PROGRAM NAUCZANIA ZAWODU nr 27/2019/Tp

TECHNIK PROGRAMISTA, 351406

opracowany 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

KWALIFIKACJE WYODRĘBNIONE W ZAWODZIE:

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

INF.04. Projektowanie, programowanie i testowanie aplikacji

Zespół Szkół im. Władysława Szybińskiego w Cieszynie
ul. Kraszewskiego 11, 43-400 Cieszyn

Technikum nr 1, 5-letnie

Cieszyn 2019

SPIS TREŚCI

PLAN NAUCZANIA ZAWODU:

I. PLAN NAUCZANIA

Nazwa i symbol cyfrowy zawodu: technik programista 351406								
Nazwa i symbol kwalifikacji: Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych **INF.03.**								
Nazwa i symbol kwalifikacji: Projektowanie, programowanie i testowanie aplikacji **INF.04.**								
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: INF.03. Tworzenie i administrowanie stronami i aplikacjami internetowymi oraz bazami danych							
1.	Język angielski zawodowy		1				30	**T**
2.	Podstawy informatyki	1	1				60	**T**
3.	Witryny internetowe	3	3				180	**T**
4.	Bazy danych	1	2				90	**T**
5.	Pracownia programowania aplikacji internetowych	3	3				180	**P**
6.	Pracownia baz danych	2	2				120	**P**
7.	Pracownia multimediów i grafiki komputerowej	1					30	**P**
	Razem liczba godzin w kwalifikacji **INF.03.** :	11	12	0	0	0	790	
	Kwalifikacja: INF.04. Projektowanie, programowanie i testowanie aplikacji							
1.	Język angielski zawodowy			1	1		30	**T**
2.	Projektowanie oprogramowania		1	2			90	**T**

3.	Programowanie obiektowe			3	2		150	**T**
4.	Pracownia programowania aplikacji desktopowych			2	3		150	**P**
5.	Pracownia programowania aplikacji mobilnych			2	4		180	**P**
6.	Pracownia programowania zaawansowanych aplikacji webowych			2	3		150	**P**
1.	**Zajęcia specjalizujące**					7	210	**P**
	Razem liczba godzin w kwalifikacji **INF.03.** :	0	1	12	13	7	900	
	Razem liczba godzin kształcenia w zawodzie:	11	13	12	13	7	1690	
	Praktyka zawodowa							Klasa III i IV
	Egzamin zawodowy w zakresie kwalifikacji **INF.03** odbywa się **pod koniec klasy 2** Egzamin zawodowy w zakresie kwalifikacji **INF.04** odbywa się **pod koniec klasy 4**							

***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***

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 organizacji pracy małych zespołów*

WSTĘP DO PROGRAMU

Opis zawodu

Programista to osoba, która tworzy programy komputerowe w pewnym języku programowania oraz nadzoruje proces ich wdrażania. Zazwyczaj programiści znają 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

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

Wykaz przedmiotów z podziałem na kwalifikacje oraz na przedmioty teoretyczne i zajęcia organizowane w formie zajęć praktycznych oraz praktyka zawodowa:

Lp. Nazwa przedmiotu

Nazwa i symbol kwalifikacji:

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

Przedmioty w kształceniu zawodowym teoretycznym

1. Język angielski zawodowy
2. Podstawy informatyki
3. Witryny internetowe
4. Bazy danych

Przedmioty w kształceniu zawodowym organizowane w formie zajęć praktycznych

1. Pracownia programowania aplikacji internetowych
2. Pracownia baz danych
3. Pracownia multimediów i grafiki komputerowej

Nazwa i symbol kwalifikacji:

INF.04. Projektowanie, programowani i testowanie aplikacji

Przedmioty w kształceniu zawodowym teoretycznym

1. Język angielski zawodowy
2. Projektowanie oprogramowania
3. Programowanie obiektowe

Przedmioty w kształceniu zawodowym organizowane w formie zajęć praktycznych

1. Pracownia programowania aplikacji desktopowych
2. Pracownia programowania aplikacji mobilnych
3. Pracownia programowania zaawansowanych aplikacji webowych
4. Zajęcia specjalizujące

Praktyka zawodowa

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.

CELE KIERUNKOWE ZAWODU

Absolwent szkoły prowadzącej kształcenie w zawodzie technik programista 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

1. Język angielski zawodowy

Cele ogólne przedmiotu

1. Poznanie podstawowego zasobu środków językowych umożliwiających realizację zadań zawodowych;
2. Nabycie umiejętności rozumienia prostych wypowiedzi ustnych w zakresie umożliwiającym realizację zadań zawodowych;
3. Poznanie zasad samodzielnego tworzenia krótkich, prostych, spójnych i logicznych wypowiedzi ustnych i pisemnych w zakresie umożliwiającym realizację zadań zawodowych;
4. Nabycie umiejętności uczestnictwa w rozmowie w typowych sytuacjach związanych realizacją zadań zawodowych;
5. Poznanie zasad parafrazy w typowych sytuacjach związanych realizacją zadań zawodowych.
6. Poznanie strategii służących doskonaleniu własnych umiejętności językowych oraz podnoszące świadomość językową.

Cele operacyjne

Uczeń potrafi:

1) rozpoznawać środki językowe umożliwiające realizację czynności zawodowych;
2) zastosować środki językowe umożliwiające realizację czynności zawodowych;
3) określać główne myśli wypowiedzi lub tekstu lub fragmentu wypowiedzi lub tekstu;
4) znajdować w wypowiedzi lub tekście określone informacje;
5) rozpoznać związki między poszczególnymi częściami tekstu;
6) przedstawiać sposób postępowania w różnych sytuacjach zawodowych (np. udziela instrukcji, wskazówek, określa zasady);
7) zastosować formalny lub nieformalny styl wypowiedzi adekwatnie do sytuacji;
8) wyrazić swoje opinie i uzasadnia je, pyta o opinie, zgadza się lub nie zgadza z opiniami innych osób;
9) prowadzić proste negocjacje związane z czynnościami zawodowymi;
10) przekazać w języku obcym nowożytnym informacje sformułowane w języku polskim lub w tym języku obcym nowożytnym oraz odwrotnie;
11) wykorzystać kontekst (tam, gdzie to możliwe), aby w przybliżeniu określić znaczenie słowa.

MATERIAŁ NAUCZANIA

Dział programowy	Tematy jednostek metodycznych	Liczba godz.	Wymagania programowe		Uwagi o realizacji
			Podstawowe Uczeń potrafi:	Ponadpodstawowe Uczeń potrafi:	Etap realizacji
I. Komunikacja w języku obcym nowożytnym	1. Proste wypowiedzi ustne i pisemne	5	– rozpoznawać oraz stosuje środki językowe umożliwiające realizację czynności zawodowych w zakresie – określać główną myśl wypowiedzi lub tekstu lub fragmentu wypowiedzi lub tekstu – znajdować w wypowiedzi lub tekście określone informacje	– rozpoznawać związki między poszczególnymi częściami tekstu – układać informacje w określonym porządku	Klasa II / Klasa III
	2. Samodzielne wypowiedzi ustne i pisemne	20	– opisywać przedmioty, działania i zjawiska związane z czynnościami zawodowymi – wyrażać i uzasadnia swoje stanowisko – stosować zasady konstruowania tekstów o różnym charakterze – stosować formalny lub nieformalny styl wypowiedzi adekwatnie do sytuacji – rozpoczynać, prowadzić i kończyć rozmowę – uzyskiwać i przekazywać informacje i wyjaśnienia – stosować zwroty i formy grzecznościowe – dostosowywać styl wypowiedzi do sytuacji	– przedstawiać sposób postępowania w różnych sytuacjach zawodowych (np. udziela instrukcji, wskazówek, określa zasady) – wyrażać swoje opinie i uzasadnia je, pyta o opinie, zgadza się lub nie zgadza z opiniami innych osób – prowadzić proste negocjacje związane z czynnościami zawodowymi	Klasa II / Klasa III

II. Samodzielność językowa	1. Tłumaczenia i parafrazy	30	– przekazywać w języku obcym nowożytnym informacje zawarte w materiałach wizualnych (np. wykresach, symbolach, piktogramach, schematach) oraz audiowizualnych (np. filmach instruktażowych) – przekazywać w języku polskim informacje sformułowane w języku obcym nowożytnym – przekazywać w języku obcym nowożytnym informacje sformułowane w języku polskim lub w tym języku obcym nowożytnym – przedstawiać publicznie w języku obcym nowożytnym wcześniej opracowany materiał, np. prezentację	– wyjaśnić, czym jest norma przekazywać w języku obcym nowożytnym informacje zawarte w materiałach wizualnych (np. wykresach, symbolach, piktogramach, schematach) oraz audiowizualnych (np. filmach instruktażowych) – przekazywać w języku polskim informacje sformułowane w języku obcym nowożytnym – przekazywać w języku obcym nowożytnym informacje sformułowane w języku polskim lub w tym języku obcym nowożytnym – przedstawiać publicznie w języku obcym nowożytnym wcześniej opracowany materiał, np. prezentację	Klasa II / Klasa III
	2. Samodoskonalenie językowe	5	– korzysta ze słownika dwujęzycznego i jednojęzycznego – współdziała z innymi osobami, realizując zadania językowe – korzysta z tekstów w języku obcym nowożytnym, również za pomocą technologii informacyjno-komunikacyjnych – identyfikuje słowa klucze i internacjonalizmy – upraszcza (jeżeli to konieczne) wypowiedź,	– wykorzystuje kontekst (tam, gdzie to możliwe), aby w przybliżeniu określić znaczenie słowa	Klasa II / Klasa III

			zastępuje nieznane słowa innymi, wykorzystuje opis, środki niewerbalne		

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Zajęcia edukacyjne mogą być prowadzone w pracowni lokalnych sieci komputerowych. W pracowni w której prowadzone będą zajęcia edukacyjne powinny się znajdować: narzędzia i urządzenia związane z typowymi czynnościami zawodowymi. Komputer z dostępem do Internetu (1 stanowisko dla dwóch uczniów). Urządzenia multimedialne. Autorzy programu proponują stosowanie aktywizujących metod kształcenia, ze szczególnym uwzględnieniem metody ćwiczeń, dyskusji dydaktycznej. Zajęcia powinny być prowadzone w grupach do 15 osób. Dominującą formą organizacyjną pracy uczniów jest praca indywidualna i w grupach dwuosobowych.

