej należy zwracać uwagę na taki podział zadań między członków zespołu, by każdy wykonywał tę część zadania, której podoła. Uczniom szczególnie zdolnym i posiadającym określone zainteresowania zawodowe należy zaplanować zadania o większym stopniu trudności, proponować samodzielne poszerzanie wiedzy, korzystanie z dodatkowej literatury.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

W celu sprawdzenia osiągnięć edukacyjnych ucznia proponuje się: testy wielokrotnego wyboru, testy zawierające zadania otwarte, kartkówki, prezentacje uczniów, projekty.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu, na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* realizowanych ćwiczeń praktycznych w formie zadań,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń,
* produktu projektu i jego prezentacji,

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (tutoriali, dokumentacji danego języka – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest też, by uczniowie dokonywali samooceny i oceny kolegów z zespołu według zaproponowanych przez nauczyciela kryteriów.

Stosowane przez nauczyciela ocenianie powinno korzystać z zasad występujących w ocenianiu kształtującym, ma być dla ucznia informacją zwrotną, która pomaga mu się uczyć. Nauczyciel wskazuje silne strony ucznia i doradza, jak je rozwijać; wyraźnie i konstruktywnie informuje o stronach słabych i o tym, jak można je eliminować; stwarza uczniom możliwość poprawienia własnej pracy. Ocenianie skupiające się na postępach i osiągnięciach, a nie na podkreślaniu niepowodzeń, zachęca uczniów do uczenia się. Porównywanie osiągnięć poszczególnych uczniów z osiągnięciami ich kolegów, tworzenie rankingów, nie motywuje, a często zniechęca do uczenia się. Nauczyciel korzysta też z oceniania sumującego, nie rezygnuje ze stopni.

**Przykładowe zadania**

1. Opisz rodzaje licencji komputerowych.
2. Scharakteryzuj narzędzia i metodologię planowania i zarządzania projektem.
3. Zaimplementuj w języku programowania aplikacji mobilnych klasę pojazd opisującą ogólne właściwości i metody dla pojazdów. Na podstawie klasy pojazd utwórz klasę samochód dodając własności i metody dla samochodów. Na podstawie klasy pojazd utwórz klasę motor opisującą własności i metody dla motorów. Po utworzeniu powyższych trzech klas, przedyskutuj swoje spostrzeżenia z grupą. Wspólnie zastanówcie się czy na podstawie klasy samochód można zbudować kolejne podklasy.
4. Zaimplementuj interfejs aplikacji kalkulator w języku XAML.
5. Wymień i scharakteryzuj kolejne kroki testowania aplikacji mobilnej.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Koncepcja ewaluacji przedmiotu polegać będzie na tzw. twardej analizie danych, czyli ocen zdobytych przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych oraz z przygotowanych przez uczniów projektów. Zebrane wyniki zostaną poddane analizie ilościowej i jakościowej z wykorzystaniem statystyki matematycznej.

Uzyskane informacje pozwolą na wyodrębnienie zagadnień sprawiających uczniom problemy. W następstwie, będzie można zwiększyć liczbę godzin dydaktycznych przypisanych do danego działu programowego. Korekta taka powinna się przyczynić do podwyższenia jakości kształcenia i poprawy indywidualnych wyników uczniów z egzaminu w danej kwalifikacji.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

* planowania i zarządzania projektem,
* projektowania aplikacji,
* znajomości zasad programowania desktopowych aplikacji okienkowych.
* stosowania implementacji poleceń wybranego języka programowania aplikacji mobilnych,
* tworzenie interfejsu użytkownika z wykorzystaniem języka XAML,
* tworzenie interfejsu użytkownika na podstawie dostępnych elementów UI,
* dokumentowanie projektów programistycznych.

**ZALECANA LITERATURA**

* Dawn Griffiths, David Griffiths, Android. Programowanie aplikacji. Rusz głową! Wydanie II, wyd. Helion,
* Marcin Płonkowski, Android Studio. Tworzenie aplikacji mobilnych. wyd. Helion,
* Matt Neuburg, iOS 12. Wprowadzenie do programowania w Swifcie. Wydanie V, wyd. Helion,
* Steven F. Daniel, Xamarin. Tworzenie interfejsów użytkownika, wyd. Helion.

## Aplikacje webowe

**Cele ogólne przedmiotu**

1. Poznanie pojęć związanych ze środowiskiem do tworzenia aplikacji webowych np. Visual Studio, Eclipse, JetBrains;
2. Poznanie narzędzi charakterystycznych dla programowania aplikacji webowych;
3. Poznanie podstawowych informacji związanych z typowymi frameworkami dla aplikacji webowych np. ASP.NET Core, Django, Angular, React.js, Node.js
4. Nabycie wiedzy na temat zastosowania frameworku Node.js w programowaniu aplikacji webowych;
5. Nabycie wiedzy na temat zastosowania frameworku Diango w programowaniu aplikacji webowych;
6. Nabycie podstawowej wiedzy nt. składni i poleceń języka Python;
7. Nabycie wiadomości o implementacji języka Python do tworzenia aplikacji webowych;

**Cele operacyjne**

**Uczeń potrafi:**

1. dobrać środowisko programistyczne do projektu aplikacji webowej,
2. dobrać niezbędne narzędzia do pracy w środowisku programistycznym,
3. dobrać środowisko pracy z wybranymi frameworkami Node.js i Django,
4. opisać podstawowe cechy frameworku Node.js,
5. rozpocząć pracę z Node.js,
6. rozpoznać wbudowane moduły Node.js,
7. opisać kluczowe techniki stosowane w frameworku Node.js;
8. stosować pliki, strumienie i bufory w Node.js,
9. scharakteryzować moduł http w Node.js,
10. zainstalować menadżer pakietów npm dla Node.js,
11. zainstalować framework expres.js dla Node.js,
12. połączyć Node.js z bazą danych,
13. korzystać z przydatnych narzędzi np. WebSocket, moduł Assert, biblioteka Chai oraz Mocha przeznaczonych dla Node.js,
14. opisać podstawowe cech frameworku Django,
15. zainstalować Django na serwerze,
16. opisać podstawowe elementu projektu wykonanego w Django,
17. opisać kolejne etapy pracy nad projektem aplikacji tworzonej w Django,
18. opisać system komentarzy w Django,
19. opisać zasady tworzenia okna logowania i okna rejestracji użytkowników w Django,
20. scharakteryzować profile użytkowników w Django,
21. obsługiwać sesje i ciasteczka z poziomu Django,
22. dobrać środowisko programowania w języku Python,
23. opisać typy liczbowe w języku Python,
24. opisać typy sekwencyjne w języku Python,
25. opisać instrukcje warunkowe w języku Python,
26. opisać pętle while i for w języku Python,
27. opisać słowniki i listy składane w języku Python,
28. opisać sposób deklaracji funkcje w języku Python,
29. stosować programowanie funkcyjne,
30. wskazać metody formatowania łańcuchów w języku Python,
31. opisać metody jak pobierać argumenty ze standardowego wejścia w języku Python,,
32. opisać obsługę wyjątków w języku Python,
33. podać techniki pracy z plikami w języku Python,
34. korzystać z daty i czasu w języku Python,
35. współpracować z systemem operacyjnym z poziomu języka Python,
36. deklarować klasy i instancje, atrybuty i metody w języku Python,
37. deklarować konstruktor klasy,
38. stosować technikę dziedziczenia w języku Python,
39. przeciążać operatory w języku Python,
40. obsługiwać interfejs graficzny w języku Python,
41. obsługiwać bazy danych w języku Python,
42. realizować współpracę języka Python z serwerem Apache.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Zaawansowanie techniki programowania aplikacji webowych** | 1. Środowisko pracy. |  | - dobrać środowisko programistyczne do projektu aplikacji webowej,  - dobrać niezbędne narzędzia do pracy w środowisku programistycznym,  - dobrać środowisko pracy z wybranymi frameworkami Node.js i Django, |  | Klasa III |
| 2. Framework Node.js |  | - rozpocząć pracę z Node.js,  - rozpoznać wbudowane moduły Node.js,  - opisać kluczowe techniki stosowane w frameworku Node.js;  - stosować pliki, strumienie i bufory w Node.js,  - zainstalować menadżer pakietów npm dla Node.js,  - zainstalować framework expres.js dla Node.js, | - scharakteryzować moduł http w Node.js,  - połączyć Node.js z bazą danych,  - korzystać z przydatnych narzędzi np. WebSocket, moduł Assert, biblioteka Chai oraz Mocha przeznaczonych dla Node.js, | Klasa III |
| 3. Framework Django |  | - opisać podstawowe cech frameworku Django,  - opisać podstawowe elementu projektu wykonanego w Django,  - opisać kolejne etapy pracy nad projektem aplikacji tworzonej w Django,  - opisać system komentarzy w Django,  - scharakteryzować profile użytkowników w Django, | - zainstalować Django na serwerze,  - opisać zasady tworzenia okna logowania i okna rejestracji użytkowników w Django,  - obsługiwać sesje i ciasteczka z poziomu Django, | Klasa III |
| **II. Specyfika programowania aplikacji mobilnych.** | 1. Podstawy języka Python |  | - dobrać środowisko programowania w języku Python,  - opisać typy liczbowe w języku Python,  - opisać typy sekwencyjne w języku Python,  - opisać instrukcje warunkowe w języku Python,  - opisać pętle while i for w języku Python,  - opisać słowniki i listy składane w języku Python,  - opisać metody jak pobierać argumenty ze standardowego wejścia w języku Python, | - opisać obsługę wyjątków w języku Python,  - podać techniki pracy z plikami w języku Python,  - korzystać z daty i czasu w języku Python,  - współpracować z systemem operacyjnym z poziomu języka Python,  - opisać sposób deklaracji funkcje w języku Python,  - stosować programowanie funkcyjne,  - wskazać metody formatowania łańcuchów w języku Python, | Klasa III |
| 2. Programowanie obiektowe w języku Python. |  | - deklarować klasy i instancje, atrybuty i metody w języku Python,  - deklarować konstruktor klasy, | - stosować technikę dziedziczenia w języku Python,  - przeciążać operatory w języku Python, | Klasa III |
| 3.Elementy języka Python przydatne w aplikacjach webowych. |  |  | - obsługiwać interfejs graficzny w języku Python,  - obsługiwać bazy danych w języku Python,  - realizować współpracę języka Python z serwerem Apache. | Klasa III |
| Razem | |  |  |  |  |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągania założonych celów kształcenia w zakresie przedmiotu jest opracowanie odpowiednich dla danego zawodu procedur, a w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

**METODY NAUCZANIA**

Dla przedmiotu Aplikacje desktopowe i mobilne, który należy do przedmiotów teoretycznych, zaleca się stosowanie metod nauczania podających, eksponujących i problemowych takich jak:

* wykład informacyjny,
* pokaz z objaśnieniem,
* wykład problemowy,
* dyskusja dydaktyczna,
* burza mózgów,
* ćwiczenia praktyczne,

Zajęcia mogą także odbywać się w grupach. Dominująca metodą kształcenia powinny być także ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku, polegająca na analizowaniu przypadku opisującego problem. W trakcie realizacji zajęć nauczyciel powinien:

– motywować uczniów do pracy,

– dostosowywać stopień trudności planowanych ćwiczeń do możliwości uczniów,

– uwzględniać zainteresowania uczniów,

– przygotowywać zadania o różnym stopniu trudności i złożoności,

– zachęcać uczniów do korzystania z różnych źródeł informacji zawodowej,

– stosować metody aktywizujące,

– stosować nowoczesne środki kształcenia, np. tablice multimedialne.

