



KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu

Podstawy informatyki

Przedmiot

Kierunek studiów

Automatyka i robotyka

Studia w zakresie (specjalność)

Systemy automatyki i robotyki

Poziom studiów

pierwszego stopnia

Forma studiów

niestacjonarne

Rok/semestr

1/1

Profil studiów

ogólnoakademicki

Język oferowanego przedmiotu

polski

Wymagalność

obligatoryjny

Liczba godzin

Wykład

18

Laboratoria

18

Inne (np. online)

0

Ćwiczenia

0

Projekty/seminaria

0

Liczba punktów ECTS

5

Wykładowcy

Odpowiedzialny za przedmiot/wykładowca:

dr inż. Rafał Kapela

email: Rafal.Kapela@put.poznan.pl

tel. 61 6652184

Wydział Automatyki, Robotyki i Elektrotechniki

Odpowiedzialny za przedmiot/wykładowca:

dr inż. Jakub Kołota

email: Jakub.Kolota@put.poznan.pl

tel. 61 6652751

Wydział Automatyki, Robotyki i Elektrotechniki

ul. Piotrowo 3, 60-965 Poznań

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z zakresu programowania strukturalnego oraz sprzętu komputerowego i jego obsługi. Powinien posiadać umiejętność rozwiązywania



podstawowych problemów w obszarze modelowania algorytmów, programowania funkcyjnego oraz umiejętność pozyskiwania informacji ze wskazanych źródeł. Powinien również rozumieć konieczność poszerzania swoich kompetencji jak również być gotowym do podjęcia współpracy w ramach zespołu. Ponadto w zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu

Cel modułu kształcenia:

1. Zapoznanie z metodologią i zasadami programowania obiektowego wykorzystując język programowania C++.
2. Rozwijanie u studentów umiejętności rozwiązywania problemów w obszarze modelowania i implementacji systemów informatycznych. Studenci uczyć się przeprowadzać symulację i analizę działania obiektowych programów informatycznych oraz planować i dokumentować wykonaną pracę informatyczną.
3. Kształtowanie u studentów umiejętności programistycznych. Kreowanie świadomości konieczności profesjonalnego podejścia do zagadnień technicznych, skrupulatnego zapoznania się z dokumentacją systemów informatycznych typu UML. Student uczy się wyznaczać cele i określać priorytety prowadzące do realizacji zadania poprzez obiektową implementację kodu.

Przedmiotowe efekty uczenia się

Wiedza

1. ma uporządkowaną w zaawansowanym stopniu wiedzę w zakresie wybranych algorytmów i struktur danych oraz metodyki i technik programowania proceduralnego i obiektowego; - [KW_8]
2. ma uporządkowaną wiedzę w zakresie architektur komputerów, systemów i sieci komputerowych oraz systemów operacyjnych w tym systemów operacyjnych czasu rzeczywistego; - [KW_9]

Umiejętności

1. potrafi pozyskiwać informacje z literatury, baz danych i innych źródeł także w wybranym języku obcym; - [KU_1]

Kompetencje społeczne

- FORMTEXT1.jest gotów do krytycznej oceny posiadanej wiedzy;rozumie potrzebę i zna możliwości ciągłego doksztalcania się – podnoszenia kompetencji zawodowych, osobistych i społecznych, potrafi inspirować i organizować proces uczenia się innych osób; - [K_K1]
- 2.posiada świadomość odpowiedzialności za pracę własną oraz gotowość podporządkowania się zasadom pracy w zespole i ponoszenia odpowiedzialności za wspólnie realizowane zadania; potrafi kierować małym zespołem, wyznaczać cele i określać priorytety prowadzące do ich realizacji; jest gotów do odpowiedzialnego pełnienia ról zawodowych; - [K_K3]

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca:



a) w zakresie wykładów: na podstawie odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach,

b) w zakresie laboratoriów / ćwiczeń: na podstawie oceny bieżącego postępu realizacji zadań,

Ocena podsumowująca:

a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez:

i. ocenę wiedzy i umiejętności wykazanych na egzaminie pisemnym o charakterze problemowym złożonym z 5 pytań spośród 40 pytań przedstawionych na ogólnej liście pytań, udostępnionej wcześniej studentom.