Przykładowe zadania:

Zadaniem grupy jest wykonanie pracy zgodnie z opisem.
- Wykonywanie ćwiczeń gramatycznych.
Uczniowie wykonują ćwiczenia, po ich wykonaniu dokonują samooceny.
- Wykonywanie ćwiczeń weryfikujących rozumienie tekstu ze słuchu.
Uczniowie wykonują ćwiczenia, po ich wykonaniu dokonują samooceny.
- Wydawanie poleceń w języku obcym, dotyczących wykonywania zadań zawodowych.
Uczniowie wykonują ćwiczenia, po ich wykonaniu dokonują samooceny.
- Sporządzanie notatki z tekstu słuchanego i czytanego.
Uczniowie wykonują ćwiczenia, po ich wykonaniu dokonują samooceny.
- Wysyłanie i odbieranie informacji w języku angielskim, pocztą elektroniczną.
Uczniowie wykonują ćwiczenia, po ich wykonaniu dokonują samooceny.
- Tłumaczenie tekstów zawodowych z języka polskiego na język angielski.
Uczniowie wykonują ćwiczenia, po ich wykonaniu dokonują samooceny

PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA

Sprawdzanie efektów kształcenia może być przeprowadzone na podstawie prezentacji. W ocenie należy uwzględnić następujące kryteria ogólne: zawartość merytoryczną prezentacji, sposób prezentacji (układ, czytelność, poprawność gramatyczna), opracowanie pisemne prezentacji.

PROPONOWANE METODY EWALUACJI PRZEDMIOTU

Strategia przeprowadzanej ewaluacji będzie polegała analizie danych (oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów oraz zadań i testów praktycznych). Dodatkowo ważnym elementem strategii jest opracowanie ankiety zwrotnej dla uczniów i rodziców na temat realizowanych treści. Dane zostaną poddane analizie ilościowej i jakościowej. Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych, przypisanych do danego działu programowego. Również w trakcie realizacji procesu kształcenia, ze względu na szybkość zmian w branży, ewaluacji będzie podlegać również przekazywany materiał.

2. Podstawy informatyki

Cele ogólne przedmiotu

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	- określić zakres odpowiedzialności pracodawcy i pracownika, - podać przykłady regulacji w opracowywaniu regulaminów, układów zbiorowych pracy w części dotyczącej warunków pracy instrukcji obsługi,	Klasa I

			higieny pracy, - scharakteryzować prawa i obowiązki pracodawcy w zakresie bezpieczeństwa i higieny pracy.		
	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 II
	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 II
	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 II

	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 II
	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 II

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Warunkiem osiągnięcia założonych celów kształcenia w zakresie przedmiotu 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,

3. Bazy danych

Cele ogólne przedmiotu

1. Poznanie pojęć dotyczących baz danych;
2. Poznanie zasad projektowania baz danych;
3. Poznanie zasad administrowania bazami danych;
4. Poznanie strukturalnego języka zapytań SQL;

Cele operacyjne

Uczeń potrafi:

1) określić podstawowe pojęcie związane z bazami danych;
2) określić typy danych;
3) rozpoznać postacie normalne baz danych;
4) rozpoznać środki ochrony indywidualnej i zbiorowej;
5) opisać cech baz danych;
6) tworzyć diagramy E/R (Entity-Relationship Diagram);
7) korzystać z systemów zarządzania bazami danych SZBD (Database Management System);
8) stosować strukturalny język zapytań SQL (Structured Query Language);
9) zarządzać systemem 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. Teoretyczne aspekty baz danych	1. Pojęcia dotyczące baz danych	15	– określać pojęcia związane z bazami danych: encja, związki encji, atrybuty encji, klucz relacji – określać typy danych używanych w bazach danych	– rozpoznawać postacie normalne baz danych – opisywać cechy relacyjnej bazy danych	Klasa III

	2. Diagramy E/R	15	– rozróżniać bloki składowe diagramów E/R	– charakteryzować typy notacji diagramów E/R – analizować diagramy E/R	Klasa III
II. Systemy bazodanowe	1. Obsługa i zarządzanie SZBDe	15	– rozróżniać dostępne SZBD – określać uprawnienia dla użytkowników	– rozróżniać dostępne SZBD – określać uprawnienia dla użytkowników	Klasa IV
	2. Wstęp do języka SQL	45	– opisywać polecenia języka SQL	– opisywać polecenia języka SQL	Klasa IV

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Zajęcia edukacyjne mogą być prowadzone w pracowni komputerowej z podziałem na grupy. W sali lekcyjnej, w której prowadzone będą zajęcia edukacyjne, powinny się znajdować komputer z dostępem do Internetu oraz urządzenia multimedialne jak również:

- komputer / komputery z dostępem do Internetu i zainstalowanym system operacyjnym pozwalającym na instalację oprogramowania do realizacji zapytań w języku SQL.
- oprogramowanie umożlwiające realizację zapytań w języku SQL.

Przedmiot Bazy danych jest wstępem teoretycznym dla przedmiotu praktycznego z tematyki baz danych. Dlatego nauczyciel, dobierając metodę kształcenia, powinien przede wszystkim przygotować uczniów do wykonywania zadań praktycznych na przedmiocie. W tym celu powinien odpowiedzieć sobie na pytania:

- jakie efekty chce osiągnąć?
- w jaki sposób osiągnięte efekty zostaną wykorzystane podczas zajęć praktycznych?
- jakie metody będą najbardziej odpowiednie dla uczniów?
- jak motywować uczniów i zapewnić ich zaangażowanie?

Autorzy programu proponują stosowanie aktywizujących metod kształcenia, ze szczególnym uwzględnieniem metody ćwiczeń, dyskusji dydaktycznej.

Przykładowe zadania:

- Test wiedzy ze znajomości pojęć z zakresu baz danych SZBD.
- Test wiedzy ze znajomości poleceń SQL.

PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA

Do oceny osiągnięć edukacyjnych uczniów, proponuje się stosowanie obserwacji pracy ucznia podczas wykonywania ćwiczeń oraz sprawdziany. Sprawdzenie osiągnięcia przez ucznia założonych szczegółowych celów kształcenia, będzie możliwe poprzez zastosowanie narzędzi pomiaru dydaktycznego oraz obserwację ucznia podczas wykonywania przez niego ćwiczeń.

PROPONOWANE METODY EWALUACJI PRZEDMIOTU

Strategia przeprowadzanej ewaluacji będzie polegała analizie danych (oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów oraz zadań i testów praktycznych) z przedmiotu Bazy danych oraz Pracownia baz danych, które są ściśle ze sobą skorelowane. Dodatkowo ważnym elementem strategii jest opracowanie ankiety zwrotnej dla uczniów i rodziców na temat realizowanych treści. Dane zostaną poddane analizie ilościowej i jakościowej. Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych, przypisanych do danego działu programowego. Również w trakcie realizacji procesu kształcenia, ze względu na szybkość zmian w branży, ewaluacji będzie podlegać również przekazywany materiał.

4. Witryny internetowe

Cele ogólne przedmiotu

1. Rozpoznaje znaczniki języka HTML;
2. Określa strukturę CSS na stronach internetowych.
3. Poznanie pojęć z zakresu multimediów;
4. Zastosowanie CSS na stronach internetowych.

Cele operacyjne

Uczeń potrafi:

1) wymienić udogodnienia dla osób z niepełnosprawnościami,
2) rozróżnić i dobrać odpowiedni hosting dla potrzeb użytkownika,
3) wymienić podstawowe znaczniki HTML;
4) określić składnię podstawowych znaczników HTML;
5) omówić style zewnętrzne i wewnętrzne;
6) zdefiniować style,
7) opisać systemy CMS;
8) dobrać systemy CMS.

MATERIAŁ NAUCZANIA

Dział programowy	Tematy jednostek metodycznych	Liczba godz.	Wymagania programowe		Uwagi o realizacji
			Podstawowe Uczeń potrafi:	Ponadpodstawowe Uczeń potrafi:	Etap realizacji
I. Język znaczników	1. Język HTML	10	– wymienić dostępne udogodnienia dla osób z niepełnosprawnościami, – wymienić standardy dokumentów hipertekstowych – wymienić narzędzia ułatwiające tworzenie kodu HTML – opisać znaczniki języka HTML – wymienić podstawowe znaczniki HTML (np. <html>, <head> <body>, <p>, ,<img>, <table> itp.) – opisać składnię podstawowych znaczników HTML	– opisać potrzeby użytkowników z różnymi niepełnosprawnościami przy projektowaniu stron internetowych, np. kontrast, powiększenie, inne elementy wspomagające niepełnosprawnych – opisać zasady i znaczenie wytycznych dotyczących ułatwień w dostępie do treści publikowanych w internecie, – wymienić wymagania dotyczące poziomu dostępności według wytycznych WCAG 2.0 – omówić style lokalne, wewnętrzne i zewnętrzne, – określić proces walidacji strony internetowej – określić proces pozycjonowania strony internetowej – scharakteryzować pakiety hostingowe – dobrać pakiety hostingowe uwzględniając potrzeby: o użytkownika, o wsparcie dla	Klasa II

				hostingu, ○ cenę, – możliwość instalacji systemów cms itp.	
	2. Arkusze stylów	20	– omówić budowę pliku css, – podać przykłady selektorów – podać przykłady deklaracji	– rozróżnić selektory elementów, atrybutów, specjalne, pseudoklas i pseudoelementów	Klasa II
	3. Systemy CSS	20	– określić funkcje systemów zarządzania treścią – wymienić przykłady systemów zarządzania treścią – scharakteryzować selektory elementów, atrybutów, specjalne, pseudoklas i pseudoelementów, – wymienić przykładowe selektory CSS	– zanalizować systemy zarządzania treścią pod względem funkcjonalności – omówić przykładowe funkcje panelu administratora w systemach zarządzania treścią – omówić znaczenie selektorów CSS	Klasa II / Klasa III
	4. Systemy zarządzania treścią	15	– określić funkcje systemów zarządzania treścią – wymienić przykładowe systemy zarządzania treścią, – scharakteryzować przykładowe systemy zarządzania treścią – dobrać przykładowe systemy zarządzania treścią pod kątem potrzeb oraz możliwości oprogramowania	– określić funkcje panelu administratora w systemach zarządzania treścią	Klasa III
II. Podstawy stron internetowych	1. Język HTML	15	– skorzystać ze standardów dokumentów hipertekstowych, – stworzyć dokument na	– zdefiniować hierarchię treści stosując znaczniki nagłówków i paragrafu – wykonać formularze na	Klasa II / Klasa III