**ŚRODKI DYDAKTYCZNE**

W pracowni, w której prowadzone będą zajęcia powinny się znajdować: plansze elementami/modułami/funkcjami charakterystycznymi dla wybranych frameworków Django, Node.js, plansze z poleceniami w wybranym języku programowania np. Python, plansze z typowymi konstrukcjami programistycznymi w wybranym języku programowania, literatura dotycząca tworzenia aplikacji webowej, języków programowania oraz środowisk programistycznych aplikacji webowych, komputer wyposażony w oprogramowanie służące do programowania aplikacji webowych, dostęp do Internetu, projektor, tablica interaktywna, drukarka.

**FORMY ORGANIZACYJNE**

Zajęcia powinny być prowadzone z wykorzystaniem różnych form organizacyjnych: indywidualnie i zespołowo. W przypadku realizacji ćwiczeń praktycznych powinny być stosowane formy organizacyjne indywidualne. Bardzo ważną kwestią w kształceniu zawodowym jest indywidualizacja pracy w kierunku potrzeb i możliwości ucznia w zakresie metod, środków oraz form kształcenia. Ponadto uczniowie powinni samodzielnie budować swoją wiedzę i kształtować umiejętności poprzez uczenie się we współpracy oraz korzystanie z różnych źródeł informacji.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Metody sprawdzania osiągnięć edukacyjnych ucznia: testy wielokrotnego wyboru, testy zawierające zadania otwarte, odpowiedzi ustne, prezentacje uczniów.

Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu, na podstawie kryteriów przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* ustnych sprawdzianów, sprawdzających poziom wiedzy i umiejętności,
* pisemnych sprawdzianów i testów osiągnięć szkolnych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania ćwiczeń praktycznych,
* projektu aplikacji i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest, aby uczniowie dokonywali także własnej samooceny pracy i kolegów z zespołu, wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

**Przykładowe zadania**

Zadanie 1

Wymień i opisz kolejne etapy pracy nad projektem aplikacji webowej realizowanej z wykorzystaniem frameworku Django.

Zadanie 2

Jakie wyróżnisz sposoby połączenie aplikacji webowej napisanej z wykorzystaniem frameworku Node.js z bazą danych? Który z wymienionych sposobów polecisz do napisania aplikacji typu ogłoszenia internetowe, która gromadzi informacje o użytkownikach, ich ogłoszeniach, posiada panel administracyjny oraz dostęp dla użytkowników niezarejestrowanych typu gość?

Zadanie 3

Jakie czynności należy wykonać po stronie serwera Apache współpracował z nim język Python? Do udzielenia odpowiedzi przyjmij, że na serwerze jest zainstalowana tylko usługa Apache.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Strategia przeprowadzanej ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych. Zebrane dane zostaną poddane analizie ilościowej i jakościowej przy użyciu narzędzia, którym jest statystyka matematyczna. Przydatnym narzędziem w tej analizie może być na przykład korzystanie z platformy testowej office 365.com lub podobnej, która daje możliwość analizy, które z pytań testowych sprawiają trudność.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego. Spowoduje to podwyższenie jakości kształcenia i znacząco wpłynie na indywidualne wyniki uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. znajomość składni kodu dla frameworku Node.js,

2. znajomość składni kodu dla frameworku Django,

3. znajomość modułów i rozszerzeń używanych we frameworkach,

4. znajomość składni języka Python stosowanego w frameworku Django.

**ZALECANA LITERATURA**

* Gniewomir Sarbicki, Python. Kurs dla nauczycieli i studentów, wyd. Helion,
* David Herron, Platforma Node.js. Przewodnik webdevelopera. Wydanie III, wyd. Helion,
* Antonio Mele, Django 2. Praktyczne tworzenie aplikacji sieciowych. Wydanie II, wyd. Helion.

## Tworzenie aplikacji webowych

**Cele ogólne przedmiotu**

1. Poznanie frameworków Node.js iDjango;
2. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem wzorców projektowych;
3. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem frameworku jQuery;
4. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem frameworku Node.js;
5. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem frameworku Django;
6. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem języka programowania np. PHP, C#, Python, JavaScript;
7. Nabycie umiejętności wdrażania aplikacji webowej;

**Cele operacyjne**

**Uczeń potrafi:**

1. stosować środowisko programistyczne do realizacji projektu aplikacji webowej,
2. korzystać z dedykowanych narzędzi dostępnych w środowisku programistycznym przy tworzeniu aplikacji webowej w danym frameworku,
3. korzystać z dedykowanych narzędzi dostępnych w środowisku programistycznym przy tworzeniu aplikacji webowej w danym języku prrogramowania,
4. implementować podstawowe operacje w jQuery,
5. implementować wybór elementów w tym selektorów poprzez jQuery,
6. zmodyfikować wygląd aplikacji webowej poprzez kodu w tym HTML poprzez jQuery,
7. odbierać zdarzenia w jQuery,
8. odpowiadać na zdarzenia w jQuery,
9. generować żądania GET w w jQuery,
10. przekazywać nagłówki HTTP w jQuery,
11. wczytywać kod XML w jQuery,
12. obsługiwać zdarzenia AJAX w jQuery,
13. odczytać dane JSON z zewnętrznego serwera w jQuery,
14. implementować przeciąganie elementów w jQuery UI,
15. implementować upuszczanie elementów w jQuery UI,
16. implementować zmianę kolejności elementów przy wykorzystaniu elementów sortowalnych w jQuery UI,
17. implementować zaznaczanie elementów w jQuery UI,
18. implementować grupowanie treści w jQuery,
19. implementować uzupełnianie treści w jQuery,
20. implementować zmianę elementu w przycisk w jQuery,
21. implementować okna dialogowe w jQuery,
22. implementować pobieranie liczb za pomocą suwaka w jQuery,
23. implementować nawigację w aplikacji przy użyciu kart w jQuery,
24. implementować pasek postępu w jQuery,
25. dostosować aplikację webową do urządzeń mobilnych w jQuery,
26. implementować nawigowanie aplikacje webowej przy użyciu jQuery Mobile,
27. implementować interakcję aplikacji webowej z użytkownikiem na urządzeniu mobilnym przy użyciu jQuery Mobile,
28. skorzystać z gotowych wtyczek w jQuery,
29. utworzyć własne wtyczki w jQuery,
30. utworzyć własne skrypty śródliniowe w Node.js,
31. zdefiniować własne moduły w Node.js,
32. implementować obsługę zdarzeń w Node.js,
33. implementować odczyt danych z pliku w Node.js,
34. implementować zapis danych w pliku w Node.js,
35. wykonywać operacje na plikach w Node.js,
36. kopiować pliki za pomocą strumieni w Node.js,
37. implementować prosty serwer HTTP w Node.js,
38. implementować odpowiedź typu HTML w Node.js,
39. zdefiniować routing w Node.js,
40. implementować odpowiedź JSON w Node.js,
41. wykorzystać strumienie i HTTP w node.js,
42. stworzyć aplikację z wykorzystaniem frameworku express.js w Node.js,
43. zainstalować MongoDB w Node.js,
44. implementować podstawowe polecenia Mongo,
45. połączyć Node.js z bazą danych,
46. dodawać nowe wartości do nazy danych z poziomu Node.js,
47. wykorzystać Web Socket w Node.js,
48. wykorzystać inne biblioteki Node.js,
49. zainstalować język Python na serwerze,
50. dokonać edycji zmiennych środowiskowych na serwerze w celu instalacji Django,
51. zainstalować Django na serwerze,
52. tworzyć projekt aplikacji webowej w Django,
53. przygotować bazę danych do projektu aplikacji wykonanej w Django,
54. opracować model danych stosowanych w aplikacji webowej wykonane w Django,
55. wprowadzić dane do tabel w bazie danych dla aplikacji webowej wykonanej w Django,
56. implementować formularz logowania się użytkowników do aplikacji webowej wykonanej w Django,
57. zautomatyzować procesy np. powiadamiane użytkowników w aplikacji webowej wykonanej w Django,
58. zaprogramować aplikacje internetowe w wybranym języku programowania np. PHP, C#, Python, JavaScript,
59. zaprogramować aplikacje webowe wykorzystujące mechanizm sesji i ciasteczek,
60. zaprogramować aplikacje webowe zawierające dynamiczne formularze,
61. zaprogramować systemy logowania do aplikacji webowej,
62. zaprogramować system kontroli dostępu do określonych elementów witryny,
63. zaprogramować aplikacje webowe z dostępem do baz danych,
64. zaprogramować wybrane elementy e-sklepu,
65. zaprogramować wybrane elementy portalu społecznościowego,
66. zaprogramować wybrane elementy serwisu ogłoszeń internetowych,
67. zaprogramować wybrane elementy serwisu rezerwacyjnego,
68. testować zaprogramowaną aplikację użytkownika,
69. dokumentować kod zaprogramowanej aplikacji,
70. publikować aplikację webową na serwerze.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Programowanie aplikacji webowych z wykorzystaniem frameworków** | 1. Programowanie aplikacji webowych w jQuery. |  | - dobrać środowisko programistyczne do projektu aplikacji webowej,  - dobrać niezbędne narzędzia do pracy w środowisku programistycznym,  - dobrać środowisko pracy z wybranymi frameworkami Node.js i Django, |  | Klasa III |
| 2. Programowanie aplikacji webowych w Node.js. |  | - rozpocząć pracę z Node.js,  - rozpoznać wbudowane moduły Node.js,  - opisać kluczowe techniki stosowane w frameworku Node.js;  - stosować pliki, strumienie i bufory w Node.js,  - zainstalować menadżer pakietów npm dla Node.js,  - zainstalować framework expres.js dla Node.js, |  | Klasa III |
| 3. Programowanie aplikacji webowej w Django. |  | - opisać podstawowe cech frameworku Django,  - opisać podstawowe elementu projektu wykonanego w Django,  - opisać kolejne etapy pracy nad projektem aplikacji tworzonej w Django,  - opisać system komentarzy w Django,  - scharakteryzować profile użytkowników w Django, |  | Klasa III |
| **II. Programowanie aplikacji webowej w wybranym języku programowania.** | 1. Programowanie wybranych mechanizmów w programowaniu aplikacji webowych. |  | - programować aplikacje internetowe w wybranym języku programowania np. PHP, C#, Python, JavaScript,  - programować aplikacje webowe wykorzystujące mechanizm sesji i ciasteczek,  - programować aplikacje webowe zawierające dynamiczne formularze, | - programować systemy logowania do aplikacji webowej,  - programować system kontroli dostępu do określonych elementów witryny,  - programować aplikacje webowe z dostępem do baz danych, | Klasa III |
| 2. Programowanie wybranych elementów aplikacji webowej. |  | - testować zaprogramowaną aplikację użytkownika,  - dokumentować kod zaprogramowanej aplikacji,  - publikować aplikację webową na serwerze. | - programować wybrane elementy e-sklepu,  - programować wybrane elementy portalu społecznościowego,  - programować wybrane elementy serwisu ogłoszeń internetowych,  - programować wybrane elementy serwisu rezerwacyjnego, | Klasa III |
| Razem | |  |  |  |  |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągania założonych celów kształcenia w zakresie przedmiotu jest opracowanie odpowiednich dla danego zawodu procedur, a w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