Zasady oceniania:

5,0 - powyżej 90% punktów z egzaminu(W);średnia ocen z ćwiczeń lab. powyżej 4,75 (L)

4,5 - 80%-90% punktów z egzaminu (W);średnia ocen z ćwiczeń lab. 4,25-4,75 (L)

4,0 - 70%-80% punktów z egzaminu (W);średnia ocen z ćwiczeń lab. 3,75-4,25 (L)

3,5 - 60%-70% punktów z egzaminu(W);średnia ocen z ćwiczeń lab. 3,25-3,75 (L)

3,0 - 50%-60% punktów z egzaminu(W);średnia ocen z ćwiczeń lab. 2,75-3,25 (L)

2,0 - poniżej 50% punktów z egzaminu(W);średnia ocen z ćwiczeń lab. poniżej 2,75 (L)

ii. omówienie wyników zaliczenia,

b) w zakresie laboratoriów / ćwiczeń weryfikowanie założonych efektów kształcenia realizowane jest przez:

a. ocenę przygotowania studenta do poszczególnych sesji zajęć laboratoryjnych (sprawdzian wejściowy)

b. ocenę przeprowadzonego ćwiczenia laboratoryjnego (sprawozdanie)

Treści programowe

Pogram wykładu obejmuje następujące zagadnienia:

W trakcie semestru prowadzący przedmiot kompleksowo omawia podczas wykładu semantykę języka C++ w ujęciu obiektowym. Każdy z wykładów poświęcony jest innemu zagadnieniu i prezentuje rozwiązania w postaci gotowych implementacji. Studenci otrzymują materiały wykładowe w postaci plików zawierających gotowe pliki źródłowe wraz z kodem omawianym podczas danego wykładu. Treści wykładu uzupełnione są o dwa wykłady poświęcone modelowaniu UML, podczas których student zapoznaje się ze sposobem tworzenia dokumentacji i notacją UML (diagram klas i obiektów, diagram czynności, diagram sekwencji, diagram przypadków użycia, itd.) Ćwiczenia laboratoryjne przygotowane zostały w postaci plików pdf, które stanowią zadania do samodzielnej realizacji podczas



zajęć. Student wykorzystuje w tym celu konto studenta na portalu e-learningowym Katedry Inżynierii Komputerowej (<http://dydaktyka.cce.put.poznan.pl/moodle>) i pobiera dokument z treścią zadania. Podczas zajęć, prowadzący przypomina w skrócie treści programowe związane z danym ćwiczeniem (omawiane wcześniej przez prowadzącego przedmiot podczas wykładu). Kolejne laboratoria obejmują następujące zagadnienia programowania obiektowego:

- Język C++ - powtórka z semestru pierwszego i wyrównanie poziomu podstawowego grupy laboratoryjnej
- Klasa i obiekt – kwantyfikatory sekcji prywatnej oraz publicznej klasy
- Konstruktor (konstruktor kopiujący), destruktor, składniki statyczne klasy (modyfikator static)
- Wskaźniki, tablice dynamiczne (poruszanie się po pamięci i dynamiczne jej przydzielanie/zwalnianie)
- Dziedziczenie (kwantyfikator protected, zagadnienia dziedziczenia wielopoziomowego) oraz mechanizm funkcji zaprzyjaźnionych
- Metody wirtualne oraz zagadnienie polimorfizmu
- Abstrakcja klasy
- Przeładowanie funkcji i operatorów
- Rzutowanie i konwersja typów
- STL, kontener listy (list) oraz kontener tablicy (vector)
- Interfejs graficzny aplikacji okienkowej (dwa spotkania laboratoryjne)

Metody dydaktyczne

Metody dydaktyczne:

1. wykład: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy oraz pokazami multimedialnymi i demonstracjami wykorzystującymi m.in. program CodeBlocks i kompilator gcc oraz g++
2. ćwiczenia praktyczne, implementacja zadań i algorytmów zdefiniowanych w opracowaniach powierzonych studentowi, dyskusja nad złożonością i optymalizacją kodu w trakcie zajęć (wszelkie materiały opracowane elektronicznie i osadzone na platformie e-learningowej)

Literatura

Podstawowa

- 1.M. Dadaj, Programowane zorientowane obiektowo. Wyd. Helion 2005.
2. B. Stroustrup, Język C++, wydanie V, WNT, Warszawa 2000



3. Jerzy Grębosz, Symfonia C ++ Standard, Editions 2000, Kraków 2005

4. Zbigniew Koza, Język C++. Pierwsze starcie, Helion, Gliwice, 2008

Uzupełniająca

1. B. Eckel, Thinking In C++, Edycja polska, Wydawnictwo Helion.

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
łączy nakład pracy	125	5,0
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	36	2,0
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwium/egzaminu, wykonanie projektu) ¹	89	3,0

¹ niepotrzebne skreślić lub dopisać inne czynności