			podstawie szablonu – zdefiniować strukturę dokumentu hipertekstowego korzystając ze znaczników sekcji – zdefiniować elementy strony internetowej: o listy, o tabele, o obrazy, o odnośniki, – kontrolki,	stronie internetowej – zaprojektować szablon strony internetowej	
	2. Arkusze stylów	20	– zastosować style lokalne, wewnętrzne i zewnętrzne, – zastosować kaskadowość stylów – zastosować selektory elementów, atrybutów, specjalne, pseudoklas i pseudoelementów, – rozpoznać selektory CSS	– rozróżnić selektory elementów, atrybutów, specjalne, pseudoklas i pseudoelementów – zastosować selektory CSS, ich własności i wartości – zaprojektować wygląd strony internetowej przy wykorzystaniu języka CSS – wykonać responsywne strony internetowe z wykorzystaniem CSS	Klasa II / Klasa III
11. Projektowanie i testowanie stron internetowych	1. Projektowanie stron internetowych	20	– zanalizować projekt strony internetowej pod kątem potrzebnych plików graficznych, multimedialnych oraz narzędzi – przygotować strukturę strony internetowej zgodnie z projektem – stworzyć stronę zgodną z wytycznymi dotyczącymi ułatwień w dostępie do treści publikowanych w internecie	– wykonać projekt układ sekcji na stronie internetowej – dobrać paletę barw dla strony internetowej – dobrać czcionki dla strony internetowej – uwzględnić potrzeby użytkowników z różnymi niepełnosprawnościami przy projektowaniu stron internetowych, np. kontrast, powiększenie, inne elementy wspomagające	Klasa III / Klasa IV

				niepełnosprawnych	
	2. Testowanie i walidacja stron internetowych	20	– wykonać test strony internetowej w różnych przeglądarkach – wykonać test responsywności strony internetowej	– wykonać walidację strony internetowej – dobrać narzędzia walidacji strony internetowej – wykonać walidację strony internetowej – zoptymalizować stronę internetową – zastosować zasady dostępności (WCAG) i pozycjonowania strony internetowej	Klasa III / Klasa IV
	3. Publikowanie stron internetowych	20	– dobrać program do przesyłania danych na serwer – przesłać stronę interetową na serwer – publikować witryny internetowe zweryfikować poprawność publikowanych stron www	– dobrać pakiety serwerowe www	Klasa IV
IV. Aplikacje internetowe	1. Stosuje systemy CMS	20	– dobierać systemy zarządzania treścią – skonfigurować systemy zarządzania treścią – administrować systemem zarządzania treścią – wyszukiwać szablony dla systemów zarządzania treścią – zastosować szablony dla systemów zarządzania treścią – skonfigurować szablony dla systemów zarządzania treścią	– przygotować do instalacji system zarządzania treścią – zainstalować systemy zarządzania treścią – określić funkcje panelu administratora w systemach zarządzania treścią – projektować strony internetowe przy wykorzystaniu systemów zarządzania treścią	Klasa IV

			– instalować gotowe szablony dla systemów zarządzania treścią, – konfigurować gotowe szablony dla systemów zarządzania treści – zaktualizować systemy zarządzania treści – zaimportować materiały multimedialne do systemów zarządzania treści		

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Zajęcia edukacyjne mogą być prowadzone w sali lekcyjnej bez podziału na grupy. Możliwa jest również realizacja tematyki w pracowni komputerowej z podziałem na grupy. W sali lekcyjnej, w której prowadzone będą zajęcia edukacyjne, powinny się znajdować komputer z dostępem do Internetu oraz urządzenia multimedialne jak również z dostępem do narzędzi:

- pakiety oprogramowania zawierające serwer WWW, SQL, PHP,
- serwer hostingowy do testowania projektów webowych.

Na zajęciach zalecamy sortować aktywizujące metody kształcenia, ze szczególnym uwzględnieniem metody ćwiczeń, dyskusji dydaktycznej oraz projektów. W zakresie organizacji pracy należy stosować instrukcje do zadań, podawanie dodatkowych zaleceń, instrukcji do pracy indywidualnej.

Zadania realizowane w grupach należy szczególnie monitorować, 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.

Przykładowy test:

1. Dołączenie zewnętrznego arkusza stylów do kodu HTML jest realizowane przy użyciu znacznika
   A. <css>
   B. <link>
   C. <style>
   D. <meta>

2. Które z zadań programistycznych powinno być wykonane po stronie serwera?
   A. Zmiana stylu HTML na stronie wywołana przesunięciem kursora.
   B. Zapisanie danych pobranych z aplikacji internetowej w bazie danych.
   C. Ukrywanie i pokazywanie elementów strony w zależności od aktualnego stanu kursora.
   D. Sprawdzanie danych wpisywanych do pola tekstowego w czasie rzeczywistym.

PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA

Zaleca się, aby osiągnięcia uczniów sprawdzane były systematycznie według kryteriów oceniania kształtującego podanych na początku zajęć. W procesie oceniania osiągnięć edukacyjnych uczniów należy uwzględnić wyniki wszystkich metod sprawdzania efektów kształcenia zastosowanych przez nauczyciela oraz ocenę za wykonane ćwiczenia. Oceniając osiągnięcia uczniów, należy zwrócić uwagę na kreatywność i innowacyjność podawanych rozwiązań.

Osiągnięcia uczniów należy oceniać na podstawie: prezentowanych umiejętności przy wykonywaniu zadań praktycznych, ukierunkowanej obserwacji indywidualnej i zespołowej pracy ucznia podczas wykonywania ćwiczeń.

PROPONOWANE METODY EWALUACJI PRZEDMIOTU

Strategia przeprowadzanej ewaluacji będzie polegała analizie danych (oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów oraz zadań i testów praktycznych) z przedmiotu **Witryny internetowe** oraz **Pracownia programowania aplikacji internetowych**, które są ściśle ze sobą skorelowane. Dodatkowo ważnym elementem strategii jest opracowanie ankiety zwrotnej dla uczniów i rodziców na temat realizowanych treści. Dane zostaną poddane analizie ilościowej i jakościowej.

Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych, przypisanych do danego działu programowego.

Również w trakcie realizacji procesu kształcenia, ze względu na szybkość zmian w branży, ewaluacji będzie podlegać również przekazywany materiał.

5. Pracownia baz danych

Cele ogólne przedmiotu

1. Poznanie zasad tworzenia diagramów E/R;
2. Nabycie umiejętności projektowania baz danych;
3. Nabycie umiejętności administrowania bazami danych;
4. Nabycie umiejętności stosowania strukturalnego języka zapytań SQL;

Cele operacyjne

Uczeń potrafi:

1) zdefiniować składowe diagramów E/R,
2) dobrać elementy diagramów E/R;
3) zastosować polecenia języka SQL do tworzenia baz danych,
4) zastosować polecenia języka SQL do modyfikowania baz danych,
5) zastosować polecenia języka SQL do tworzenia zapytań w bazach danych,
6) zastosować polecenia języka SQL do tworzenia skryptów,
7) zastosować polecenia języka SQL do zarządzania bazami danych,
8) zastosować polecenia języka SQL do naprawy baz danych,
9) zastosować narzędzia typu phpMyAdmin do realizacji punktów 1) - 8).

MATERIAŁ NAUCZANIA

Dział programowy	**Tematy jednostek metodycznych**	**Liczba godz.**	**Wymagania programowe**		**Uwagi o realizacji**
			Podstawowe Uczeń potrafi:	Ponadpodstawowe Uczeń potrafi:	Etap realizacji
I. Wstęp do tworzenia baz danych.	1. Narzędzia bazodanowe	5	– stosować odpowiednie typy danych przy zdefiniowaniu encji	– dobierać SZBD do określonego zastosowania	Klasa III
	2. Koncepcja bazy danych	15	– definiować encje i atrybuty encji – definiować związki między encjami i określa ich liczebność	– dobierać typ danych do określonych atrybutów encji – określać klucz główny dla encji	Klasa III

II. Użytkowanie języka SQL	1. Znajomość języka SQL	20	– stosować polecenia języka SQL – zmieniać rekordy w bazie danych przy użyciu języka SQL – usuwać rekordy w bazie danych przy użyciu języka SQL – tworzyć skrypty w strukturalnym języku zapytań	– definiować struktury baz danych przy użyciu instrukcji języka zapytań – wyszukiwać informacje w bazie danych przy użyciu języka SQL	Klasa Iii / Klasa IV
	2. Tworzenie bazy danych	20	– definiować tabele w bazie danych na podstawie projektu – wprowadzać dane do bazy danych – importować dane z pliku – eksportować strukturę bazy danych i dane do pliku	– definiować typy danych oraz atrybuty kolumn – programować skrypty automatyzujące proces tworzenia struktury bazy danych	Klasa III / Klasa IV
	3. Użytkowanie baz danych	20	– tworzyć formularze do wprowadzania danych i modyfikowania danych – identyfikować rodzaje zapytań – tworzyć zapytania i podzapytania do tabel bazy danych	– tworzyć raporty w bazie danych	Klasa III / Klasa IV
III. Modyfikacja i zarządzanie bazami danych	1. Modyfikacja baz danych	40	– analizować strukturę bazy danych w celu jej modyfikacji – rozbudowywać strukturę bazy danych tworząc tabele, pola, relacje i atrybuty – weryfikować poprawność struktury bazy danych po rozbudowie – usuwać elementy struktury bazy danych oraz dane – modyfikować strukturę bazy oraz dane bazy	– analizować strukturę bazy danych w celu jej modyfikacji – rozbudowywać strukturę bazy danych tworząc tabele, pola, relacje i atrybuty – weryfikować poprawność struktury bazy danych po rozbudowie – usuwać elementy struktury bazy danych oraz dane – modyfikować strukturę bazy oraz dane bazy	Klasa IV