**METODY NAUCZANIA**

Dla przedmiotu Tworzenie aplikacji webowych, który należy do przedmiotów praktycznych, by dominującą metodą kształcenia powinny były ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku, polegająca na analizowaniu przypadku opisującego problem. W trakcie realizacji zajęć nauczyciel powinien:

– motywować uczniów do pracy,

– dostosowywać stopień trudności planowanych ćwiczeń do możliwości uczniów,

– uwzględniać zainteresowania uczniów,

– przygotowywać zadania o różnym stopniu trudności i złożoności,

– zachęcać uczniów do korzystania z różnych źródeł informacji zawodowej,

– stosować metody aktywizujące,

– stosować nowoczesne środki kształcenia, np. tablice multimedialne.

**ŚRODKI DYDAKTYCZNE**

Pracownia informatyczna wyposażona w stanowiska komputerowe dla uczniów i nauczyciela z dostępem do Internetu wyposażone w monitor o 21” o rozdzielczości co najmniej Full HD (w przypadku monitorów z mniejszą rozdzielczością należy rozważyć 2 monitory), środowisko do tworzenia aplikacji webowych np. Visual Studio, Eclipse Jet Brains lub inne, zainstalowane frameworki i języki programowania, zainstalowane lokalnie oprogramowanie serwera usług hostingowych i baz danych, dostęp do zdalnego serwera z usługami hostingowymi i bazami danych, dostęp do repozytorium Git z zamieszczonymi przykładami, biblioteka cyfrowa zawierająca dokumentację do zainstalowanych aplikacji, frameworków i języków programowania oraz podręcznikami. Dodatkowo w pracowni znajduje się tablica interaktywna, projektor, nagłośnienie, drukarka sieciowa, stoły i krzesła umożliwiające pracę grupową bez komputerów. Wskazane jest aby uczniowie mieli dostęp do aplikacji umożliwiających pracę i komunikację grupową np. Team, Slack.

**FORMY ORGANIZACYJNE**

Zajęcia z przedmiotu Tworzenie aplikacji webowych powinny być prowadzone indywidualnie lub zespołowo. Należą one do grupy zajęć praktycznych. O liczebności grup ogólnie mówi §6 Rozporządzenia Ministra Edukacji Narodowej z dnia 22 lutego 2019 r. w sprawie praktycznej nauki zawodu. W celu spełnienia wymienionych w nim warunków realizacji kształcenia praktycznego, zajęcia edukacyjne powinny się odbywać w pracowni z podziałem na grupy o liczebności maksymalnie do 12 osób.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Do głównych metod sprawdzania osiągnięć edukacyjnych ucznia należą ćwiczenia praktyczne oraz metoda projektów. Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu na podstawie kryteriów, przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* realizowanych zadań praktycznych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania zadań praktycznych,
* produktu projektu i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

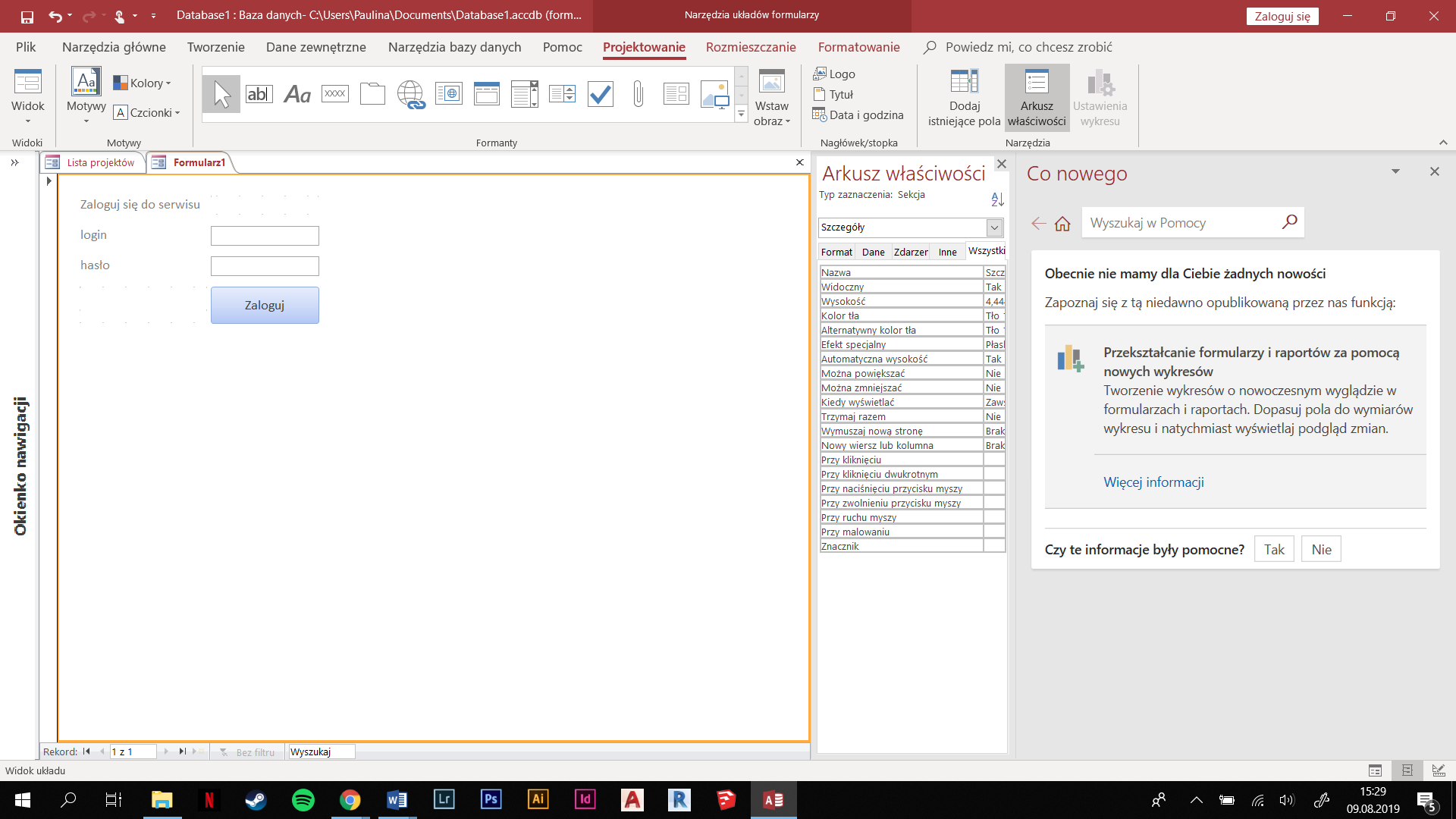
Wskazane jest, aby uczniowie dokonywali także własnej samooceny pracy i kolegów z zespołu wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

**Przykładowe zadania**

Zadanie 1

Zaprogramuj animowany przycisk aplikacji webowej z wykorzystaniem biblioteki jQuery o wymiarach 200x80 pikseli w kolorze #4444FF z napisem kliknij mnie umieszczonym w środku przycisku w kolorze #FFFFFF. Po kliknięciu przycisku zmienia się jego szerokość do 500 pikseli, tło wypełnienia uzyskuje przezroczystość 0.4, tekst jest 2-krotnie większy i pojawia się obramowanie przycisku o szerokości 10 pikseli. Czas animacji przycisku wynosi 1 sekundę.

Zadanie 2

Zaprogramuj formularz logowania do aplikacji webowej jak na rysunku. Skrypt obsługujący formularz reaguje na następujące dane użytkowników: login: admin hasło: zaq1@WSX, login: user hasło: user1234, po wprowadzeniu danych użytkownika admin na ekranie aplikacji powinien pojawić się napis Panel administracyjny a po podaniu danych użytkownika na ekranie powinien pojawić się napis Konto użytkownika user. W przypadku podania błędnych danych powinien pojawić się ponownie formularz logowania.

Zadanie 3

Utwórz bazę danych o nazwie **serwis** na serwerze, w bazie utwórz tabelę menu i podmenu oraz wypełnij danymi zgodnie z relacją na rysunkiem poniżej.



