KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Podstawy programowania [N1Mech2>PoPr]

Przedmiot

Kierunek studiów
Mechatronika

Rok/Semestr
1/1

Studia w zakresie (specjalność)
–

Profil studiów
ogólnoakademicki

Poziom studiów
pierwszego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
niestacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
8

Laboratorium
16

Inne
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

4,00

Koordynatorzy

Wykładowcy

Wymagania wstępne

Wiedza:Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę w zakresie informatyki, matematyki, sprzętu komputerowego, obsługi komputera, systemu operacyjnego Windows oraz podstawowego oprogramowania użytkowego. Powinien również posiadać umiejętność pozyskiwania informacji ze wskazanych źródeł oraz mieć gotowość do podjęcia współpracy w ramach zespołu.

Cel przedmiotu

Przekazanie podstawowej wiedzy z zakresu informatyki, budowy i zasady działania mikrokomputerów, opanowanie umiejętności opracowywania prostych algorytmów oraz podstaw programowania strukturalnego i obiektowego w języku C++.

Przedmiotowe efekty uczenia się

Wiedza:
Ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną w zakresie kluczowych dla obszaru elektromobilności zagadnień informatyki, w tym programowania oraz wykorzystania narzędzi informatycznych w modelowaniu, symulacji i projektowaniu.

Umiejętności:
Potrafi korzystać ze źródeł literaturowych, integrować pozyskane informacje, oceniać je oraz dokonywać

ich interpretacji i wyciągać wnioski, w celu rozwiązania złożonych i nietypowych problemów w obszarze mechatroniki. Potrafi posłużyć się właściwie dobranymi metodami i narzędziami, w tym zaawansowanymi technikami informacyjno-komunikacyjnymi, a także opracować proste aplikacje, w celu przeprowadzenia symulacji, analizy i projektowania układów właściwych dla kierunku studiów.

Kompetencje społeczne:
Rozumie znaczenie podnoszenia kompetencji zawodowych, osobistych i społecznych; ma świadomość, że wiedza i umiejętności w obszarze mechatroniki szybko ewoluują. Rozumie znaczenie wiedzy w rozwiązywaniu problemów z zakresu mechatroniki; jest świadomy konieczności wykorzystania wiedzy ekspertów podczas rozwiązywania zadań inżynierskich w zakresie wykraczającym poza własne kompetencje.

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocena wiedzy i umiejętności na pisemnym kolokwium zaliczeniowym o charakterze łączonym testowym i problemowym. Uzyskiwanie punktów dodatkowych za aktywność podczas wykładów, a szczególnie za: przygotowywanie odpowiedzi na pytania i zadania problemowe podawane przez wykładającego, staranność estetyczną zadań opracowywanych w ramach nauki własnej, aktywność na wykładach oraz laboratorium przy rozwiązywaniu bieżących zadań problemowych.

Treści programowe

Historia informatyki, obszary jej zastosowań i badań. Systemy liczbowe, stało i zmiennopozycyjna reprezentacja liczb, kodowanie informacji, podstawy działania układów cyfrowych, struktura systemu komputerowego, magistrale, ogólna charakterystyka procesorów, pamięci RAM i ROM. Systemy operacyjne, sieci komputerowe, praca komputerów w sieci, zagadnienia bezpieczeństwa w sieciach komputerowych. Internet, intranet. Algorytmy i struktury danych. Wybrane algorytmy rozwiązywalnych analityczne problemów z matematyki, fizyki oraz algorytmy problemu sortowania. Języki programowania. Język programowania C++. Programowanie strukturalne. Programowanie w środowisku MS Visual C++.
Laboratorium: podstawy programowania w języku C++ (składnia, opracowanie prostych algorytmów i programów).