	2. Zarządzanie bazami danych	30	– tworzyć użytkowników bazy danych – określać uprawnienia dla użytkowników – tworzyć kopię zapasową struktury bazy danych – przywracać dane z kopii zapasowej bazy danych – importować i eksportować tabele bazy danych	– kontrolować spójność bazy danych – weryfikować poprawność kopii zapasowej bazy danych – diagnozować i naprawiać bazę danych	Klasa IV

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Przedmiot Tworzenie i zarządzanie bazami danych należy realizować z podziałem na grupy w pracowni stron WWW, baz danych i aplikacji wyposażonej w:

- stanowisko dla nauczyciela wyposażone w komputer stacjonarny lub mobilny podłączony do sieci lokalnej z dostępem do Internetu, z oprogramowaniem systemowym i użytkowym, tablet z możliwością podłączenia do projektora, ekran lub tablicę multimedialną, projektor lub telewizor oraz urządzenie wielofunkcyjne lub drukarkę i skaner, oprogramowanie do tworzenia grafiki rastrowej i wektorowej oraz animacji, obróbki materiałów audio i wideo, różne systemy zarządzania bazą danych, oprogramowanie umożliwiające tworzenie aplikacji internetowych po stronie serwera i klienta w wybranych językach programowania, pakiety oprogramowania zawierające serwer WWW, SQL, PHP, serwer hostingowy do testowania projektów webowych, dokumentację techniczną,
- stanowiska komputerowe dla uczniów (jedno stanowisko dla jednego ucznia) wyposażone w komputer stacjonarny lub mobilny podłączony do intranetu, oprogramowanie do tworzenia grafiki rastrowej i wektorowej oraz animacji, obróbki materiałów audio i wideo, różne systemy zarządzania bazą danych, oprogramowanie umożliwiające tworzenie aplikacji internetowych po stronie serwera i klienta w wybranych językach programowania, podłączenie do sieci lokalnej z dostępem do Internetu, pakiety oprogramowania zawierające serwer WWW, SQL, PHP, serwer hostingowy do testowania projektów webowych, dokumentację techniczną.

Główną metodą nauczania powinny być:

- ćwiczenia praktyczne,
- praca projektami.

W celu indywidualizacji i dostosowania procesu nauczania, treści powinny charakteryzować się rosnącym (od podstawowego) poziomem trudności, biorąc pod uwagę indywidualne możliwości uczniów na danym etapie kształcenia. W celu wsparcia oraz uatrakcyjnienia procesu nauczania, do części zadań uczeń można łączyć w zespoły w taki sposób, aby ich wiedza i umiejętności uzupełniały się. Pozwoli to na wykształcenie umiejętności współdziałania, komunikacji oraz innych kompetencji personalnych i społecznych, takich jak odpowiedzialność i sumienność. Proces kształcenia należy też na bieżąco dostosowywać do możliwości uczniów oraz wprowadzać rozwiązania indywidualizujące.

Przykładowe zadania:

- Wykonaj zapytania do bazy danych wg podanego scenariusza,
- Stwórz bazę danych zgodnie z podanymi wytycznymi.

PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA

- ocena zaangażowania w pracę nad projektem,
- ocena wykonanych projektów.

PROPONOWANE METODY EWALUACJI PRZEDMIOTU

Strategia przeprowadzanej ewaluacji będzie polegała analizie danych (oceny zdobywane przez uczniów ze sprawdzianów, kartkówek i testów oraz zadań i testów praktycznych) z przedmiotu Bazy danych oraz Pracownia baz danych, które są ściśle ze sobą skorelowane. Dodatkowo ważnym elementem strategii jest opracowanie ankiety zwrotnej dla uczniów i rodziców na temat realizowanych treści. Dane zostaną poddane analizie ilościowej i jakościowej. Uzyskane wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych, przypisanych do danego działu programowego. Również w trakcie realizacji procesu kształcenia, ze względu na szybkość zmian w branży, ewaluacji będzie podlegać również przekazywany materiał.

6. Pracownia programowania aplikacji internetowych

Cele ogólne przedmiotu

1. Nabycie umiejętności tworzenia stron internetowych na podstawie projektu;
2. Poznanie i wykorzystanie algorytmów podczas programowania aplikacji internetowych;
3. Poznanie języków skryptowych realizowanych po stronie klienta;
4. Poznanie języków skryptowych realizowanych po stronie serwera;
5. Zastosowanie wybranych języków skryptowych wykonywanych po stronie klienta i po stronie serwera;
6. Wykonanie aplikacji internetowych;
7. Kształtowanie umiejętności współpracy w zespole;
8. Kształtowanie umiejętności planowania i organizacji pracy zespołu w celu wykonania przydzielonych zadań;
9. Obsługa baz danych.

Cele operacyjne

Uczeń potrafi:

1) wykorzystywać algorytmy poznane na zajęciach z informatyki;
2) określać typy danych;
3) definiować zmienne;
4) posługiwać się operatorami języka wykonywanego po stronie klienta;
5) posługiwać się operatorami języka wykonywanego po stronie serwera;
6) korzystać z funkcji języka wykonywanego po stronie klienta;
7) korzystać z funkcji języka wykonywanego po stronie serwera;
8) korzystać z obiektów języka wykonywanego po stronie klienta;
9) korzystać z obiektów języka wykonywanego po stronie serwera;
10) korzystać z metod języka wykonywanego po stronie klienta;
11) korzystać z metod języka wykonywanego po stronie serwera;
12) korzystać z bibliotek języka wykonywanego po stronie klienta;
13) korzystać z bibliotek języka wykonywanego po stronie serwera;
14) programować w językach skryptowych wykonywanych po stronie klienta;
15) programować w językach skryptowych wykonywanych po stronie serwera;
16) obsługiwać zdarzenia;
17) stosować biblioteki języków programowania;
18) analizować problem algorytmiczny pod kątem aplikacji webowych;
19) obsłużyć połączenie z bazą danych;
20) wysłać zapytanie do bazy danych;
21) wyświetlić odpowiedź z bazy danych;
22) obsługiwać zdarzenia baz danych;
23) wykonać testy aplikacji;
24) wyszukać błędy w aplikacji;
25) naprawić błędy w aplikacji;
26) przygotować plik pomocy dla użytkownika.

MATERIAŁ NAUCZANIA

Dział programowy	Tematy jednostek metodycznych	Liczba godz.	Wymagania programowe		Uwagi o realizacji
			Podstawowe Uczeń potrafi:	Ponadpodstawowe Uczeń potrafi:	Etap realizacji
III. Skryptowe języki wykonywane po stronie klienta	1. Zastosowanie algorytmów w aplikacjach internetowych	10	– omówić przykładowe algorytmy stosowane w aplikacjach internetowych (np. algorytm obliczający średnią arytmetyczną, algorytm obliczający wartość podatku, algorytmy z podstawy programowej informatyki z kształcenia ogólnego) – dokonać analizy algorytmów zapisanych w językach skryptowych o rozróżnić skryptowe języki skryptowe wykonywane po stronie klienta	– omówić zasady programowania strukturalnego, – skonstruować algorytmy wykorzystywane w aplikacjach internetowych, – zapisać algorytm postaci listy kroków, schematu blokowego itp.	Klasa III
	2. Typy danych, zmienne, stałe, łańcuchy, tablice	10	– zdefiniować proste typy danych stosowane w języku programowania – zdefiniować zmienne o typach prostych – zdefiniować stałe, – zdefiniować łańcuchy – opisywać operacje na zmiennych łańcuchowych, o tworzyć zmienne typu tablicowego	– posługiwać się prostymi i złożonymi typami danych – rozpoznawać złożone typy danych – definiować złożone typy danych – omówić przykładowe algorytmy na łańcuchach, – wykonywać operacje na zmiennych typu tablicowego	Klasa III

	3. Operatory i instrukcje sterujące	10	– rozpoznawać operatory arytmetyczne – rozpoznawać operatory przypisania – rozpoznawać operatory logiczne – omówić instrukcje sterujące: o Instrukcja warunkowa if, o pętla while, o pętla do...while, o pętla for,	– omówić zasadę działania operatorów arytmetycznych – omówić zasadę działania operatorów logicznych – dokonać analizy fragmentów kodu zapisanych z wykorzystaniem instrukcji sterujących: o Instrukcja warunkowa if, o pętla while, o pętla do...while, – pętla for	Klasa III
	4. Funkcje	10	– omówić sposób definiowania funkcji – tworzyć własne funkcje o analizować gotowe funkcje	– modyfikować funkcje pod potrzeby klienta zapisanej w języku skryptowym	Klasa III
	5. Klasy	10	– omówić etapy projektowania klas o omówić przykładowe klasy	– zdefiniować przykładowe metody klasy – zdefiniować konstruktor w klasie – omówić proces dziedziczenia	Klasa III
	6. Biblioteki	5	– wymienić przykładowe biblioteki i frameworki języka JavaScript, o wymienić przykładowe funkcje bibliote i frameworki języka JavaScript	– omówić wybrane funkcje z bibliotek i frameworków języka JavaScript: o jQuery, o Angular, – React	Klasa III
	7. Obsługa zdarzeń	5	– omówić obsługę zdarzeń myszy o omówić obsługę zdarzeń klawiatury		Klasa III