Na podstawie tak utworzonych danych napisz skrypt aplikacji webowej, który wygeneruje menu aplikacji webowej wraz z podmenu na podstawie danych zawartych w tabelach. Teksty menu powinny być rozmieszczone rosnąco zgodnie z id w tabeli menu. Teksty w podmenu powinny być posortowane rosnąco zgodnie z literami alfabetu. Jeśli skrypt poprawnie wyświetla menu to do tabeli podmenu dodaj kolejną pozycję nazwa ->Deskorolki, id\_m -> 1 i sprawdź jak teraz wyświetla się menu aplikacji webowej oraz na której pozycji w podmenu jest nazwa Deskorolki.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Strategia przeprowadzanej ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych. Zebrane dane zostaną poddane analizie ilościowej i jakościowej przy użyciu narzędzia, którym jest statystyka matematyczna. Przydatnym narzędziem w tej analizie może być na przykład korzystanie z platformy testowej office 365.com lub podobnej, która daje możliwość analizy, które z pytań testowych sprawiają trudność.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego. Spowoduje to podwyższenie jakości kształcenia i znacząco wpłynie na indywidualne wyniki uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. tworzenie aplikacji webowej z wykorzystaniem frameworku Node.js,

2. tworzenie aplikacji webowej z wykorzystaniem frameworku Django,

3. tworzenie aplikacji webowej z wykorzystaniem frameworku jQuery,

4. tworzenie aplikacji webowej z wykorzystaniem wybranego języka: PHP, C#, Python, JavaScriopt,

5. programowanie wybranych elementów aplikacji webowych,

6. programowanie aplikacji webowych z wykorzystaniem baz danych.

**ZALECANA LITERATURA**

* Gniewomir Sarbicki, Python. Kurs dla nauczycieli i studentów, wyd. Helion,
* Adriaan de Jonge, Phillip Dutson, jQuery, jQuery UI oraz jQuery Mobile. Receptury, wyd. Helion,
* Jon Duckett, JavaScript i jQuery. Interaktywne strony WWW dla każdego. Podręcznik Front-End Developera wyd. Helion,
* David Herron, Platforma Node.js. Przewodnik webdevelopera. Wydanie III, wyd. Helion,
* Antonio Mele, Django 2. Praktyczne tworzenie aplikacji sieciowych. Wydanie II, wyd. Helion.

## Tworzenie aplikacji mobilnych

**Cele ogólne przedmiotu**

1. Nabycie umiejętności tworzenia interfejsu aplikacji mobilnej dla środowiska iOS lub Android;
2. Nabycie umiejętności łączenia kodu programu z elementami UI w wybranym środowisku iOS lub Android;
3. Nabycie umiejętności korzystania z zasobów sprzętowych urządzenia mobilnego w aplikacji mobilnej;
4. Nabycie umiejętności wykorzystania zewnętrznych zasobów przez aplikacje mobilną;
5. Nabycie umiejętności zaawansowanego programowania aplikacji mobilnych;
6. Nabycie umiejętności uruchamiania, testowania i publikowania aplikacji mobilnych;
7. Kształtowanie postawy aktywnego słuchania i włączania się do dyskusji;
8. Kształtowanie postawy ustawicznego kształcenia i doskonalenia.

**Cele operacyjne**

**Uczeń potrafi:**

1. stosować wybrane środowisko do programowania aplikacji mobilnych tj. Visual Studio, X-Code, Android Studio,
2. stosować narzędzia wybranego środowiska programistycznego,
3. korzystać z gotowych motywów aplikacji mobilnych oferowanych przez środowisko programistyczne,
4. stworzyć aplikację mobilną zgodnie z wzorcem MVC,
5. stworzyć aplikację mobilna zgodnie z wzorcem MVVM,
6. wykorzystywać do budowy interfejsu aplikacji elementy UI dla systemu iOS lub Android,
7. wykorzystywać język XAML do budowy interfejsu aplikacji mobilnej,
8. zastosować podstawowe typy zmiennych do przechowywania danych,
9. zastosować tabele do przechowywania wielu danych tego samego typu,
10. zastosować struktury do przechowywania danych różnego typu,
11. zastosować obiekty do przechowywania danych,
12. zastosować instrukcje warunkowe w programowaniu aplikacji mobilnych,
13. zastosować pętle w programowaniu aplikacji mobilnych,
14. zastosować instrukcji przełączające w programowaniu aplikacji mobilnych,
15. przesłać dane do aplikacji po kliknięciu w przycisk interfejsu UI,
16. prezentować dane z aplikacji na elementach interfejsu UI,
17. pobrać czas, datę i inne dane systemowe z urządzenia mobilnego,
18. udostępnić aplikacji mobilnej kontakty z urządzenia mobilnego,
19. udostępnić aplikacji mobilnej domyślne zasoby takie jak zdjęcia, muzyka, filmy danego urządzenia mobilnego,
20. zaprezentować udostępnione dane i zasoby z urządzenia mobilnego w aplikacji mobilnej np. zegar, pokaz zdjęć, powiadomienia itp.
21. dodać animacje do elementów interfejsu UI,
22. przesłać dane pomiędzy aktywnościami,
23. stworzyć aplikację wykorzystującą wiele elementów interfejsu UI np. kalkulator, gra memo,
24. stworzyć aplikację do malowania na ekranie urządzenia mobilnego rozpoznającą dotyk: stuknięcie, przytrzymanie, przeciągnie,
25. przechowywać preferencje użytkownika dla danej aplikacji w urządzeniu mobilnym,
26. wykorzystać lokalizację GPS w aplikacji mobilnej,
27. zapisać dane z aplikacji w pamięci masowej urządzenia,
28. wykorzystać standard JSON w celu pobierania i przesyłania danych z poziomu aplikacji mobilnej do Internetu,
29. stworzyć aplikację mobilną korzystającą z bazy danych np. dziennik zadań, pamiętnik itp.,
30. wykorzystać dedykowane biblioteki do programowania zaawansowanych aplikacji mobilnych np. prostych gier 2D,
31. dostosować interfejs aplikacji mobilnej do konkretnego rodzaju urządzenia,
32. testować i uruchamiać aplikacje mobilne na emulatorach urządzeń,
33. testować i uruchamiać aplikacje mobilne na urządzeniach,
34. publikować aplikacje mobilne w dedykowanych sklepach.

**MATERIAŁ NAUCZANIA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dział programowy** | **Tematy jednostek metodycznych** | **Liczba godz.** | **Wymagania programowe** | | **Uwagi o realizacji** |
| Podstawowe  Uczeń potrafi: | Ponadpodstawowe  Uczeń potrafi: | Etap realizacji |
| **I. Programowanie aplikacji mobilnych** | 1. Środowisko programistyczne aplikacji mobilnych. |  | - stosować wybrane środowisko do programowania aplikacji mobilnych tj. Visual Studio, X-Code, Android Studio,  - stosować narzędzia wybranego środowiska programistycznego,  - korzystać z gotowych motywów aplikacji mobilnych oferowanych przez środowisko programistyczne, |  | Klasa IV |
| 2.Tworzenie interfejsu aplikacji. |  | - tworzyć aplikację mobilną zgodnie z wzorcem MVC,  - tworzyć aplikację mobilna zgodnie z wzorcem MVVM,  - wykorzystywać do budowy interfejsu aplikacji elementy UI dla systemu iOS lub Android, | - wykorzystywać język XAML do budowy interfejsu aplikacji mobilnej, | Klasa IV |
| 3. Programowanie logiki aplikacji. |  | - stosować podstawowe typy zmiennych do przechowywania danych,  - stosować tabele do przechowywania wielu danych tego samego typu,  - stosować instrukcje warunkowe w programowaniu aplikacji mobilnych,  - stosować pętle w programowaniu aplikacji mobilnych,  - stosować instrukcji przełączające w programowaniu aplikacji mobilnych, | - stosować struktury do przechowywania danych różnego typu,  - stosować obiekty do przechowywania danych,  - stosować rozbudowane instrukcje warunkowe w programowaniu aplikacji mobilnych, | Klasa IV |
| 4. Programowanie aplikacji korzystających z elementów UI i zasobów systemowych |  | - przesyłać dane do aplikacji po kliknięciu w przycisk interfejsu UI,  - prezentować dane z aplikacji na elementach interfejsu UI,  - pobierać czas, datę i inne dane systemowe z urządzenia mobilnego,  - przesyłać dane pomiędzy aktywnościami, | - udostępnić aplikacji mobilnej kontakty z urządzenia mobilnego,  - udostępnić aplikacji mobilnej domyślne zasoby takie jak zdjęcia, muzyka, filmy danego urządzenia mobilnego,  - prezentować udostępnione dane i zasoby z urządzenia mobilnego w aplikacji mobilnej np. zegar, pokaz zdjęć, powiadomienia itp.  - dodać animacje do elementów interfejsu UI, | Klasa IV |
| **II. Zaawansowane programowanie aplikacji mobilnych.** | 1. Zaawansowane techniki programowania aplikacji mobilnych. |  | - stworzyć aplikację wykorzystującą wiele elementów interfejsu UI np. kalkulator, gra memo  - przechowywać preferencje użytkownika dla danej aplikacji w urządzeniu mobilnym,  - wykorzystać lokalizację GPS w aplikacji mobilnej,  - dostosować interfejs aplikacji mobilnej do konkretnego rodzaju urządzenia, | - stworzyć aplikację do malowania na ekranie urządzenia mobilnego rozpoznającą dotyk: stuknięcie, przytrzymanie, przeciągnie,  - wykorzystać dedykowane biblioteki do programowania zaawansowanych aplikacji mobilnych np. prostych gier 2D,  - zapisywać dane z aplikacji w pamięci masowej urządzenia,  - stworzyć responsywny interfejs aplikacji mobilnej dla określonego systemu iOS lub Android, | Klasa IV |
| 2. Dostęp aplikacji mobilnej do zewnętrznych źródeł danych. |  | - wykorzystać standard JSON w celu pobierania danych z Internetu do aplikacji mobilnej,  - stworzyć aplikację mobilną korzystającą z lokalnej bazy danych np. pamiętnik itp., | - wykorzystać standard JSON w celu przesyłania danych z poziomu aplikacji mobilnej do Internetu,  - stworzyć aplikację mobilną korzystającą ze zdalnej bazy danych np. dziennik zadań, | Klasa IV |
| **III. Testowanie i publikowanie aplikacji mobilnych** | Testowanie aplikacji |  | - testować i uruchamiać aplikacje mobilne na emulatorach urządzeń,  - testować i uruchamiać aplikacje mobilne na urządzeniach, | - publikować aplikacje mobilne w dedykowanych sklepach, | Klasa IV |
| RAZEM | |  |  |  |  |

**PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU**

Warunkiem osiągania założonych celów kształcenia w zakresie przedmiotu jest opracowanie odpowiednich dla danego zawodu procedur, a w tym:

* zaplanowanie lekcji (wskazanie celów szczegółowych jakie powinny zostać osiągnięte),
* wykorzystanie różnorodnych metod nauczania (w szczególności takich, które aktywizują ucznia do pracy),
* dobór środków dydaktycznych do treści i celów nauczania,
* dobór formy pracy z uczniami – określenie ilości osób w grupie, określenie indywidualizacji zajęć,
* systematyczne sprawdzanie wiedzy i umiejętności uczniów poprzez sprawdziany w formie tekstu wielokrotnego wyboru oraz testów praktycznych i innych form sprawdzania wiedzy i umiejętności w zależności od metody nauczania,
* stosowanie oceniania sumującego i kształtującego,
* przeprowadzenie ewaluacji doboru treści nauczania do założonych celów, metod pracy, środków dydaktycznych, sposobu oceniania i informacji zwrotnej od ucznia.