Tematyka zajęć

Wykład:
I. Podział języków programowania. Kompilacja, interpretacja. Języki niskiego i wysokiego poziomu. Operacje arytmetyczne, operatory logiczne, funkcje standardowe, instrukcje warunkowe, instrukcje sekwencyjne
II. Funkcje. Definiowanie funkcji. Zwracanie rezultatu przez funkcję. Przesyłanie argumentów do funkcji przez wartość i przez referencję. Tablice, typy wyliczeniowe, wektory.
III. Rekurencja. Definicja i zasady działania rekurencji. Przypadek podstawowy i przypadek rekurencyjny. Wskaźniki. Definiowanie wskaźników. Dynamiczna alokacja pamięci. Przekazywanie argumentów do funkcji przez wskaźnik.
IV. Kolokwium zaliczeniowe.
Laboratoria:
I. Omówienie środowiska programistycznego Visual Studio
• Omówienie regulaminu pracowni komputerowej
• Omówienie regulaminu BHP oraz warunków zaliczenia przedmiotu
• Omówienie i konfiguracja środowiska Visual Studio, Konsola
• Dyrektywy preprocesora - np. #using, #pragma, #define
• Omówienie składni języka C++, typy zmiennych, deklaracja zmiennych, stałe - programowanie strukturalne
II. Operacje arytmetyczne, operatory logiczne, funkcje standardowe, instrukcje warunkowe
• Podstawowe operatory arytmetyczne i logiczne - programy ilustrujące działanie operatorów i funkcji standardowych
• Instrukcje sterujące typu: if, if...else, switch, break, continue, goto
• Programy testujące działanie powyższych instrukcji z wykorzystaniem operatorów
III. Oprogramowywanie sekwencji instrukcji

• Instrukcje sterujące typu: while, do..while, for, foreach (Range-based for)
• Wybrane programy testujące instrukcje sekwencyjne (Obliczanie sumy szeregu, sumowanie n liczb nieparzystych, szukanie wartości maksymalnej ciągu liczb zadanego z klawiatury, wybór z ciągu liczb zadanego z klawiatury liczb podzielnych przez 3, poszukiwanie liczby doskonałej, poszukiwanie liczby pierwszej)
IV. Funkcje
• Definiowanie funkcji
• Zwracanie rezultatu przez funkcję
• Przesyłanie argumentów do funkcji przez wartość i przez referencję
• Opracowanie przykładowych funkcji (największy wspólny dzielnik, najmniejsza wspólna wielokrotność, zamiana liczby dziesiętnej na dwójkową, obliczanie silni, obliczanie wartości średniej, itp.)
V. Tablice, typy wyliczeniowe, wektory
• Definiowanie tablic, tablice statyczne jedno i wielowymiarowe
• Obsługa tablic
• Definiowanie wektorów, wektory jedno i wielowymiarowe
• Obsługa wektorów
VI. Wskaźniki
• Definiowanie wskaźników
• Wyrażenie null i nullptr
• Tablice dynamiczne
• Dynamiczna alokacja pamięci
• Przekazywanie argumentów do funkcji przez wskaźnik
VII. Struktury i unie
• Definiowanie struktur
• Definiowanie unii
• Wyrażenie sizeof
• Operacje na strukturach i uniach
VIII. Kolokwium zaliczeniowe

Metody dydaktyczne

Zastosowane metody kształcenia: a) wykład z prezentacją multimedialną (w tym: rysunki, zdjęcia, animacje, dźwięk, filmy) uzupełniany przykładami podawanymi na tablicy, b) wykład prowadzony w sposób interaktywny z formułowaniem pytań do grupy studentów lub do wskazywanych konkretnych studentów, c)uwzględnia się aktywność studentów w czasie zajęć przy wystawianiu oceny końcowej, c) teoria przedstawiana w ścisłym powiązaniu z praktyką i z aktualną wiedzą studentów. Laboratorium: demonstracje, samodzielne wykonywanie zadań programistycznych (obliczeniowych).

Literatura

Podstawowa:
1. Jerzy Grębosz - "Opus magnum C++11. Programowanie w języku C++" (2017)
2. Jerzy Grębosz - "Opus magnum C++. Misja w nadprzestrzeń C++14/17" (2023)
3. Stephen Prata - "Język C++. Szkoła programowania. Wydanie VI" (2012)
4. Bjarne Stroustrup - "Programowanie. Teoria i praktyka z wykorzystaniem C++" (2011)

Uzupełniająca:
1. Scott Meyers - "Skuteczny nowoczesny C++" (2016)
2. Bruce Eckel - "Thinking in C++. Edycja polska. Tom 1" (2002)
3. Bruce Eckel, Chuck Allison - "Thinking in C++. Edycja polska. Tom 2" (2003)

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	100	4,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	24	1,00
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	76	3,00