IV. Skryptowe języki wykonywane po stronie serwera	1. Typy danych, zmienne, stałe, łańcuchy, tablice	10	– zdefiniować proste typy danych stosowane w języku programowania – zdefiniować zmienne o typach prostych – zdefiniować stałe, zdefiniować łańcuchy, – opisywać operacje na zmiennych łańcuchowych, o tworzyć zmienne typu tablicowego	– posługiwać się prostymi i złożonymi typami danych, – rozpoznawać złożone typy danych – definiować złożone typy danych, omówić przykładowe algorytmy na łańcuchach – wykonywać operacje na zmiennych typu tablicowego	Klasa IV
	2. Operatory i Instrukcje sterujące	10	– rozpoznawać operatory arytmetyczne – rozpoznawać operatory przypisania – rozpoznawać operatory logiczne – omówić instrukcje sterujące: o Instrukcja warunkowa if, o pętla while, o pętla do...while, o pętla for,	– omówić zasadę działania operatorów arytmetycznych – omówić zasadę działania operatorów logicznych – dokonać analizy fragmentów kodu zapisanych z wykorzystaniem instrukcji sterujących: o Instrukcja warunkowa if, o pętla while, o pętla do...while, – pętla for,	Klasa IV
	3. Funkcje	10	– omówić sposób definiowania funkcji – tworzyć własne funkcje o analizować gotowe funkcje	– modyfikować funkcje pod potrzeby klienta zapisanej w języku skryptowym	Klasa IV
	4. Biblioteki	10	– wymienić przykładowe biblioteki i frameworki języka JavaScript – wymienić przykładowe funkcje bibliote i frameworki języka JavaScript, – wykazywać świadomość odpowiedzialności za wykonywaną pracę,	– omówić wybrane funkcje z bibliotek języka wykonywanego po stronie serwera, – przewidywać skutki podejmowanych działań, w tym skutki prawne – wybierać techniki radzenia sobie ze stresem odpowiednio do sytuacji,	Klasa IV

			– ocenić podejmowane działania – przewidywać konsekwencje niewłaściwego wykonywania czynności zawodowych na stanowisku pracy, w tym niewłaściwej eksploatacji maszyn i urządzeń na stanowisku pracy – rozpoznawać źródła stresu podczas wykonywania zadań zawodowych – przewidywać różne formy zachowań asertywnych, jako sposobów radzenia sobie ze stresem – rozróżniać techniki rozwiązywania konfliktów związanych z wykonywaniem zadań zawodowych – określić zakres umiejętności i kompetencji niezbędnych w wykonywaniu zawodu, – planować drogę rozwoju zawodowego o wskazywać możliwości podnoszenia kompetencji zawodowych, osobistych i społecznych	– wskazywać najczęstsze przyczyny sytuacji stresowych w pracy zawodowej – określić skutki stresu, – analizować własne kompetencje – wyznaczyć własne cele rozwoju zawodowego – identyfikować sygnały werbalne i niewerbalne, – stosować aktywne metody słuchania, – prowadzić dyskusje – udzielić informacji zwrotnej, – charakteryzować pożądaną postawę podczas prowadzenia negocjacji – wskazywać sposób prowadzenia negocjacji warunków porozumienia	
IV. Aplikacje internetowe	1. W językach wykonywanych po stronie klienta	35	– zastosować proste algorytmy w aplikacjach internetowych (np. algorytm obliczający średnią arytmetyczną, algorytm obliczający wartość podatku, algorytmy z podstawy programowej informatyki z kształcenia ogólnego)	– zastosować zasady programowania strukturalnego i obiektowego – tworzyć własne algorytmy, – zapisać algorytm w językach skryptowych wykonywanych po stronie klienta – posługiwać się prostymi i	klasa IV

			– zapisać algorytmy w językach skryptowych – interpretować algorytmy zapisane w językach skryptowych po stronie klienta, rozpoznawać proste typy danych – zastosować proste typy danych w programach – definiować zmienne o typach prostych – definiować stałe – definiować własne łańcuchy – wyświetlać łańcuchy – rozpoznawać operatory arytmetyczne, przypisania, logiczne – zastosować w programach instrukcje sterujące: ○ Instrukcja warunkowa if, ○ pętla while, ○ pętla do...while, ○ pętla for – tworzyć proste aplikacje – zastosować gotowe funkcje zdefiniowane w języku programowania – tworzyć zmienne typu tablicowego – tworzyć proste klasy, – tworzyć obiekty – dołączać biblioteki do kodu programu – skorzystać z wybranych funkcji z bibliotek i frameworków języka JavaScript – zastosować biblioteki	złożonymi typami danych, – rozpoznawać złożone typy danych – definiować złożone typy danych, – stosować złożone typy danych w programach – wykonywać operacje na łańcuchach – stosować operatory arytmetyczne, przypisania, logiczne – analizować kod zapisany w języku skryptowym po stronie klienta, – tworzyć własne funkcje, – wykonywać operacje na zmiennych typu tablicowego, – tworzyć metody klasy, – tworzyć konstruktor w klasie, – korzystać z dziedziczenia, – zastosować gotowe klasy języka programowania – skorzystać z wybranych funkcji z bibliotek i frameworków języka JavaScript: ○ jQuery, ○ Angular, ○ React – zastosować w programie obsługę zdarzeń myszy – zastosować w programie obsługę zdarzeń klawiatury – odwoływać się do elementów strony – pebireać element za pomocą funkcji np. getElementById(), getElementsByTagName() – zastosować pętle po kolekcjach	

			wykorzystywane w skryptach po stronie klienta – wymienić przykładowe zdarzenia – wymienić funkcje obsługujące zdarzenia – używać języka JavaScript do kontroli drzewa "DOM" – skorzystać z DOM API dla JavaScript – modyfikować DOM API – zdefiniować skrypty obsługujące formularze i kontrolki HTML – utworzyć formularz odczytujący dane z poszczególnych pól – wyszukać błędy w kodzie źródłowym programu – poprawiać błędy w tworzonych programach – zastosować komentarze w kodzie źródłowym programu	– stworzyć stronę internetową reagującą na zdarzenia użytkownika, takie jak klikanie, przewijanie czy wprowadzanie danych do formularza – animować strony internetowe korzystając z np. window.setInterval, window.requestAnimationFrame – utworzyć formularz weryfikujący poprawność wprowadzanych danych – wykorzystuje mechanizmy walidacji formularzy HTML za pomocą mechanizmów HTMLS	
	2. W językach wykonywanych po stronie serwera	50	– zastosować proste typy danych w programach – definiować zmienne o typach prostych – definiować stałe – definiować własne łańcuchy, – wyświetlać łańcuchy – rozpoznawać podstawowe operatory arytmetyczne przypisania, logiczne – zastosować instrukcje sterujące: ○ Instrukcja warunkowa if, ○ pętla while, ○ pętla do...while, ○ pętla for	– posługiwać się prostymi i złożonymi typami danych, – rozpoznawać złożone typy danych – definiować złożone typy danych, – stosować złożone typy danych w programach – wykonywać operacje na łańcuchach – stosować podstawowe operatory arytmetyczne, przypisania, logiczne – analizować zaawansowane kody zapisane w języku skryptowym – tworzyć własne funkcje – wyszukiwać błędy w funkcjach – analizować utworzone funkcje	Klasa IV

			– analizować kody zapisane w języku skryptowym – stosować funkcje zdefiniowane w języku programowania – analizować funkcje zapisane w języku skryptowym – tworzyć zmienne typu tablicowego, – tworzyć proste klasy – tworzyć obiekty – definiuje skrypty obsługujące formularze – stosuje metody przesyłania danych z formularza. – omówić funkcje do obsługi ciasteczek – zastosować funkcje do obsługi sesji – zastosować mechanizm do obsługi plików	– wykonywać operacje na zmiennych typu tablicowego, – tworzyć metody klasy – tworzyć konstruktor w klasie – korzystać z dziedziczenia – zastosować gotowe klasy języka programowania – programuje wysyłanie danych z formularza HTML za pomocą metody GET, POST – korzystać z funkcji do obsługi ciasteczek – korzystać z funkcji do obsługi sesji – korzysta z funkcji do obsługi plików – omówić funkcje do obsługi sesji, – omówić mechanizm do obsługi plików	
	3. Wyszukiwać błędy w aplikacjach	10	– wyszukać błędy w kodzie źródłowym programu – poprawiać błędy w tworzonych programach	– zastosować debugger w przeglądarce internetowej – wykonać testy tworzonych programów	Klasa IV
	4. Tworzyć dokumentację aplikacji	15	– zastosować komentarze w kodzie źródłowym programu – określić czas realizacji zadań – zrealizować działania w wyznaczonym czasie – pracować w zespole – ponosząc odpowiedzialność za wspólnie realizowane zadania, – przestrzegać podziału ról, zadań i odpowiedzialności w zespole – określić strukturę zespołu,	– tworzyć instrukcję użytkownika programu – tworzyć dokumentację programu – monitorować realizację zaplanowanych działań – dokonać modyfikacji zaplanowanych działań – dokonać samooceny wykonanej pracy – opisać sposób przeciwdziałania problemom w zespole realizującym zadania	Klasa IV

			– przygotować zadania zespołu do realizacji, – planować realizację zadań zapobiegających zagrożeniom bezpieczeństwa i ochrony zdrowia – komunikować się ze współpracownikami, – wskazywać wzorce prawidłowej współpracy w zespole – rozdzielać zadania według umiejętności i kompetencji członków zespołu – koordynować realizację zadań zapobiegających zagrożeniom bezpieczeństwa i ochrony zdrowia – monitorować proces wykonywania zadań – dokonać analizy rozwiązań technicznych i organizacyjnych warunków i jakości pracy	– opisać techniki rozwiązywania problemów – wskazać na wybranym przykładzie, metody i techniki rozwiązywania problemu, – angażować się w realizację wspólnych działań zespołu, – modyfikować sposób zachowania, uwzględniając stanowisko wypracowane wspólnie z innymi członkami zespołu – oszacować czas potrzebny na realizację określonego zadania – przydzielać zadania członkom zespołu zgodnie z harmonogramem planowanych prac – oceniać przydatność poszczególnych członków zespołu do wykonania zadania, – ustalać kolejność wykonywania zadań zgodnie z harmonogramem prac – formułować zasady wzajemnej pomocy – wydawać dyspozycje osobom wykonującym poszczególne zadania – opracować dokumentację dotyczącą realizacji zadania według panujących standardów – 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 – proponować rozwiązania techniczne i organizacyjne mające na celu poprawę warunków i jakości pracy	

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

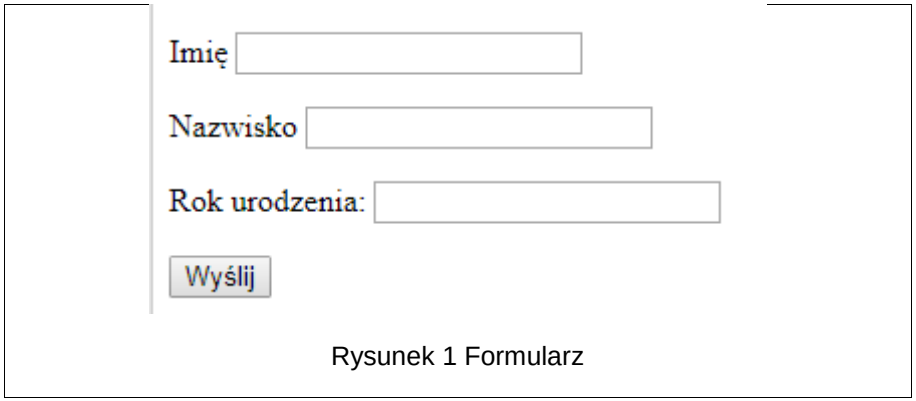
Zajęcia należy realizować w pracowni sieciowych systemów operacyjnych, z podziałem na grupy. Należy zaznaczyć, że przy jednym stanowisku komputerowym przebywa jedna osoba. Zajęcia edukacyjne powinny być realizowane zgodnie z podstawą programową w pracowni wyposażonej w:

- stanowiska komputerowe dla uczniów (jeden komputer stacjonarny lub mobilny podłączony do intranetu dla jednego ucznia)
- różne systemy zarządzania bazą danych,
- oprogramowanie umożliwiające tworzenie aplikacji internetowych po stronie serwera i klienta w wybranych językach programowania,
- pakiety oprogramowania zawierające serwer WWW, SQL, PHP,
- serwer hostingowy do testowania projektów webowych,
- dokumentację techniczną.