**METODY NAUCZANIA**

Dla przedmiotu Tworzenie aplikacji mobilnych, który należy do przedmiotów praktycznych, zaleca się, by dominująca metodą kształcenia były ćwiczenia praktyczne, które ułatwią uczniom samodzielne zbieranie i analizowanie informacji oraz metoda przypadku, polegająca na analizowaniu przypadku opisującego problem. W trakcie realizacji zajęć nauczyciel powinien:

– motywować uczniów do pracy,

– dostosowywać stopień trudności planowanych ćwiczeń do możliwości uczniów,

– uwzględniać zainteresowania uczniów,

– przygotowywać zadania o różnym stopniu trudności i złożoności,

– zachęcać uczniów do korzystania z różnych źródeł informacji zawodowej,

– stosować metody aktywizujące,

– stosować nowoczesne środki kształcenia, np. tablice multimedialne.

**ŚRODKI DYDAKTYCZNE**

Pracownia komputerowa wyposażona w stanowiska komputerowe dla każdego ucznia i nauczyciel z monitorami pracującymi w minimalnej rozdzielczości Full HD, dostępem do Internetu, zainstalowanym środowiskiem programistycznym dla danego systemu mobilnego w którym będą tworzone aplikacje, urządzenia mobilne, oprogramowanie do kontroli wersji, oprogramowanie do tworzenia i modyfikowania grafiki rastrowej i wektorowej, udostępnione repozytoria przykładowych aplikacji mobilnych, szablony aplikacji do wykorzystania w trakcie zajęć, wydzielone miejsce do pracy zespołowej wyposażone w flipchart i stoły. W pracowni powinna znajdować się biblioteczka z dokumentacją dotyczącą zainstalowanego oprogramowania, dokumentacją deweloperską oraz literatura, projektor, tablica multimedialna, zestaw nagłaśniający. Wskazane jest również korzystanie z oprogramowania do pracy zespołowej i komunikacji typu Teams, Slack. Uczniowie powinni mieć również do sklepu z aplikacjami na poziomie deweloperskim.

**FORMY ORGANIZACYJNE**

Zajęcia z przedmiotu powinny być prowadzone indywidualnie lub zespołowo. Należą one do grupy zajęć praktycznych. O liczebności grup ogólnie mówi §6 Rozporządzenia Ministra Edukacji Narodowej z dnia 22 lutego 2019 r. w sprawie praktycznej nauki zawodu. W celu spełnienia wymienionych w nim warunków realizacji kształcenia praktycznego, zajęcia edukacyjne powinny się odbywać w pracowni z podziałem na grupy o liczebności maksymalnie do 12 osób.

**PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA**

Do głównych metod sprawdzania osiągnięć edukacyjnych ucznia należą ćwiczenia praktyczne oraz metoda projektów. Sprawdzanie osiągnięć ucznia powinno odbywać się przez cały czas realizacji programu na podstawie kryteriów, przedstawionych na początku zajęć. Sprawdzanie i ocenianie osiągnięć uczniów powinno dostarczyć informacji dotyczących zakresu i stopnia realizacji celów kształcenia działu programowego. Zaleca się systematyczne ocenianie postępów ucznia.

Osiągnięcia uczniów należy oceniać na podstawie:

* realizowanych zadań praktycznych,
* ukierunkowanej obserwacji pracy ucznia podczas wykonywania zadań praktycznych,
* produktu projektu i jego prezentacji.

Obserwując czynności ucznia podczas wykonywania ćwiczeń i dokonując oceny jego pracy, należy zwrócić uwagę na:

* umiejętność radzenia sobie w sytuacjami zbliżonych do rzeczywistych zadań zawodowych,
* umiejętność pracy w zespole,
* umiejętność korzystania z różnych źródeł informacji (norm, katalogów, dokumentacji technicznej – w tym w języku obcym i z wykorzystaniem technologii informacyjnej).

Wskazane jest, aby uczniowie dokonywali także własnej samooceny pracy i kolegów z zespołu wg. zaproponowanych przez nauczyciela arkuszy samooceny i oceny oraz sprawdzianów postępów.

**Przykładowe zadania**

Zadanie 1

Zaprogramuj aplikację mobilną na wskazane urządzenie mobilne która:

- po uruchomieniu wyświetli w postaci siatki 4x4 16 ostatnich wykonanych zdjęć znajdujących się w bibliotece urządzenia zawierającej zdjęcia,

- po kliknięciu na danej miniaturze zdjęcia zostanie ono wyświetlone na całym ekranie bez przeskalowania,

- po kliknięciu w zdjęcie na pełnym ekranie zostanie ponownie wyświetlony ekran z siatką zdjęć.

Zadanie 2

Zaprogramuj aplikację mobilną prostego kalkulatora realizującą podstawowe operacje matematyczne takie jak dodawania, odejmowanie, mnożenie i dzielenia w tym celu wykonaj następujące elementy projektu:

- interfejs aplikacji zawierający 16 przycisków w tym:

- 10 z cyframi od „0 do 9”,  
- 4 z symbolami działań arytmetycznych „+, -, \*, /”,  
- przycisk z „=”,  
- przycisk z „C”,

- nad przyciskami umieść pole tekstowe,

- kliknięcie dowolnego przycisku z cyframi powoduje dopisanie na z lewej strony pola tekstowego odpowiedniej cyfry,

- naciśnięcie przycisku „C” czyści pole tekstowe oraz zeruje wszystkie zmienne w aplikacji,

- naciśnięcie dowolnego przycisku z działaniem arytmetycznym powoduje zapamiętanie wartości liczbowej wprowadzonej w polu tekstowym, zapamiętanie rodzaju działania znajdującego się na przycisku i wyczyszczenie pola tekstowego,

- naciśnięcie przycisku „=” powoduje sprawdzenie czy jest zapamiętane działanie do wykonania

jeśli tak to wykonywane jest działanie na podstawie zapisanej poprzednio wartości liczbowej, zapisanego działania oraz odczytanej wartości liczbowej z pola tekstowego, a wynik działania wyświetlony jest w polu tekstowym,

jeśli nie to aplikacja nie robi niczego.

Tak wykonaną aplikację przetestuj na wybranym urządzeniu przenośnym. Gdy stwierdzisz poprawność działania aplikacji, zastanów się jakie elementy można poprawić aby logika aplikacji była bardziej odporna na błędy które wynikają z przyjętych założeń.

Zadanie 3

Stwórz aplikację mobilną wykorzystującą geolokalizację dostępną w urządzeniu mobilnym. Zadaniem aplikacji będzie zapisanie aktualnej pozycji w postaci długości i szerokości geograficznej. Następnie aplikacja co 1 minutę będzie odczytywać pozycję użytkownika i dodawać do już istniejących danych. W każdej chwili użytkownik będzie miał możliwość kliknięcia w przycisk pokaż znajdujący się na ekranie, co spowoduje graficzną prezentację w postaci połączonych za pomocą linii następujących po sobie odczytów. Aplikacja po zamknięciu usuwa zgromadzone dane. Po zaprogramowaniu aplikacji i jej przetestowaniu zastanów się czy można w jakiś sposób przechowywać dane poszczególnych cykli odczytów tzn. kolejne cykle są dopisywane do już istniejącej bazy odczytów.

**PROPONOWANE METODY EWALUACJI PRZEDMIOTU**

Strategia przeprowadzanej ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów z poszczególnych działów programowych. Zebrane dane zostaną poddane analizie ilościowej i jakościowej przy użyciu narzędzia, którym jest statystyka matematyczna. Przydatnym narzędziem w tej analizie może być na przykład korzystanie z platformy testowej office 365.com lub podobnej, która daje możliwość analizy, które z pytań testowych sprawiają trudność.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego. Spowoduje to podwyższenie jakości kształcenia i znacząco wpłynie na indywidualne wyniki uczniów z egzaminu zawodowego.

Kluczowe umiejętności podlegające ewaluacji w ramach przedmiotu dotyczą:

1. programowanie logiki aplikacji mobilnej w wybranym języku programowania,

2. tworzenie interfejsu użytkownika z wykorzystaniem języka XAML,

3. tworzenie interfejsu użytkownika na podstawie dostępnych elementów UI,

4. programowanie wykorzystującej zasoby systemowe urządzenia mobilnego,

5. stosowanie zaawansowane technik programowania aplikacji mobinej,

6. tworzenie aplikacji mobilnych korzystających z zewnętrznych źródeł danych

7. dokumentowanie i testowanie projektów programistycznych.

**ZALECANA LITERATURA**

* Dawn Griffiths, David Griffiths, Android. Programowanie aplikacji. Rusz głową! Wydanie II, wyd. Helion,
* Marcin Płonkowski, Android Studio. Tworzenie aplikacji mobilnych. wyd. Helion,
* Matt Neuburg, iOS 12. Wprowadzenie do programowania w Swifcie. Wydanie V, wyd. Helion,
* Steven F. Daniel, Xamarin. Tworzenie interfejsów użytkownika, wyd. Helion.

