

## KARTA KURSU

Nazwa	<b>Programowanie obiektowe</b>
Nazwa w j. ang.	Object oriented programming

Koordynator	dr hab. prof. UKEN Piotr Czerski	Zespół dydaktyczny
		dr inż. M. Ciura dr hab. prof. UKEN Piotr Czerski dr W. Gwizdała mgr inż. Katarzyna Marczak
Punktacja ECTS*	st. stacjonarne: 6 st. niestacjonarne: 6	

### Opis kursu (cele kształcenia)

Celem kursu jest zapoznanie studentów z podstawami analizy, projektowania i programowania obiektowego oraz nauczenie postaw programowania w języku C++.  
Kurs prowadzony jest w języku polskim.

### Warunki wstępne

Wiedza	Student zna podstawowe zagadnienia z algorytmiki (struktury danych i proste algorytmy) oraz składnię języka C.
Umiejętności	Potrafi zapisywać podstawowe algorytmy i definiować struktury danych za pomocą języka C.
Kursy	Podstawy programowania, Programowanie proceduralne

### Efekty uczenia się

	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
Wiedza	W01: wymienia i omawia cechy obiektowego podejścia do programowania.	K_W06
	W02: ma wiedzę na temat mechanizmów pozwalających na programowanie obiektowe z zastosowaniem języka C++.	K_W06
	W03: orientuje się na poziomie podstawowym w zagadnieniach programowania generycznego w języku C++ (zna szablony klas i funkcji).	K_W06
	W04: zna składnię języka C++ i potrafi wskazać różnice między językiem C i C++.	K_W06

	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
Umiejętności	U01: potrafi zapisywać podstawowe algorytmy i struktury danych w języku C++.	K_U01
	U02: projektuje i tworzy z wykorzystaniem podstaw metodologii obiektowej proste programy w języku C++.	K_U05
	U03: kompiluje, uruchamia i znajduje błędy w napisanych przez siebie programach w języku C++.	K_U05
	U04: potrafi korzystać z wybranych funkcji, klas i szablonów z biblioteki standardowej i używać ich w pisanych przez siebie programach.	K_U05

	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
Kompetencje społeczne	K01: potrafi korzystać z różnych źródeł informacji (w tym zasobów sieciowych) do poszerzania własnej wiedzy i zdobywania nowych umiejętności.	K_K01, K_K02
	K02: wykazuje umiejętność stosowania w praktyce zdobytej wiedzy przedmiotowej i potrafi działać kreatywnie w celu rozwiązywania napotkanych problemów.	K_K01, K_K02

### Studia stacjonarne

Organizacja											
Forma zajęć	Wykład (W)	Ćwiczenia w grupach									
		A		K		L		S		P	E
Liczba godzin	20					45					

### Studia niestacjonarne

Organizacja											
Forma zajęć	Wykład (W)	Ćwiczenia w grupach									
		A		K		L		S		P	E
Liczba godzin	15					30					

### Opis metod prowadzenia zajęć

Kurs składa się z wykładu i ćwiczeń prowadzonych w formie laboratoriów. W ramach laboratoriów studenci projektują i tworzą zadane programy w języku C++, które następnie są omawiane. Poza zajęciami w formie tradycyjnej studenci biorą udział w zajęciach z wykorzystaniem platformy e-learningowej.

## Formy sprawdzania efektów uczenia się

	E – learning	Gry dydaktyczne	Ćwiczenia w szkole	Zajęcia terenowe	Praca laboratoryjna	Projekt indywidualny	Projekt grupowy	Udział w dyskusji	Referat	Praca pisemna (esej)	Egzamin ustny	Egzamin pisemny	Inne
W01	X				X			X				X	X
W02	X				X			X				X	X
W03	X				X			X				X	X
W04	X				X							X	X
U01	X				X			X				X	X
U02	X				X			X				X	X
U03	X				X			X				X	X
U04	X												
K01	X				X			X				X	X
K02	X				X			X					