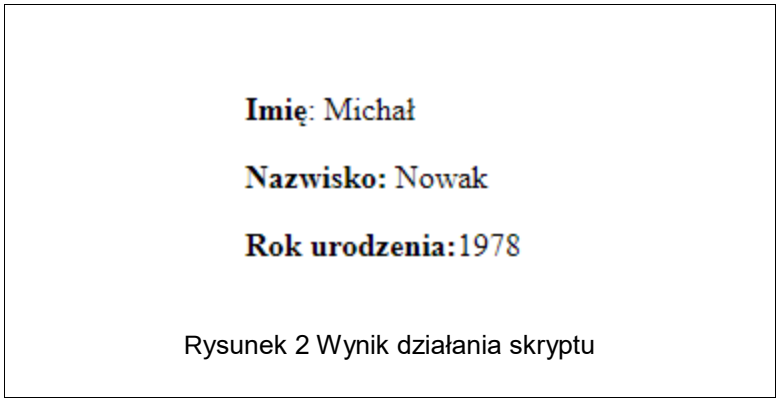
Na zajęciach zalecamy sortować aktywizujące metody kształcenia, ze szczególnym uwzględnieniem metody ćwiczeń, dyskusji dydaktycznej oraz projektów. W zakresie organizacji pracy należy stosować instrukcje do zadań, podawanie dodatkowych zaleceń, instrukcji do pracy indywidualnej.
Zadania realizowane w grupach należy szczególnie monitorować, 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.

Przykładowe zadanie:

Utwórz formularz zgodnie z rysunkiem 1. Następnie napiszą aplikację, która wyświetli po wciśnięciu przycisku *Wyślij* na stronie wynik.php wyświetli dane wpisane do formularza. Wynik działania skryptu pokazano na rysunku 2.



Rysunek 1 Formularz



Rysunek 2 Wynik działania skryptu

PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA

Przedmiot **Pracownia programowania aplikacji internetowych** jest przedmiotem praktycznym, dlatego na zajęciach oceniamy ucznia z wykonywanych ćwiczeń i sprawdzianów praktycznych. Dopuszczalne jest stosowanie również odpowiedzi ustnych.

PROPONOWANE METODY EWALUACJI PRZEDMIOTU

Ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, pracy na lekcji uczniów oraz ankiet przygotowanych dla uczniów i rodziców.

Wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego.

W trakcie realizacji procesu kształcenia ewaluacji musi podlegać również materiał edukacyjny. W branży teleinformatycznej zmienia się on bardzo szybko.

7. Pracownia multimediów i grafiki komputerowej

Cele ogólne przedmiotu

1. Przygotowanie grafiki na stronę internetową;
2. Przygotowanie multimediów do wykorzystania na stronie internetowej.

Cele operacyjne

Uczeń potrafi:

1) wymienić formaty plików graficznych;
2) scharakteryzować formaty plików graficznych;
3) dobrać formaty plików graficznych dla strony internetowej;
4) wymienić formaty plików wideo;
5) scharakteryzować formaty plików wideo;
6) dobrać formaty plików wideo dla strony internetowej;
7) wymienić formaty plików dźwiękowych;
8) scharakteryzować formaty plików dźwiękowych;
9) dobrać formaty plików dźwiękowych dla strony internetowej

MATERIAŁ NAUCZANIA

Dział programowy	Tematy jednostek metodycznych	Liczba godz.	Wymagania programowe		Uwagi o realizacji
			Podstawowe Uczeń potrafi:	Ponadpodstawowe Uczeń potrafi:	Etap realizacji
I. Grafika i multimedia na stronach internetowych	1. Grafika komputerowa	17	– dobrać oprogramowanie do obróbki grafiki komputerowej – wykonać edycję plików graficznych na potrzeby stron internetowych	– zastosować różne modele barw – osadzić tekst na grafice oraz dobiera jego krój i styl – skorzystać z funkcji edytora grafiki wektorowej – skorzystać z funkcji edytora	Klasa II

				grafiki rastrowej – zaprojektować elementy graficzne dla strony internetowej – przygotować grafikę dla strony internetowej	
	2. Multimedia	13	– dobrać oprogramowanie do edycji obrazu ruchomego i dźwięku – wykonać animacje na potrzeby strony internetowej – osadzić elementy multimedialne na stronie internetowej	– przygotować materiały wideo na potrzeby strony internetowej – przygotować materiał dźwiękowy na potrzeby strony internetowej	Klasa II

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Zajęcia należy realizować w pracowni stron WWW, baz danych i aplikacji, z podziałem na grupy. Należy zaznaczyć, że przy jednym stanowisku komputerowym przebywa jedna osoba. Zajęcia edukacyjne powinny być realizowane zgodnie z podstawą programową w pracowni wyposażonej w:

- stanowiska komputerowe dla uczniów (jeden komputer stacjonarny lub mobilny podłączony do intranetu dla jednego ucznia)
- oprogramowanie umożliwiające tworzenie grafiki wektorowej i rastrowej oraz 3D,
- oprogramowanie umożliwiające tworzenie i obróbkę plików multimedialnych.

Na zajęciach zalecamy sortować aktywizujące metody kształcenia, ze szczególnym uwzględnieniem metody ćwiczeń, dyskusji dydaktycznej oraz projektów. W zakresie organizacji pracy należy stosować instrukcje do zadań, podawanie dodatkowych zaleceń, instrukcji do pracy indywidualnej.
Zadania realizowane w grupach należy szczególnie monitorować, 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.

Przykładowe zadanie:

Wykonaj montaż krótkiego filmu multimedialnego, celem udostępnienia go na stronie internetowej.

PROPONOWANE METODY SPRAWDZANIA OSIĄGNIĘĆ EDUKACYJNYCH UCZNIA

Przedmiot **Pracownia multimediów i grafiki komputerowej** jest przedmiotem praktycznym, dlatego na zajęciach oceniamy ucznia z wykonywanych ćwiczeń i sprawdzianów praktycznych. Dopuszczalne jest stosowanie również odpowiedzi ustnych.

PROPONOWANE METODY EWALUACJI PRZEDMIOTU

Ewaluacji będzie polegała na tzw. twardej analizie danych, którymi są oceny zdobywane przez uczniów ze sprawdzianów, pracy na lekcji uczniów oraz ankiet przygotowanych dla uczniów i rodziców.

Wyniki pozwolą na określenie, które zagadnienia sprawiają uczniom problemy, a dzięki temu będzie można skorygować liczbę godzin dydaktycznych przypisanych do danego działu programowego.

W trakcie realizacji procesu kształcenia ewaluacji musi podlegać również materiał edukacyjny. W branży teleinformatycznej zmienia się on bardzo szybko.

8. 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 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	Klasa III

				wypowiedzi/tekście określone informacje.	
	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

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.

9. Projektowanie oprogramowania

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,

2. 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

3. 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 II
	2. 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 II
	3. 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 II
	4. 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,	podać przykłady wyrażeń wykorzystujących priorytety operatorów	Klasa II

			logicznymi, operatorami do obsługi łańcuchów, bitowymi		
	5. 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 II
	6. Instrukcje sterujące		określić składnię instrukcji warunkowej i wyboru, określić składnię instrukcji pętli		Klasa II
	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 III
	2. Zasady programowania		zapisywać proste algorytmy w języku programowania	objaśnić pojęcie rekurencji, definiować funkcję rekurencyjną, zapisywać złożone algorytmy w języku programowania	Klasa III
II. Obiekto-wość	3. Zasady programowania obiektowego		stosować paradygmaty programowania obiektowego, wyjaśnić pojęcia klasa, obiekt, metoda, pole, dziedziczenie, hermetyzacja, polimorfizm, zdefiniować klasę jako reprezentację obiektu		Klasa III

			ze świata rzeczywistego, tworzyć obiekty jako instancje klasy, odwołać się do pól i metod obiektu		
	4. 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 III
	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
	2. Szablony (wzorce) klas		wyjaśnić rolę szablonów klas w programowaniu obiektowym	określać szablony klas dla prostych typów liczbowych	Klasa III
	3. 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	4. 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	Klasa III

				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	

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Do osiągania założonych celów kształcenia w zakresie przedmiotu Projektowanie oprogramowania 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 Projektowanie oprogramowania, 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 Projektowanie oprogramowania 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.