**Kwalifikacje: INF.03.** Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych oraz **INF.04.** Projektowanie, programowanie i testowanie aplikacji

## Praktyka zawodowa

**Cele ogólne przedmiotu**

1. Poznanie

* przepisów bezpieczeństwa i higieny pracy, ochrony przeciwpożarowej i ochrony środowiska w miejscu pracy,
* struktury organizacyjnej przedsiębiorstwa,
* organizacji działalności gospodarczej i administracyjnej przedsiębiorstwa,
* zasad organizacji stanowiska pracy,
* przepisów prawnych związanych z zatrudnieniem,
* obowiązków pracownika i pracodawcy,
* zadań zawodowych w warunkach zakładu pracy,

1. Nabycie umiejętności

* przestrzegania przepisów BHP, przeciwpożarowych i ochrony środowiska,
* wykonywania zadań zawodowych w warunkach zakładu pracy.

1. Kształtowanie postawy, świadomości

* stosowania zasad kultury osobistej i ogólnie przyjętych norm zachowania w miejscu pracy;
* odpowiedzialności za realizowane działania;
* kreatywności w rozwiązywaniu problemów;
* stosowania właściwej techniki twórczego myślenia przy rozwiązaniu problemu;
* doskonalenia jakości wykonywanych działań;
* analizowania i oceny podejmowanych działań;
* pracy w zespole;
* przestrzegania przepisów prawa pracy;

**Cele operacyjne**

Uczeń potrafi:

1. stosować przepisy bezpieczeństwa i higieny pracy, ochrony przeciwpożarowej i ochrony środowiska w miejscu pracy,
2. scharakteryzować organizację działalności gospodarczej i administracyjnej przedsiębiorstwa,
3. charakteryzować strukturę organizacyjną przedsiębiorstwa oraz służb informatycznych w przedsiębiorstwie,
4. rozróżniać rodzaj działalności przedsiębiorstwa,
5. organizować własne stanowisko pracy,
6. dobrać konfigurację sprzętu i oprogramowania komputerowego dla różnych zastosowań,
7. realizować zadania zawodowe w warunkach zakładu pracy,
8. posługiwać się gotowymi pakietami oprogramowania użytkowego i narzędziowego,
9. zbierać dane dla systemów przetwarzania informacji,
10. korzystać z zasobów lokalnych sieci komputerowych i internetu,
11. organizować i wykonywać prace w zakresie usług informatycznych dla użytkowników lub zleceniodawców,
12. zaprojektować, stworzyć i administrować witrynami internetowymi oraz innymi technologiami webowymi,
13. utworzyć aplikacje internetowe,
14. utworzyć programy desktopowe,
15. utworzyć programy mobilne,
16. modelować, projektować i zajmować się drukiem 3D,
17. administrować bazami danych i systemami przetwarzania informacji w przedsiębiorstwie,
18. zaprojektować bazy danych na użytek przedsiębiorstwa,
19. administrować bazami danych i systemami przetwarzania informacji w przedsiębiorstwie informatycznym,
20. posługiwać się terminologią zawodową w języku angielskim,
21. korzystać z instrukcji i literatury w języku angielskim,
22. pracować w zespole.

**Materiał nauczania**

* + - 1. Organizacja stanowiska pracy
* przestrzeganie przepisów bezpieczeństwa i higieny pracy, ochrony przeciwpożarowej i ochrony środowiska,
* udzielanie pierwszej pomocy w stanach zagrożenia zdrowia i życia,
* organizowanie stanowiska pracy programisty według zasad ergonomii,
* rozpoznawanie czynników szkodliwych i uciążliwych występujących w miejscu pracy,
* stosowanie zasad współpracy w zespole,
* przestrzeganie przepisów, regulaminów i zasad obowiązujących pracowników firmy,
* charakterystyka form działalności gospodarczej i administracyjnej firmy ,
* określanie struktury organizacyjnej firmy i charakteru jej działalności,
* określanie miejsca i znaczenia prac informatycznych w działalności firmy,

1. Obsługa oprogramowania używanego w firmie

* obsługa oprogramowania systemowego i użytkowego stosowanego w firmie,
* ochrona danych, programów i procesów przetwarzania informacji.

1. Organizacja i wyposażenie przedsiębiorstwa na potrzeby przetwarzania informacji

* określenie zakresu prac prowadzonych w firmie,
* wykorzystywanie technicznych środków do zbierania informacji przeznaczonych do przetwarzania,
* wprowadzanie danych do systemu, przedstawienie wyników przetwarzania danych i ich zastosowanie,
* administrowanie danymi,
* ochrona i bezpieczeństwo gromadzonych danych,
* wykorzystywanie sieci internet w działalności firmy.

1. Projektowanie i programowanie.

* określenie elementów procesu projektowania, programowania i uruchamiania programów komputerowych i systemów przetwarzania danych,
* organizowanie pracy projektantów i programistów na stanowiskach komputerowych,
* wybieranie odpowiedniego wariantu rozwiązania danego problemu programistycznego,
* projektowanie aplikacji,
* programowane aplikacji,
* czytanie dokumentacji technicznej oprogramowania i języka programowania,
* obsługa programów do wspomagania procesu projektowania i programowania.

**Procedury osiągania celów kształcenia przedmiotu**

**Warunki realizacji**

Praktyka zawodowa realizowana jest w klasie III i IV w wymiarze 20 dni roboczych (4 tygodnie). Czas pracy ucznia wynosi 7 godz. dziennie. Łączny czas trwania praktyki wynosi 280 godz. Praktykę zawodową można odbywać w kraju lub w krajach Unii Europejskiej.

Za podstawą programową, miejscem realizacji praktyk zawodowych mogą być:

* przedsiębiorstwa usługowe zajmujące się projektowaniem, tworzeniem i obsługą systemów informatycznych,
* przedsiębiorstwa zajmujące się hostingiem oraz projektowaniem, tworzeniem i administracją witryn internetowych oraz innych technologii webowych,
* przedsiębiorstwa zajmujące się tworzeniem programów desktopowych i aplikacji internetowych,
* przedsiębiorstwa zajmujące się tworzeniem aplikacji mobilnych,
* przedsiębiorstwa zajmujące się projektowaniem UI,
* przedsiębiorstwa zajmujące się modelowaniem, projektowaniem i drukiem 3D,
* inne podmioty stanowiące potencjalne miejsce zatrudnienia absolwentów szkół prowadzących kształcenie w zawodzie.

Plan i organizację zajęć w ramach praktyki należy stosować elastycznie i dostosować do możliwości danego przedsiębiorstwa, mając na uwadze realizację założonych w programie celów kształcenia. Przewidziana programem nauczania praktyka zawodowa powinna odbywać się na stanowiskach, na których w przyszłości będzie pracował technik programista, np.: programisty, projektanta czy administratora baz danych. Praktyka powinna stwarzać możliwość poznania praktycznych zastosowań informatyki i organizacji prac informatycznych w przedsiębiorstwie podczas wykonywania zadań na rzecz użytkowników lub zleceniodawców. Przed przystąpieniem do zajęć uczeń powinien poznać obowiązujące przepisy bezpieczeństwa i higieny pracy. Uczniowie odbywający praktykę zawodową zobowiązani są do prowadzenia dzienniczka praktyk, w którym odnotowują tematy prac i zakres wykonywanych czynności. Zapisy powinny być sprawdzane i potwierdzane przez osobę prowadzącą praktykę zawodową.

**Propozycje metod sprawdzania osiągnięć edukacyjnych ucznia**

Sprawdzanie i ocenianie osiągnięć uczniów powinno odbywać się na bieżąco podczas realizacji programu praktyki zawodowej. Kryteria oceniania powinny dotyczyć poziomu oraz zakresu opanowania przez ucznia umiejętności wynikających z celów kształcenia. Ze względu na charakter zajęć w procesie oceniania dominować powinna obserwacja pracy ucznia oraz ocena efektów jego pracy. Dokonując oceny pracy uczniów należy uwzględnić:

* przestrzeganie dyscypliny pracy (punktualność, rzetelność w wykonywaniu zleconych zadań)
* organizację pracy,
* samodzielność wykonywania zadań zawodowych,
* pracowitość,
* jakość wykonywanej pracy,
* podejście ucznia do zadań zawodowych i współpracowników, kulturę osobistą.

Po odbyciu przez ucznia praktyki zawodowej, opiekun z ramienia przedsiębiorstwa powinien wpisać w dzienniczku praktyk opinię o pracy ucznia oraz wystawić ocenę końcową.

# PROJEKT EWALUACJI PROGRAMU NAUCZANIA ZAWODU TECHNIK PROGRAMISTA

Cele ewaluacji

Określenie jakości i skuteczności realizacji programu nauczania zawodu w zakresie:

