

SYLABUS**DOTYCZY CYKLU KSZTAŁCENIA 2021/22 – 2024/25***(skrajne daty)*

Rok akademicki 2022/2023

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	Programowanie obiektowe
Kod przedmiotu*	
Nazwa jednostki prowadzącej kierunek	Kolegium Nauk Przyrodniczych
Nazwa jednostki realizującej przedmiot	Instytut Inżynierii Materiałowej
Kierunek studiów	Mechatronika
Poziom studiów	Studia I-go stopnia
Profil	Praktyczny
Forma studiów	Studia niestacjonarne
Rok i semestr/y studiów	Rok II, semestr 3
Rodzaj przedmiotu	Przedmiot kierunkowy
Język wykładowy	Polski
Koordinator	dr hab. inż. Lucyna Leniowska, prof. UR
Imię i nazwisko osoby prowadzącej / osób prowadzących	dr hab. inż. Lucyna Leniowska, prof. UR dr inż. Marcin Grochowina

* -opcjonalnie, zgodnie z ustaleniami w Jednostce

1.1. Formy zajęć dydaktycznych, wymiar godzin i punktów ECTS

Semestr (nr)	Wykł.	Ćw.	Konw.	Lab.	Sem.	ZP	Prakt.	Inne (jakie?)	Liczba pkt. ECTS
3	18			18					5

1.2. Sposób realizacji zajęć

- ☐ zajęcia w formie tradycyjnej
☒ zajęcia realizowane z wykorzystaniem metod i technik kształcenia na odległość

1.3 Forma zaliczenia przedmiotu (z toku) (egzamin, zaliczenie z oceną, zaliczenie bez oceny)

wykład - egzamin

laboratorium - zaliczenie z oceną

2. WYMAGANIA WSTĘPNE

Podstawy języka programowania C++. Znajomość zagadnień z przedmiotu podstawy programowania .

3. CELE, EFEKTY UCZENIA SIĘ, TREŚCI PROGRAMOWE I STOSOWANE METODY DYDAKTYCZNE

3.1 Cele przedmiotu

C1	Kontynuacja i rozwinięcie zagadnień realizowanych w zakresie przedmiotu „Podstawy programowania”. Zapoznanie z obiektowymi technikami programowania.
C2	Nabycie umiejętności praktycznych w zakresie stosowania złożonych struktur danych oraz opracowania programów w języku C++ z zastosowaniem podejścia obiektowego.

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu Student:	Odniesienie do efektów kierunkowych ¹
EK_01	Ma ogólną wiedzę z zakresu obiektowych technik programowania. Definiuje podstawowe pojęcia: abstrakcja, dziedziczenie, hermetyczność, polimorfizm. Posiada wiedzę na temat struktur statycznych i dynamicznych. Definiuje klasy i tworzy ich instancje.	K_W09
EK_02	Potrafi pozyskać wiedzę z dostępnych źródeł w celu stworzenia efektywnie działającego oprogramowania, dokonywać interpretacji, a także wyciągać wnioski oraz formułować i uzasadniać opinie.	K_U01
EK_03	Potrafi sformułować specyfikację zadania programistycznego, wykonać projekt w grupie wraz z jego dokumentacją.	K_U07
EK_04	Ma umiejętność samokształcenia się, m.in. w celu doskonalenia umiejętności programowania i podnoszenia kompetencji zawodowych, korzystając również z materiałów anglojęzycznych	K_U16
EK_05	Potrafi posłużyć się właściwie dobranymi metodami i algorytmami. Stosuje typy obiektowe w programowaniu.	K_U13
EK_06	Potrafi przygotować założenia i zaprojektować aplikację na potrzeby wybranego zadania oraz opracować i zrealizować harmonogram prac.	K_U18
EK_07	Ma świadomość, iż w przypadku trudności z rozwiązaniem problemu należy skorzystać z wiedzy ekspertów	K_K02

3.3 Treści programowe

A. Problematyka wykładu

Treści merytoryczne
1. Programowanie proceduralne a programowanie obiektowe. Ogólne wiadomości o obiektowej technice programowania. Podstawowe pojęcia. Rozwój obiektowych języków programowania i ich podstawowe cechy.
2. Przegląd typów strukturalnych w języku C++. Tablice statyczne i dynamiczne, dostęp do elementów tablicy, wskaźniki.