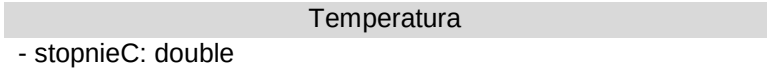
Stosowane przez nauczyciela ocenianie powinno korzystać z zasad występujących w ocenianiu kształtującym, ma być dla ucznia informacją zwrotną, która pomaga mu się uczyć. Nauczyciel wskazuje silne strony ucznia i doradza, jak je rozwijać; wyraźnie i konstruktywnie informuje o stronach słabych i o tym, 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:
   - **a.** wczytuje dwie liczby całkowite dol, gora, będące zakresem,
   - **b.** wczytuje różnicę ciągu arytmetycznego r,
   - **c.** oblicza i wyświetla kolejne wyrazy ciągu arytmetycznego mieszczące się w zakresie od wartości dol do gora,
   - **d.** 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:



+ 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. Gręboszz Jerzy, Opus magnum C++11, Programowanie w języku C++ (komplet), Wyd.: Gręboszz Jerzy,
4. Python dla każdego. Podstawy programowania – Michael Dawson,wyd. Helion,
5. Eric Matthes, Python. Instrukcje dla programisty, wyd. Helion,
6. Al Sweigart, Automatyzacja nudnych zadań z Pythonem, wyd. Helion,
7. http://www.cplusplus.com/
8. https://docs.python.org/3/tutorial/
9. http://codecademy.com
10. http://w3schools.com
11. https://doc.qt.io/qtcreator/index.html
12. http://www-cs.ccny.cuny.edu/~wolberg/cs221/qt/books/C++-GUI-Programming-with-Qt-4-1st-ed.pdf
13. https://qmlbook.github.io/assets/qt5_cadaques.pdf

10. Programowanie obiektowe

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;

2. 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;

3. 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. Programowanie desktopowych aplikacji	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

okienko-wych	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	Narzędzia i metodologie planowania i zarządzania		określić funkcje narzędzi do zarządzania projektem ,	korzystać z programów wspierających zarządzanie projektami, np. Jira,	Klasa III

zarządzanie projektem	projektem		dobrać narzędzia do zarządzania projektem, zastosować narzędzia wizualizacji w zarządzaniu etapami projektu, zadaniami i czasem, np. diagram Gantta	Trello, korzystać z systemu kontroli wersji, np. Git	
	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	Klasa III

				w programowaniu obiektowym, np. Metoda szablonowa (Template method), Fasada (Facade), Kompozyt (Composite)	
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 IV
	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 IV

	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 IV
	Obsługa plików oraz wyjątków.			- implementować obsługę plików w wybranym języku programowania, - obsługiwać wyjątki i błędy w wybranym języku programowania,	Klasa IV
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 IV
	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 IV

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 IV
	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 IV

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Do osiągania założonych celów kształcenia w zakresie przedmiotu 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, 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 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 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.

11. Pracownia programowania 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

2. 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,

3. 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 III

	2. 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 III
	3. 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	4. 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 III

	5. 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	
	6. 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 III
	Środowisko programistyczne dla obiektowych aplikacji konsolowych		kompilować i uruchamiać programy	analizować błędy w kodzie za pomocą debuggera	Klasa III

	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 IV
	Algorytmy wyszukiwania		stosować algorytm wyszukiwania dla różnych zestawów danych (tablic, list, kolejek, stosów)		Klasa IV
	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 IV
	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 IV
	Zasady programowania obiektowego		odzwierciedlać rzeczywistość w formie zbioru obiektów, definiować klasy i tworzyć obiekty, stosować specyfikatory dostępu,	planować aplikację z zastosowaniem hermetyzacji	Klasa IV
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 IV

	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 IV
	Szablony (wzorce) klas		wyjaśnić rolę szablonów klas w programowaniu obiektowym,	opisywać szablony klas dla prostych typów liczbowych	Klasa IV
	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 IV
IV. Wprowadzenie do programowania desktopowych aplikacji okienkowych	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	stworzyć klasyczne okno aplikacji desktopowej (pasek menu, pasek narzędzi, obszar dokumentu, pasek statusu), np. przeglądarka zdjęć	Klasa IV

			zaprogramować obsługę zdarzeń myszy i klawiatury,		
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.Tworzenie desktopowych aplikacji okienkowych	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,	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

			doskonalić jakość wykonywanych działań; doskonalić umiejętności zawodowe, pracować w zespole.		

PROCEDURY OSIĄGANIA CELÓW KSZTAŁCENIA PRZEDMIOTU

Do osiągnięcia założonych celów kształcenia w zakresie przedmiotu Pracownia programowania 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 Pracownia programowania 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 Pracownia programowania 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)

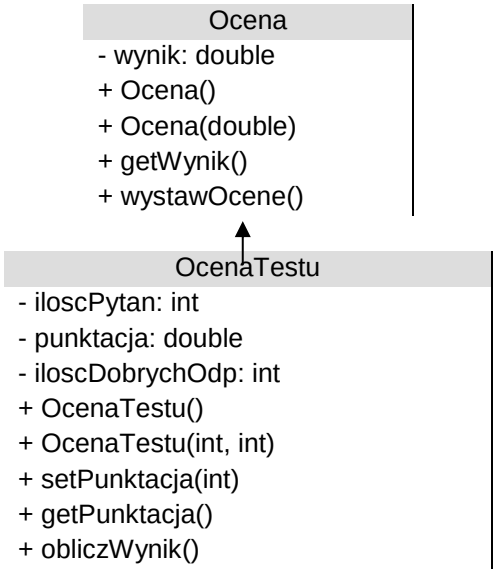
Wskazane jest też, by uczniowie dokonywali samooceny i oceny kolegów z zespołu według zaproponowanych przez nauczyciela kryteriów.

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



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:

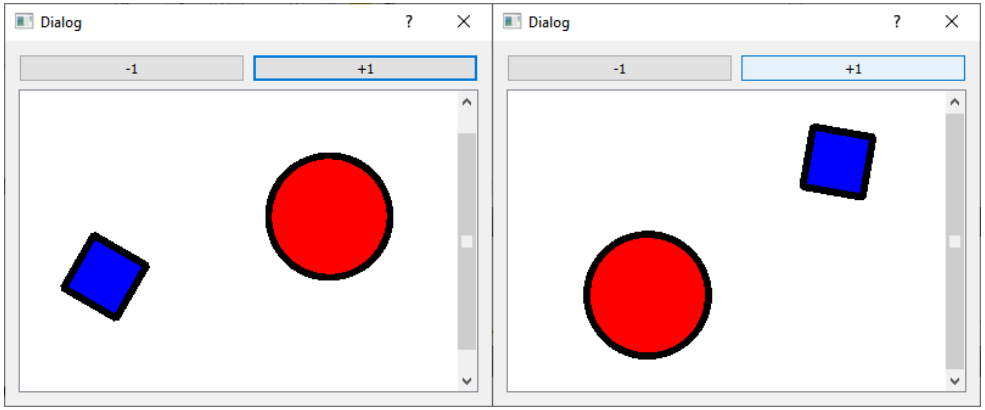
- konstruktor domyślny – zeruje wszystkie pola,
- konstruktor parametryczny – wypełnia pola wartościami argumentów konstruktora,

- 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

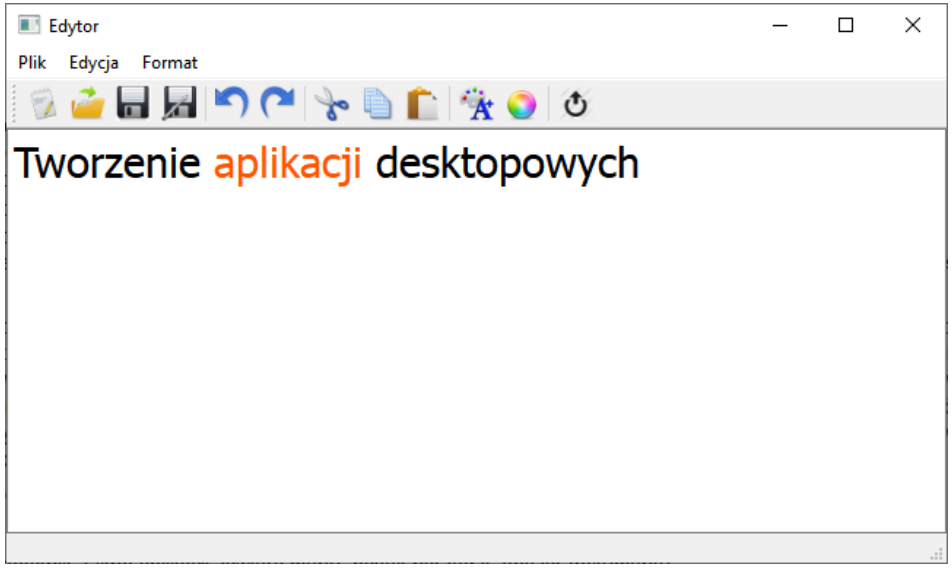
- wczytaj ilość pytań w teście i ilość udzielonych dobrych odpowiedzi,
- utwórz obiekt klasy OcenaTestu z parametrami równymi wczytanym wartościom,
- wyświetl: punktację, wynik punktowy testu i ocenę za test.

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

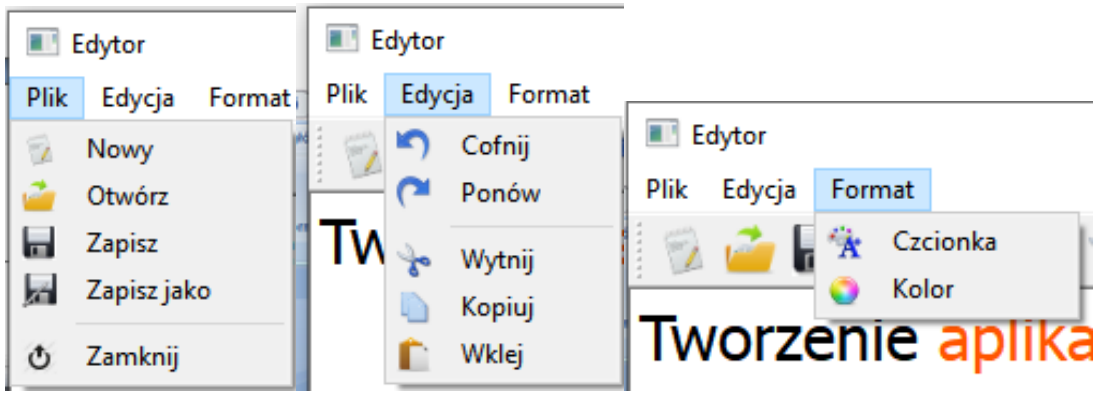


Rysunek.