Kryteria oceny	<p>Ocenę dobrą lub bardzo dobrą może uzyskać student, który:</p> <ul style="list-style-type: none"> <li>- potrafi projektować i implementować w języku C++ średniozaawansowane klasy (np. klasę macierzy, klasę reprezentującą drzewo, graf, itp.),</li> <li>- implementuje różne operatory dla zaprojektowanych klas,</li> <li>- zna i stosuje w praktyce problematykę dynamicznego zarządzania pamięcią (w tym tworzy własne operatory przypisania, konstruktory i destruktory dla klas korzystających z dynamicznego zarządzania pamięcią),</li> <li>- potrafi implementować i wykorzystywać szablony funkcji,</li> <li>- poprawnie korzysta z funkcji wirtualnych,</li> <li>- potrafi w praktyce stosować zasady poprawnego definiowania i używania obiektów stałych.</li> </ul>
----------------	--

Uwagi	
-------	--

## Treści merytoryczne (wykaz tematów)

<ol style="list-style-type: none"> <li>1. Podstawy analizy, projektowania i programowania obiektowego.</li> <li>2. Pojęcie klasy, hermetyzacja, dziedziczenie, polimorfizm.</li> <li>3. Podstawy programowania w języku C++.</li> <li>4. Dynamiczne zarządzanie pamięcią w C++ – operatory new i delete.</li> <li>5. Dostęp: publiczny, chroniony i prywatny do pól i metod.</li> <li>6. Wybrane elementy biblioteki standardowej języka C++.</li> <li>7. Przeciążanie funkcji, funkcje zaprzyjaźnione z klasą.</li> <li>8. Przeciążanie operatorów.</li> <li>9. Referencje.</li> <li>10. Stałe wartości, stałe obiekty oraz metody obiektów stałych.</li> <li>11. Tworzenie i niszczenie obiektów – konstruktory i destruktory.</li> <li>12. Wskaźnik „this” – jego znaczenie i sposób użycia.</li> <li>13. Funkcje i zmienne statyczne.</li> <li>14. Szablony funkcji</li> <li>15. Szablony klas</li> <li>16. Klasy pojemnikowe</li> <li>17. Iteratory</li> <li>18. Obsługa sytuacji wyjątkowych</li> <li>19. Szablony o zmiennej liczbie parametrów</li> <li>20. Szablon klas o dowolnej i zmiennej liczbie parametrów</li> </ol>
--

21. Atrybuty C++14, C++17
22. Wyrażenia poskładane w harmonijkę

#### Wykaz literatury podstawowej

1. Eckel B.: Thinking in C++, t.1, Helion 2002
2. Eckel B.: Thinking in C++, t. 2, Helion 2004
3. Grębosz J.: Opus Magnum C++, Helion 2020
4. Prata S.: Język C++. Szkoła programowania, Helion 2013
5. Stroustrup B.: Programowanie. Teoria i praktyka z wykorzystaniem C++, Helion 2013
6. Stroustrup B.: Język C++. Kompendium wiedzy, Helion 2014

#### Wykaz literatury uzupełniającej

1. Dattatri K.: Język C++. Efektywne programowanie obiektowe, Helion 2005
2. Josuttis N. M.: C++. Programowanie zorientowane obiektowo. Vademecum profesjonalisty, Helion 2003
3. Josuttis N. M.: C++. Biblioteka standardowa, Helion 2014
4. Lippman S., Lajoie J.: Podstawy języka C++ , WNT 2003
5. Schildt H., C++. Sztuka programowania, Helion 2004
6. Shtern V.: „Core C++. Inżynieria programowania”, Helion 2003

#### Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) studia stacjonarne

liczba godzin w kontakcie z prowadzącymi	Wykład	20
	Konwersatorium (ćwiczenia, laboratorium itd.)	45
	Pozostałe godziny kontaktu studenta z prowadzącym	15
liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	25
	Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu	
	Przygotowanie projektu lub prezentacji na podany temat (praca w grupie)	15
	Przygotowanie do egzaminu/zaliczenia	25
Ogółem bilans czasu pracy		145
Liczba punktów ECTS w zależności od przyjętego przelicznika		6

#### Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) **studia niestacjonarne**

liczba godzin w kontakcie z prowadzącymi	Wykład	15
	Konwersatorium (ćwiczenia, laboratorium itd.)	30
	Pozostałe godziny kontaktu studenta z prowadzącym	20
liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	30
	Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu	0
	Przygotowanie projektu lub prezentacji na podany temat (praca w grupie)	15
	Przygotowanie do egzaminu/zaliczenia	35
Ogółem bilans czasu pracy		145
Liczba punktów ECTS w zależności od przyjętego przelicznika		6