3. Referencje. Zastosowania referencji i wskaźników w funkcjach.
4. Przegląd typów strukturalnych w C++ c.d. Struktury, unie.
5. Łańcuchy w stylu C++ , klasa string. Rzutowanie i konwersja typów.
6. Pojęcie klasy i obiektu. Odwołania do obiektów, cechy obiektów. Przykłady klas.
7. Metody i funkcje. Przekazywanie parametrów, parametry domniemane. Przeciążanie metod.
8. Konstruktory i destruktory. Przykłady.
9. Hermetyzacja. Hierarchia klas. Dziedziczenie jednokrotne i wielokrotne.
10. Zgodność typów przy dziedziczeniu. Polimorfizm w hierarchii klas. Funkcje i klasy zaprzyjaźnione.
11. Metody wirtualne, dynamiczne i abstrakcyjne. Przeciążanie operatorów. Przykłady.
12. Metaprogramowanie. Szablony funkcji, struktur i klas. Podsumowanie.

B. Problematyka ćwiczeń laboratoryjnych

Treści merytoryczne
1. Zajęcia organizacyjne, zasady pracy, warunki zaliczenia, regulamin pracowni. Zapoznanie ze środowiskiem programistycznym, DevC++/CodeBlock, Pliki składowe projektu C++. Operacje wejścia/wyjścia w języku C++.
2. Operacje z zastosowaniem tablic jedno i dwuwymiarowych. Tablice statyczne i dynamiczne. Wskaźniki w odniesieniu do tablic.
3. Struktury i tablice struktur - wprowadzanie, wyświetlanie i inne operacje na danych.
4. Klasy – definicja pól i metod. Wskaźnik this. Hermetyzacja i dziedziczenie. Konstruktor, destruktor. Konstruktor kopiujący.
5. Przeciążenie metod i operatorów.
6. Szablony metod i klas.
7. Kolokwium. Zajęcia uzupełniające. Zaliczenie części praktycznej
8. Zajęcia wprowadzające, warunki zaliczenia
9. Projekt nr 1 – realizacja tematu wskazanego przez Prowadzącego (zastosowanie tablic dynamicznych i struktur)
10. Obrona projektów nr 1
11. Projekt nr 2 – realizacja tematu wskazanego przez Prowadzącego (zastosowanie klas, i ich mechanizmów: dziedziczenia, polimorfizmu;)
12. Obrona projektów nr 2

3.4 Metody dydaktyczne

Wykład – wykłady z prezentacją multimedialną,

Laboratorium – rozwiązywanie zadań – pisanie kodu, praca w grupach, dyskusja, projektowanie aplikacji, analiza przykładów, praca indywidualna i grupowa, dyskusja

4. METODY I KRYTERIA OCENY

4.1 Sposoby weryfikacji efektów uczenia się

Symbol efektu	Metody oceny efektów uczenia się (np.: kolokwium, egzamin ustny, egzamin pisemny, projekt, sprawozdanie, obserwacja w trakcie zajęć)	Forma zajęć dydaktycznych (w, ćw., projekt)
EK_01	Egzamin, kolokwium, sprawozdanie, wykonanie ćwiczeń, zaliczenie sprawdzianów	w., lab.
EK_02	sprawozdanie, wykonanie ćwiczeń, zaliczenie sprawdzianów	lab.
EK_03	sprawozdanie, wykonanie ćwiczeń, zaliczenie	lab.

	sprawdzianów	
EK_04	sprawozdanie, wykonanie ćwiczeń, zaliczenie sprawdzianów	lab.
EK_05	Egzamin, kolokwium	w. lab.
EK_06	Projekt, obserwacja w trakcie zajęć	lab.
EK_07	Obserwacja w trakcie zajęć	w., lab.