3. 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. Gręboszz Jerzy, Opus magnum C++11, Programowanie w języku C++ (komplet), Wyd.: Gręboszz Jerzy,
4. **Python dla każdego. Podstawy programowania – Michael Dawson,wyd. Helion,**
5. **Eric Matthes, Python. Instrukcje dla programisty, wyd. Helion,**
6. **Al Sweigart, Automatyzacja nudnych zadań z Pythonem, wyd. Helion,**
7. http://www.cplusplus.com/
8. https://docs.python.org/3/tutorial/
9. http://codecademy.com
10. http://w3schools.com
11. https://doc.qt.io/qtcreator/index.html
12. http://www-cs.ccny.cuny.edu/~wolberg/cs221/qt/books/C++-GUI-Programming-with-Qt-4-1st-ed.pdf
13. https://qmlbook.github.io/assets/qt5_cadaques.pdf

12. Pracownia programowania 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 III

	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 III
	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 III
	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 III
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	- stworzyć aplikację do malowania na ekranie urządzenia mobilnego rozpoznającą dotyk: stuknięcie, przytrzymanie, przeciągnie, - wykorzystać dedykowane biblioteki do	Klasa IV

			mobilnym, - wykorzystać lokalizację GPS w aplikacji mobilnej, - dostosować interfejs aplikacji mobilnej do konkretnego rodzaju urządzenia,	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,	
	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 Pracownia programowania 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.

13. Pracownia programowania zaawansowanych aplikacji webowych

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;
8. Poznanie frameworków Node.js iDjango;
9. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem wzorców projektowych;
10. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem frameworku jQuery;
11. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem frameworku Node.js;
12. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem frameworku Django;
13. Nabycie umiejętności programowania aplikacji webowej z wykorzystaniem języka programowania np. PHP, C#, Python, JavaScript;
14. Nabycie umiejętności wdrażania aplikacji webowej;

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,
43) stosować środowisko programistyczne do realizacji projektu aplikacji webowej,
44) korzystać z dedykowanych narzędzi dostępnych w środowisku programistycznym przy tworzeniu aplikacji webowej w danym frameworku,
45) korzystać z dedykowanych narzędzi dostępnych w środowisku programistycznym przy tworzeniu aplikacji webowej w danym języku prrogramowania,
46) implementować podstawowe operacje w jQuery,

47) implementować wybór elementów w tym selektorów poprzez jQuery,
48) zmodyfikować wygląd aplikacji webowej poprzez kodu w tym HTML poprzez jQuery,
49) odbierać zdarzenia w jQuery,
50) odpowiadać na zdarzenia w jQuery,
51) generować żądania GET w w jQuery,
52) przekazywać nagłówki HTTP w jQuery,
53) wczytywać kod XML w jQuery,
54) obsługiwać zdarzenia AJAX w jQuery,
55) odczytać dane JSON z zewnętrznego serwera w jQuery,
56) implementować przeciąganie elementów w jQuery UI,
57) implementować upuszczanie elementów w jQuery UI,
58) implementować zmianę kolejności elementów przy wykorzystaniu elementów sortowalnych w jQuery UI,
59) implementować zaznaczanie elementów w jQuery UI,
60) implementować grupowanie treści w jQuery,
61) implementować uzupełnianie treści w jQuery,
62) implementować zmianę elementu w przycisk w jQuery,
63) implementować okna dialogowe w jQuery,
64) implementować pobieranie liczb za pomocą suwaka w jQuery,
65) implementować nawigację w aplikacji przy użyciu kart w jQuery,
66) implementować pasek postępu w jQuery,
67) dostosować aplikację webową do urządzeń mobilnych w jQuery,
68) implementować nawigowanie aplikacje webowej przy użyciu jQuery Mobile,
69) implementować interakcję aplikacji webowej z użytkownikiem na urządzeniu mobilnym przy użyciu jQuery Mobile,
70) skorzystać z gotowych wtyczek w jQuery,
71) utworzyć własne wtyczki w jQuery,
72) utworzyć własne skrypty śródliniowe w Node.js,
73) zdefiniować własne moduły w Node.js,
74) implementować obsługę zdarzeń w Node.js,
75) implementować odczyt danych z pliku w Node.js,
76) implementować zapis danych w pliku w Node.js,
77) wykonywać operacje na plikach w Node.js,
78) kopiować pliki za pomocą strumieni w Node.js,
79) implementować prosty serwer HTTP w Node.js,
80) implementować odpowiedź typu HTML w Node.js,

81) zdefiniować routing w Node.js,
82) implementować odpowiedź JSON w Node.js,
83) wykorzystać strumienie i HTTP w node.js,
84) stworzyć aplikację z wykorzystaniem frameworku express.js w Node.js,
85) zainstalować MongoDB w Node.js,
86) implementować podstawowe polecenia Mongo,
87) połączyć Node.js z bazą danych,
88) dodawać nowe wartości do nazy danych z poziomu Node.js,
89) wykorzystać Web Socket w Node.js,
90) wykorzystać inne biblioteki Node.js,
91) zainstalować język Python na serwerze,
92) dokonać edycji zmiennych środowiskowych na serwerze w celu instalacji Django,
93) zainstalować Django na serwerze,
94) tworzyć projekt aplikacji webowej w Django,
95) przygotować bazę danych do projektu aplikacji wykonanej w Django,
96) opracować model danych stosowanych w aplikacji webowej wykonane w Django,
97) wprowadzić dane do tabel w bazie danych dla aplikacji webowej wykonanej w Django,
98) implementować formularz logowania się użytkowników do aplikacji webowej wykonanej w Django,
99) zautomatyzować procesy np. powiadamiane użytkowników w aplikacji webowej wykonanej w Django,
100) zaprogramować aplikacje internetowe w wybranym języku programowania np. PHP, C#, Python, JavaScript,
101) zaprogramować aplikacje webowe wykorzystujące mechanizm sesji i ciasteczek,
102) zaprogramować aplikacje webowe zawierające dynamiczne formularze,
103) zaprogramować systemy logowania do aplikacji webowej,
104) zaprogramować system kontroli dostępu do określonych elementów witryny,
105) zaprogramować aplikacje webowe z dostępem do baz danych,
106) zaprogramować wybrane elementy e-sklepu,
107) zaprogramować wybrane elementy portalu społecznościowego,
108) zaprogramować wybrane elementy serwisu ogłoszeń internetowych,
109) zaprogramować wybrane elementy serwisu rezerwacyjnego,
110) testować zaprogramowaną aplikację użytkownika,
111) dokumentować kod zaprogramowanej aplikacji,
112) 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. 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	- zainstalować Django na serwerze, - opisać zasady tworzenia okna logowania i okna rejestracji użytkowników w Django, - obsługiwać sesje i ciasteczka z poziomu Django,	Klasa III

			Django,		
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
III. Programowanie aplikacji webowych z wykorzystaniem	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		Klasa IV

frameworków			frameworkami Node.js i Django,		
	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 IV
	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 IV
IV. 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 IV

	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 IV

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 Pracownia programowania zaawansowanych 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 Pracowania programowania zaawansowanych 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

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.

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

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

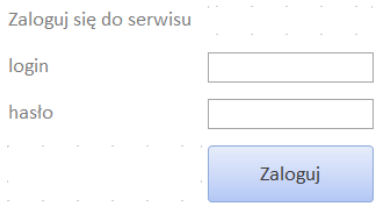
Jakie czynności należy wykonać po stronie serwera Apache współpracował z nim język Python? Do udzielenia odpowiedzi przyjmij, że na serwerze jest zainstalowana tylko usługa Apache.

Zadanie 4

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 5

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 6

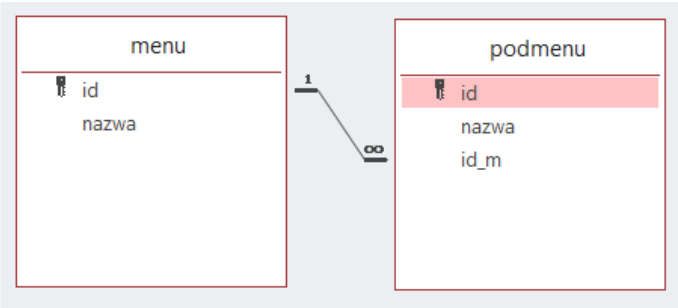
Utwórz bazę danych o nazwie **serwis** na serwerze, w bazie utwórz tabelę menu i podmenu oraz wypełnij danymi zgodnie z relacją na rysunkiem poniżej.

podmenu | menu

id	nazwa
1	Oferta
2	Nasze punkty
3	Promocje
4	Kontakt
5	O nas

podmenu | menu

id	nazwa	id_m
1	Rowery	1
2	Hulajnogi	1
3	Rolki	1
4	Motorowery	1
5	Warszawa	2
6	Gdańsk	2
7	Poznań	2
8	Telefony	4
9	Adresy	4



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. 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,
5. tworzenie aplikacji webowej z wykorzystaniem frameworku Node.js,
6. tworzenie aplikacji webowej z wykorzystaniem frameworku Django,
7. tworzenie aplikacji webowej z wykorzystaniem frameworku jQuery,

8. tworzenie aplikacji webowej z wykorzystaniem wybranego języka: PHP, C#, Python, JavaScriopt,
9. programowanie wybranych elementów aplikacji webowych,
10. programowanie aplikacji webowych z wykorzystaniem baz danych.

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.
- 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,

14. Zajęcia specjalizujące

Cele ogólne przedmiotu

1. Utrwalenie dotychczasowych wiadomości poprzez kurs / szkolenie / pracownię specjalizującą, o wybranej tematyce z zakresu treści podstawy programowej dla zawodu technik programista.
2. Realizacja – 7 godzin tygodniowo.

15. 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,

2. Nabycie umiejętności
   - przestrzegania przepisów BHP, przeciwpożarowych i ochrony środowiska,
   - wykonywania zadań zawodowych w warunkach zakładu pracy.

3. 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,

2. Obsługa oprogramowania używanego w firmie
   - obsługa oprogramowania systemowego i użytkowego stosowanego w firmie,
   - ochrona danych, programów i procesów przetwarzania informacji.
3. 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.
4. 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.

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

_