SYLABUS PRZEDMIOTU

1.	Nazwa przedmiotu w języku polskim oraz języku angielskim Programowanie w C++ C++ programming
2.	Dyscyplina naukowa nauki fizyczne (1 ECTS) informatyka techniczna i telekomunikacja (1 ECTS) informatyka (3 ECTS)
3.	Język wykładowy polski
4.	Jednostka prowadząca przedmiot Wydział Fizyki i Astronomii
5.	Rodzaj przedmiotu *(obowiązkowy, do wyboru)* obowiązkowy
6.	Kierunek studiów Informatyka stosowana i systemy pomiarowe
7.	Poziom studiów I stopień
8.	Rok studiów I rok
9.	Semestr *(zimowy lub letni)* letni
10.	Forma zajęć i liczba godzin wykład – 30 godzin laboratorium komputerowe – 30 godzin
11.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu Celem kursu jest wykształcenie umiejętności tworzenia programów w języku C++ na poziomie średniozaawansowanym, umożliwiającym swobodne korzystanie z bibliotek zewnętrznych (proceduralnych, obiektowych i generycznych). W szczególności celem kursu jest utrwalenie i rozszerzenie kompetencji w zakresie programowania proceduralnego w C++.Wprowadzenie podstawowych koncepcji programowania obiektowego i generycznego w C++. Zapoznanie studentów z wybraną biblioteką C++ do tworzenia aplikacji sterowanych

	zdarzeniami i posiadających nowoczesny graficzny interfejs użytkownika. Zapoznanie studentów z podstawowymi związkami między oprogramowaniem a sprzętem, na którym wykonywane są programy.	
12.	Cele kształcenia dla przedmiotu Przedstawienie klasycznych algorytmów i wykształcenie umiejętności projektowania własnych efektywnych algorytmów i struktur danych. Ocena złożoności algorytmów i poprawy ich efektywności, jeśli jest to możliwe. Umiejętność implementacji algorytmów i struktur danych w wybranym języku programowania. Umiejętność znalezienia optymalnych algorytmów dla zagadnień pojawiających się w praktyce programisty	
13.	Treści programowe Podstawowe koncepcje programowania w C++. Operatory, wyrażenia i instrukcje. Funkcje (w tym: argumenty i wartość funkcji, funkcje inline, funkcje składowe klas, polimorfizm nazw funkcji, funkcje rekurencyjne, operator jako funkcja, przeciążanie operatorów, funkcja main). Typy wbudowane (w tym arytmetyka całkowita i zmiennopozycyjna). Tablice, wskaźniki i referencje. Klasy i obiekty (w tym std::vector, std::string); dziedziczenie; hermetyzacja danych; składowe wirtualne; słowo kluczowe this. Dynamiczna alokacja pamięci. Przykładowa dynamiczna struktura danych (np. tablica dynamiczna i/lub lista pojedynczo linkowana). Strumienie. Szablony (na poziomie użytkownika). Wyjątki (na poziomie elementarnym). Wzorzec programistyczny RAII. Biblioteka STL, w tym iteratory, kontenery i algorytmy (bez szczegółów). Preprocesor, kompilator, linker. Używanie bibliotek (w tym pliki nagłówkowe, linkowanie). Wybrane narzędzia związane z C++, np. środowisko typu QtCreator, debuger; kompilacja programów za pomocą mechanizmu Makefile lub podobnego (np. cmake).	
14.	Zakładane efekty uczenia się Student: – zna i rozumie zasady programowania strukturalnego oraz obiektowego w C++; – zna składnię języka C++ w zakresie potrzebnym do tworzenia programów proceduralnych i obiektowych oraz używania bibliotek generycznych; – zna zasady kompilacji programów w C/C++ (preprocesor, kompilator, linker), w tym programów wykorzystujących biblioteki zewnętrzne; – zna podstawowe zasady organizacji pamięci operacyjnej (stos/sterta, inicjalizacja i	Symbole odpowiednich kierunkowych efektów uczenia się I1_W04, I1_W05 I1_U08 I1_K03

	destrukcja obiektów, zmienne statyczne / automatyczne); – zna sposoby organizacji i przekazywania danych między różnymi fragmentami programu oraz sposoby organizacji kodu w funkcji; – zna podstawy programowania obiektowego (dziedziczenie, hermetyzacja danych, funkcje wirtualne); – specyfikę arytmetyki stałoprzecinkowej i zmiennoprzecinkowej; – wybrane elementy biblioteki standardowej C++17; – potrafi pisać i kompilować proste programy i projekty w C++, w tym projekty korzystające z biblioteki standardowej i bibliotek zewnętrznych; – potrafi tworzyć i używać w kodzie różne rodzaje podprogramów (funkcje swobodne i metody klas; metody wirtualne; przeciążone operatory); – potrafi posługiwać się wybranymi klasami biblioteki standardowej C++ (np. vector, string); – potrafi posługiwać się wybranym zintegrowanym środowiskiem programistycznym umożliwiającym tworzenie programów i projektów w języku C++; – potrafi kompilować projekt za pomocą mechanizmu CMake (lub podobnego); – potrafi debugować programy w języku C++; – samodzielnie proponuje rozwiązanie postawionych zadań programistycznych, w razie potrzeby korzysta z innych bibliotek i narzędzi języka C++ niż wskazane na zajęciach.	
15.	Literatura obowiązkowa i zalecana Literatura obowiązkowa: Prezentacje z wykładu (udostępniane po kolejnych wykładach) Notatki z wykładu: https://github.com/zkoza/cpp-issp Literatura zalecana: Jerzy Grębosz, Opus magnum C++ 11. Programowanie w języku C++, Helion 2017 Portal Stack Overflow (stackoverflow.com) Portal C++ Reference (cppreference.com) Materiały na YouTube, np. seria kursów C++ „The Cherno”	

16.	Metody weryfikacji zakładanych efektów uczenia się: ciągła kontrola postępów realizacji bieżących zagadnień i zadań zawartych na listach egzamin ustny; może być poprzedzony pisemnym testem	
17.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu: ciągła kontrola obecności i postępów w zakresie tematyki zajęć, pozytywne oceny list zadań egzamin (pisemny i/lub ustny)	
	Nakład pracy studenta wyrażony w godzinach zajęć oraz punktach ECTS	liczba godzin przeznaczona na zrealizowanie danego rodzaju zajęć
	zajęcia (wg planu studiów) z prowadzącym: – wykład: – laboratorium:	 30 30
	praca własna studenta (w tym udział w pracach grupowych): – przygotowanie do zajęć: – przygotowanie do egzaminu:	 60 15
	Łączna liczba godzin zajęć	135
	Liczba punktów ECTS (*jeśli jest wymagana*)	5