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

Egzamin - uzyskanie co najmniej 50% punktów z egzaminu.

Punktacja przyjęta podczas oceny egzaminu:

Ocena z przedmiotu						
Przedział punktacji	0%-50%	51%-60%	61%-70%	71%-80%	81%-90%	91%-100%
Ocena	2,0	3,0	3,5	4,0	4,5	5,0

Laboratorium – średnia z ocen z zajęć praktycznych i projektowych

Część projektowa – ocena wykonanej aplikacji i jej dokumentacji; ocena z odpowiedzi na zadane pytania. Ocenę końcową z projektu oblicza się jako średnią 3 ocen (aplikacji, dokumentacji, odpowiedzi)

Część praktyczna – ocena z odpowiedzi i/lub sprawdzianów wejściowych; dwa kolokwia pisemne, aktywność na zajęciach.

- kolokwium - ocenę pozytywną z kolokwium student uzyskuje w przypadku uzyskania minimum połowy możliwych do uzyskania punktów.
- Ocenę końcową z oblicza się na podstawie średniej ocen otrzymanych z kolokwium i odpowiedzi / sprawdzianów oraz z aktywności w proporcji 50% ocena z kolokwium i 25% ocena aktywności, 25% ocena ze sprawdzianów wejściowych, przy czym wszystkie oceny muszą być pozytywne.

5. CAŁKOWITY NAKŁAD PRACY STUDENTA POTRZEBNY DO OSIĄGNIĘCIA ZAŁOŻONYCH EFEKTÓW W GODZINACH ORAZ PUNKTACH ECTS

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe wynikające z harmonogramu studiów	36
Inne z udziałem nauczyciela akademickiego (udział w konsultacjach, egzaminie)	7
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	99
SUMA GODZIN	132

SUMARYCZNA LICZBA PUNKTÓW ECTS	5
---------------------------------------	----------

** Należy uwzględnić, że 1 pkt ECTS odpowiada 25-30 godzin całkowitego nakładu pracy studenta.*

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	Nie dotyczy
zasady i formy odbywania praktyk	

7. LITERATURA

Literatura podstawowa:

1. Paul J. Deitel, Harvey Deitel Język C. Solidna wiedza w praktyce. Wydanie VIII, Helion, 2020
2. Jerzy Grębosz Opus magnum C++11. Programowanie w języku C++, Helion 2017, t.1-3
3. S. Prata, „Język C++. Szkoła programowania. Wydanie V”, Helion 2014
4. H. Schildt C++. Sztuka programowania, Helion, 2010
5. Robert C. Martin Mistrz czystego kodu. Kodeks postępowania profesjonalnych programistów, Helion 2015

Literatura uzupełniająca:

1. Stroustrup, „Programowanie. Teoria i praktyka z wykorzystaniem C++”, Helion 2020
2. Jerzy Grębosz Opus magnum C++. Misja w nadprzestrzeń C++14/17. Tom 4. Helion 2020.
3. Scott Meyers, Skuteczny nowoczesny C++, Helion 2020
4. Jacek Galowicz C++17 STL. Receptury, Helion 2018
5. Stasiewicz, „C++. Ćwiczenia praktyczne”, Helion 2011.
6. S. Oualline, „Jak NIE programować w C++”, Mikom 2003
7. T. H. Cormen, Ch. E. Leiserson, R. L. Rivest, C. Stein, „Wprowadzenie do algorytmów”, WNT 2004
8. strony internetowe: <http://www.cplusplus.com/doc/tutorial/>

Akceptacja Kierownika Jednostki lub osoby upoważnionej