– osiągania efektów kształcenia,

– doboru oraz zastosowania form, metod i strategii dydaktycznych,

– współpracy z pracodawcami,

– wykorzystania bazy techno-dydaktycznej.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Faza refleksyjna** | | | | |
| **Obszar badania** | **Pytania kluczowe** | **Wskaźniki świadczące o efektywności** | **Metody, techniki badania/narzędzia** | **Termin badania** |
| Układ materiału nauczania | 1. Czy program nauczania uwzględnia spiralną strukturę treści? 2. Czy efekty kształcenia kluczowe dla zawodu zostały podzielone na materiał nauczania w taki sposób, aby były rozwijane przez kilka przedmiotów w całym cyklu kształcenia w zakresie danej kwalifikacji? 3. Czy wszyscy nauczyciele współpracują przy ustalaniu kolejności realizacji treści programowych? | Program nauczania umożliwia przygotowanie do egzaminu potwierdzającego kwalifikacje | Metoda – jakościowa  Technika – ankieta  Narzędzia – kwestionariusz ankiety dla nauczycieli | wrzesień |
| Relacje między poszczególnymi elementami i częściami programu | 1. Czy program nauczania uwzględnia podział na przedmioty teoretyczne i praktyczne? 2. Czy program nauczania uwzględnia korelację międzyprzedmiotową? 3. Czy korelowanie treści nauczania pomiędzy przedmiotami ułatwia zrozumienie i zastosowanie w działaniach praktycznych? | Program nauczania ułatwia uczenie się innych przedmiotów | Metoda – jakościowa  Techniki badania – wywiad grupowy z nauczycielami wspólnie realizującymi program kształcenia w zawodzie  Narzędzia – kwestionariusz wywiadu | wrzesień |
| Trafność doboru materiału nauczania, metod, środków dydaktycznych, form organizacyjnych ze względu na przyjęte cele | 1. Jaki jest stan wiedzy uczniów z treści bazowych dla przedmiotu przed rozpoczęciem wdrażania programu? 2. Czy cele nauczania zostały poprawnie sformułowane? 3. Czy cele nauczania odpowiadają opisanym treściom programowym? 4. Czy dobór metod nauczania pozwoli na osiągnięcie celu? 5. Czy zaproponowane metody umożliwiają realizację treści? 6. Czy dobór środków dydaktycznych pozwoli na osiągniecie celu? | Materiał nauczania, zastosowane metody i dobór środków dydaktycznych wspomaga przygotowanie ucznia do zdania egzaminu zawodowego | Metoda – jakościowa  Techniki badania – wywiad grupowy z nauczycielami wspólnie realizującymi program kształcenia w zawodzie  Narzędzia – kwestionariusz wywiadu | wrzesień |
| Trafność doboru materiału w kształtowaniu kompetencji personalnych i społecznych dla zawodu | 1. Czy program nauczania rozwija kompetencje personalne i społeczne? | Program KPS jest adekwatny do potrzeb i wyraża się w postawie i działaniu ucznia | Metoda – ilościowa  Techniki badania – ankieta  Narzędzia – kwestionariusz ankiety dla nauczyciela, ucznia i pracodawcy | czerwiec I i II rok nauki |
| Stopień trudności programu z pozycji ucznia | 1. Czy program nie jest przeładowany, trudny? 2. Czy jego realizacja nie powoduje negatywnych skutków ubocznych? | Program nauczania jest atrakcyjny dla ucznia i rozwija jego zainteresowania | Metoda – ilościowa  Techniki badania – ankieta  Narzędzia – kwestionariusz ankiety dla ucznia i nauczyciela | czerwiec II rok nauki |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Faza kształtująca** | | | | |
| **Przedmiot badania**  (*wynika z założonych w podstawie programowej i realizowanych w programie nauczania efektów kształcenia)* | **Pytania** **kluczowe**  *(Jakie należy zadać pytania, aby uzyskać informację, czy dany wskaźnik został osiągnięty)* | **Wskaźniki**  *Wynika z kryteriów weryfikacji* | **Zastosowane metody, techniki narzędzia** | **Termin badania** |
| I. Organizacja stanowiska pracy technika programisty uwzględniająca bezpieczeństwo i higienę pracy | 1. Czy uczeń przestrzega zasad ergonomii w organizacji miejsca pracy? 2. Czy uczeń w trakcie wykonywania zadań zawodowych stosuje przepisy bezpieczeństwa i higieny pracy oraz ochrony przeciwpożarowej i ochrony środowiska? | 1. Opisuje specyfikę stanowiska pracy zgodnie z zasadami ergonomii przy komputerze. 2. Rozróżnia dokumenty dotyczące przepisów bezpieczeństwa i higieny pracy oraz ochrony przeciwpożarowej i ochrony środowiska. | Metoda – jakościowa  Techniki badania – test | Po zakończonym dziale nauczania |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/zestawienia danych | Marzec/wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| II. Tworzenie, administrowanie i użytkowanie relacyjnych baz danych | 1. Czy uczeń rozróżnia systemy baz danych w architekturze klient – serwer? 2. Czy uczeń rozróżnia systemy baz danych bez podziału na klienta i serwer? 3. Czy uczeń tworzy strukturę bazy danych? 4. Czy uczeń wprowadza dane do bazy danych? 5. Czy uczeń stosuje zapytania w języku SQL do bazy danych? 6. Czy uczeń tworzy formularze do wprowadzania danych w bazie? 7. Czy uczeń tworzy raporty w bazie danych? 8. Czy uczeń zarządza bazą danych? | 1. Rozróżnia systemy baz danych w architekturze klient – serwer. 2. Rozróżnia systemy baz danych bez podziału na klienta i serwer. 3. Tworzy strukturę bazy danych. 4. Wprowadza dane do bazy danych. 5. Stosuje zapytania w języku SQL do bazy danych. 6. Tworzy formularze do wprowadzania danych w bazie? 7. Tworzy raporty w bazie danych. 8. Zarządza bazą danych. | Metoda – jakościowa  Techniki badania – obserwacja, próba pracy, test  Narzędzia – obserwacja, wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/zestawienia danych | Marzec/wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| III. Tworzenie i administrowanie stronami internetowymi, | 1. Czy uczeń rozróżnia techniki tworzenia stron internetowych? 2. Czy uczeń tworzy zawartość strony w oparciu o język znaczników HTML? 3. Czy uczeń tworzy prezentację wizualną strony w oparciu o kaskadowe arkusze stylów CSS? 4. Czy uczeń tworzy i przetwarza grafikę na potrzeby Internetu? 5. Czy uczeń testuje i optymalizuje strony internetowe? | 1. Rozróżnia techniki tworzenia stron internetowych. 2. Tworzy zawartość strony w oparciu o język znaczników HTML. 3. Tworzy prezentację wizualną strony w oparciu o kaskadowe arkusze stylów CSS. 4. Tworzy i przetwarza grafikę na potrzeby Internetu. 5. Testuje i optymalizuje strony internetowe. | Metoda – jakościowa  Techniki badania – obserwacja, próba pracy, test  Narzędzia – obserwacja, wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/zestawienia danych | Marzec/wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| IV. Tworzenie i administrowanie systemami zarządzania treścią | 1. Czy uczeń rozróżnia popularne systemy zarządzania treścią? 2. Czy uczeń dobiera system zarządzania treścią do określonego typu witryny np. blog, sklep, baza ogłoszeń, portal społecznościowy? 3. Czy uczeń tworzy witrynę internetową w oparciu o wybrany system zarządzania treścią? 4. Czy uczeń zarządza witryną wykonaną w systemie zarządzania treścią? | * + - 1. Rozróżnia popularne systemy zarządzania treścią?       2. Dobiera system zarządzania treścią do określonego typu witryny np. blog, sklep, baza ogłoszeń, portal społecznościowy?       3. Tworzy witrynę internetową w oparciu o wybrany system zarządzania treścią?       4. Zarządza witryną wykonaną w systemie zarządzania treścią. | Metoda – jakościowa  Techniki badania – obserwacja, próba pracy, test  Narzędzia – obserwacja, wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/ czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/ zestawienia danych | Marzec/ wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| V. Programowanie aplikacji internetowych | 1. Czy uczeń rozróżnia języki programowania aplikacji internetowych? 2. Czy uczeń stosuje bazy danych do tworzenia aplikacji internetowych. 3. Czy uczeń tworzy aplikacje internetowe z wykorzystaniem serwera? 4. Czy uczeń tworzy aplikacje internetowe w oparciu o język skryptowy JavaScript? 5. Czy uczeń korzysta z bibliotek i frameworków języka JavaScript, w tym z biblioteki jQuery, Angular, React? 6. Czy uczeń tworzy aplikacje internetowe w oparciu o jeden z wybranych języków: Python, ASP.NET, PHP, JSP? | 1. Rozróżnia języki programowania aplikacji internetowych. 2. Stosuje bazy danych do tworzenia aplikacji internetowych. 3. Tworzy aplikacje internetowe z wykorzystaniem serwera. 4. Tworzy aplikacje internetowe w oparciu o język skryptowy JavaScript. 5. Korzysta z bibliotek i frameworków języka JavaScript, w tym z biblioteki jQuery, Angular, React. 6. Tworzy aplikacje internetowe w oparciu o jeden z wybranych języków: Python, ASP.NET, PHP, JSP. | Metoda – jakościowa  Techniki badania – test  Narzędzia – wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/ czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/ zestawienia danych | Marzec/ wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| VI. Projektowanie, programowanie i testowanie zaawansowanych aplikacji webowych | 1. Czy uczeń rozróżnia wybrane języki programowania stosowane w zawansowanym programowaniu aplikacji webowych np. PHP, C#, Python, JavaScript? 2. Czy uczeń rozróżnia wybrane frameworki stosowane w zawansowanym programowaniu aplikacji webowych np. Django, Node.js, jQuery? 3. Czy uczeń potrafi zaprogramować zaawansowaną aplikację webową w wybranym języku programowania? 4. Czy uczeń potrafi zaprogramować zaawansowaną aplikację webową z wykorzystaniem frameworku jQuery? 5. Czy uczeń potrafi zaprogramować zaawansowaną aplikację webową w wykorzystaniem wybranego frameworku np., Node.js, Django? 6. Czy uczeń potrafi testować zaprogramowane aplikacje webowe? 7. Czy uczeń potrafi dokumentować zaprogramowane aplikacje webowe? | 1. Rozróżnia wybrane języki programowania stosowane w zawansowanym programowaniu aplikacji webowych np. PHP, C#, Python, JavaScript? 2. Rozróżnia wybrane frameworki stosowane w zawansowanym programowaniu aplikacji webowych np. Django, Node.js, jQuery? 3. Programuje zaawansowaną aplikację webową w wybranym języku programowania? 4. Programuje zaawansowaną aplikację webową z wykorzystaniem frameworku jQuery? 5. Programuje zaawansowaną aplikację webową w wykorzystaniem wybranego frameworku np., Node.js, Django? 6. Testuje zaprogramowane aplikacje webowe? 7. Dokumentuje zaprogramowane aplikacje webowe? | Metoda – jakościowa  Techniki badania – obserwacja, próba pracy, test  Narzędzia – obserwacja, wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/ czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/ zestawienia danych | marzec/wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| VII. Projektowanie, programowanie i testowanie aplikacji desktopowych | 1. Czy uczeń rozróżnia techniki programowania aplikacji desktopowych w tym dla trybu tekstowego i graficznego? 2. Czy uczeń potrafi zaprogramować aplikacje desktopowe uruchamiane w trybie tekstowym? 3. Czy uczeń potrafi zaprogramować aplikacje desktopowe uruchamiane w trybie graficznym? 4. Czy uczeń potrafi testować zaprogramowane aplikacje desktopowe? 5. Czy uczeń potrafi dokumentować zaprogramowane aplikacje desktopowe? | 1. Rozróżnia techniki programowania aplikacji desktopowych w tym dla trybu tekstowego i graficznego. 2. Programuje aplikacje desktopowe uruchamiane w trybie tekstowym. 3. Programuje aplikacje desktopowe uruchamiane w trybie graficznym. 4. Testuje zaprogramowane aplikacje desktopowe? 5. Dokumentuje zaprogramowane aplikacje desktopowe? | Metoda – jakościowa  Techniki badania – obserwacja, próba pracy, test  Narzędzia – obserwacja, wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/ czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/ zestawienia danych | Marzec/ wrzesień po ogłoszeniu wyników egzaminu zewnętrznego |
| VIII. Projektowanie, programowanie i testowanie aplikacji mobilnych | 1. Czy uczeń rozróżnia techniki programowania aplikacji mobilnych dla różnych środowisk iOS, Android? 2. Czy uczeń potrafi zaprogramować aplikację mobilną w wybranym środowisku iOS lub Android? 3. Czy uczeń potrafi przetestować zaprogramowaną aplikację w wybranym środowisku iOS lub Android? 4. Czy uczeń potrafi dokumentować zaprogramowane aplikacje mobilne? | * + - 1. Rozróżnia techniki programowania aplikacji mobilnych dla różnych środowisk iOS, Android.       2. Programuje aplikację mobilną w wybranym środowisku iOS lub Androin .       3. Testuje zaprogramowaną aplikację w wybranym środowisku iOS lub Androin.       4. Dokumentuje zaprogramowane aplikacje mobilne. | Metoda – jakościowa  Techniki badania – obserwacja, próba pracy, test  Narzędzia – obserwacja, wyniki testu | W trakcie całego okresu kształcenia, po zakończonym cyklu nauczania/ czerwiec |
| Metoda – ilościowa  Techniki badania – analiza statystyczna wyników egzaminu zewnętrznego/ zestawienia danych | Wrzesień po ogłoszeniu wyników egzaminu |
| IX. Przestrzeganie zasad etycznych w relacjach z klientem i we współpracy zespołowej, stosując techniki i narzędzia skutecznej komunikacji | 1. Czy uczeń w relacjach z innymi zachowuje zasady uczciwości i lojalności, sumiennie i rzetelnie wykonuje obowiązki zawodowe? 2. Czy uczeń potrafi rozwiązywać dylematy etyczne? 3. Czy uczeń traktuje innych z szacunkiem? 4. Czy uczeń stosuje przepisy o ochronie danych osobowych? | 1. Stosuje zasady uczciwości, sumienności, rzetelności i lojalności w wykonywaniu obowiązków zawodowych. 2. Przestrzega zasad etycznych i ogólnospołecznych w realizacji zadań zawodowych. 3. Rozwiązuje dylematy etyczne pojawiające się w realizacji zadań zawodowych. 4. Wyraża szacunek wobec klientów, współpracowników, pracodawcy. 5. Przestrzega zasad etycznych i prawnych, związanych z ochroną własności intelektualnej i ochroną danych osobowych. | Metoda – jakościowa  Techniki badania – obserwacja, test, ankieta do pracodawcy  Narzędzia – obserwacja, wyniki testu i ankiety | Po zakończonym cyklu nauczania i praktykach zawodowych/ czerwiec |
| X. Planowanie zadania i przewidywanie konsekwencji swoich działań, przy jednoczesnym wykazywaniu odpowiedzialności i kreatywności. | 1. Czy uczeń wykonuje zadania zawodowe zgodnie z normami prawnymi i etycznymi, znając konsekwencje braku odpowiedzialności? | 1. Objaśnia, czym jest odpowiedzialność w życiu osobistym oraz w deontologii. 2. Ocenia przypadki naruszania norm i procedur postępowania. 3. Wskazuje obszary odpowiedzialności prawnej za podejmowane działania. 4. Określa konsekwencje braku odpowiedzialności. | Metoda – jakościowa  Techniki badania – obserwacja, test  Narzędzia – obserwacja, wyniki testu | Po zakończonym cyklu nauczania/ czerwiec |
| **Faza podsumowująca**  *Dokonujemy pomiaru osiągnięć uczniów, analizy końcowych efektów realizacji programu, oceniamy program jako całość, ewentualnie porównujemy z innymi programami i nanosimy określone zmiany w naszym programie.* | | | | |
| **Przedmiot badania** | **Pytania kluczowe** | **Wskaźniki** | **Zastosowane metody, techniki narzędzia** | **Termin badania** |
| Sprawność szkoły | 1. Liczba poprawek 2. Liczba ocen niedostatecznych końcoworocznych 3. Ilu uczniów nie otrzymało promocji do kolejnej klasy? | 70% uczniów zapisanych w pierwszej klasie ukończyło szkołę | Metoda – jakościowa Technika – analiza dokumentów  Narzędzia – dyspozycja do analizy dokumentów | czerwiec  IV rok nauki |
| Skuteczność współpracy szkoły  z przedsiębiorcami/ pracodawcami | 1. Czy pracownie technodydaktyczne zapewniają przygotowanie do wykonywania czynności zawodowych? | 70% uczniów zrealizowało praktyki zawodowe z oceną bardzo dobrą | Metoda – ilościowa  Technika – ankieta  Narzędzia – kwestionariusz ankiety dla pracodawcy | czerwiec  III rok nauki |
| Wyniki egzaminów potwierdzających kwalifikacje w zawodzie | 1. Ilu uczniów zapisano w pierwszej klasie? 2. Ilu uczniów przystąpiło do egzaminów potwierdzających kwalifikacje w zawodzie? 3. Ilu uczniów uzyskało minimalną liczbę punktów z egzaminu? | 70% uczniów przystępujących do egzaminu uzyskało świadectwo/ dyplom potwierdzający kwalifikacje w zawodzie | Metoda – ilościowa  Technika – analiza statystyczna wyników egzaminu OKE/zestawienia danych | wrzesień  po ogłoszeniu wyników egzaminu |
| Ocena funkcjonowania przyjętego w szkole programu nauczania | 1. Czy sposób przekazywania treści jest zrozumiały? 2. Czy dostosowane jest tempo realizacji materiału do możliwości ucznia? 3. Czy istnieje możliwość organizacji dodatkowych ćwiczeń? 4. Czy program pozwala na kształtowanie umiejętności ponadprzedmiotowych? 5. Czy rozwijany jest system pracy zespołowej? 6. Czy uczeń ma możliwość samodzielnego działania w planowaniu zadań? 7. Czy są sprecyzowane zasady oceniania i wymagań przedmiotowych? | Optymalizacja procesu edukacyjnego i podniesienie jakości kształcenia | Metoda – jakościowa  Technika – ankieta  Narzędzia – kwestionariusz ankiety dla nauczycieli i uczniów | czerwiec – I rok nauki  styczeń – II rok nauki |

# ZALECANA LITERATURA DO ZAWODU

1. Krzysztof Szczęch, Wanda Bukała, Bezpieczeństwo i higiena pracy, Podręcznik do kształcenia zawodowego. WSiP. Warszawa 2016.
2. Marcin Czerwonka, Zenon Nowocień Kwalifikacja INF.02. Administracja i eksploatacja systemów komputerowych, urządzeń peryferyjnych i lokalnych sieci komputerowych. Część 1. Systemy komputerowe. Podręcznik do nauki zawodu technik informatyk, wyd. Helion,
3. Jolanta Pokorska, Kwalifikacja INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych. Część 2. Projektowanie i administrowanie bazami danych. Podręcznik do nauki zawodu technik informatyk i technik programista, wyd. Helion,
4. Jolanta Pokorska, Podręcznik do zawodu technik informatyk, technik programista, część 1, Tworzenie stron internetowych, Kwalifikacja INF.03.Programowanie, tworzenie i administrowanie stronami internetowymi i bazami danych, wyd. Helion –podręcznik w przygotowaniu.
5. Adam Freeman, HTML5. Przewodnik encyklopedyczny, wyd. Helion
6. David Sawayer McFarland, CSS3 nieoficjalny podręcznik, wyd. Helion
7. David Sawayer McFarland, JavaScript i jQuery. Nieoficjalny podręcznik, wyd. Helion,
8. Eric T. Freeman, Elisabeth Robson, Programowanie w Javascript. Rusz głową!, wyd.Helion
9. Luke Welling, Laura Thomson, PHP i MySQL. Tworzenie stron WWW. Vademecum profesjonalisty, wyd. Helion
10. R. Sama, K. Sama, Język angielski zawodowy w branży informatycznej, wyd. WSiP, Warszawa 2016
11. Bjarne Stroustrup, Język C++. Kompendium wiedzy, wyd. Helion,
12. Stephen Prata, Język C++. Szkoła programowania. Wydanie VI, wyd. Helion,
13. Grębosz Jerzy, Opus magnum C++11, Programowanie w języku C++ (komplet), Wyd.: Grębosz Jerzy,
14. Python dla każdego. Podstawy programowania – Michael Dawson,wyd. Helion,
15. Eric Matthes, Python. Instrukcje dla programisty, wyd. Helion,
16. Al Sweigart, Automatyzacja nudnych zadań z Pythonem, wyd. Helion,
17. Dawn Griffiths, David Griffiths, Android. Programowanie aplikacji. Rusz głową! Wydanie II, wyd. Helion,
18. Marcin Płonkowski, Android Studio. Tworzenie aplikacji mobilnych. wyd. Helion,
19. Matt Neuburg, iOS 12. Wprowadzenie do programowania w Swifcie. Wydanie V, wyd. Helion,
20. Steven F. Daniel, Xamarin. Tworzenie interfejsów użytkownika, wyd. Helion.
21. Gniewomir Sarbicki, Python. Kurs dla nauczycieli i studentów, wyd. Helion,
22. Adriaan de Jonge, Phillip Dutson, jQuery, jQuery UI oraz jQuery Mobile. Receptury, wyd. Helion,
23. Jon Duckett, JavaScript i jQuery. Interaktywne strony WWW dla każdego. Podręcznik Front-End Developera wyd. Helion,
24. David Herron, Platforma Node.js. Przewodnik webdevelopera. Wydanie III, wyd. Helion,
25. Antonio Mele, Django 2. Praktyczne tworzenie aplikacji sieciowych. Wydanie II, wyd. Helion.
26. http://codecademy.com
27. http://w3schools.com
28. http://www.cplusplus.com/
29. https://docs.python.org/3/tutorial/
30. https://doc.qt.io/qtcreator/index.html
31. http://www-cs.ccny.cuny.edu/~wolberg/cs221/qt/books/C++-GUI-Programming-with-Qt-4-1st-ed.pdf
32. https://qmlbook.github.io/assets/qt5\_cadaques.pdf