KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Podstawy programowania [S1Bioinf1>PROG]

Przedmiot

Kierunek studiów
Bioinformatyka

Rok/Semestr
1/1

Studia w zakresie (specjalność)
–

Profil studiów
ogólnoakademicki

Poziom studiów
pierwszego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
30

Laboratorium
30

Inne (np. online)
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

6,00

Koordynatorzy

dr inż. Marcin Radom
marcin.radom@put.poznan.pl

Wykładowcy

dr inż. Marcin Radom
marcin.radom@put.poznan.pl

mgr inż. Bartłomiej Szawulak
bartlomiej.szawulak@put.poznan.pl

Wymagania wstępne

Student rozpoczynający ten moduł powinien posiadać podstawową wiedzę o działaniu komputera, o podstawowej terminologii oraz umiejętność obsługi programów w języku angielskim.

Cel przedmiotu

1. Przekazanie studentom podstawowej wiedzy o językach programowania na bazie języka C / C++ w zakresie podstawowym i średnio-zaawansowanym. 2. Zapoznanie studentów z podstawowymi elementami języków programowania oraz elementami programowania strukturalnego. 3. Rozwinięcie u studentów umiejętności rozwiązywania problemów algorytmicznych. 4. Wykształcenie u studentów umiejętności rozpisywania postawionego problemu na kolejne kroki elementarne możliwe do zakodowania w języku programowania. 5. Zapoznanie studentów z ideą zintegrowanych środowisk deweloperskich (IDE) na bazie MS Visual Studio.

Przedmiotowe efekty uczenia się

Wiedza:
W wyniku przeprowadzonych zajęć student:
1. Zna zagadnienia z zakresu algorytmów (np. pętle, instrukcje warunkowe, itd.) i struktur danych (listy, kolejki, drzewa) oraz elementy teorii złożoności obliczeniowej.
2. Zna zasady programowania strukturalnego w języku ANSI C i podobnych.

Umiejętności:
W wyniku przeprowadzonych zajęć student:
1. Potrafi pozyskiwać informacje z literatury oraz innych właściwie dobranych źródeł (serwisy internetowe poświęcone językom programowania i zagadnieniom pochodnym), także w języku angielskim.
2. Projektuje i tworzy oprogramowanie komputerowe zgodnie z zadaną specyfikacją, używając właściwych metod, technik i narzędzi.
3. Samodzielnie zdobywa wiedzę i podnosi swoje kwalifikacje, niezbędne z uwagi na często zmieniające się style i podejścia do programowania.

Kompetencje społeczne:
Zaliczenie przedmiotu oznacza, że student:
1. Rozumie potrzebę uczenia się przez całe życie i podnoszenia swoich kompetencji z uwagi na ciągłe rozwijanie języków programowania o wciąż nowe i przydatne funkcjonalności.

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca
a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez:
• odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach
b) w zakresie laboratoriów weryfikowanie założonych efektów kształcenia realizowane jest przez:
• ocenę i „obronę” zrealizowanych przez studenta ćwiczeń laboratoryjnych
• ocenę wiedzy i umiejętności związanych z realizacją zadań laboratoryjnych poprzez dwa sprawdziany w semestrze
Ocena podsumowująca
a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez:
• ocenę wiedzy i umiejętności wykazanych na egzaminie pisemnym w formie testu wielokrotnego wyboru oraz nielicznych pytań otwartych. Egzamin składa się z 10-20 pytań o łącznej wartości 20-40 punktów rozdzielonych w zależności od stopnia trudności pytania. Ocenę pozytywną studenci otrzymują po zdobyciu minimum połowy punktów.
• omówienie wyników egzaminu
b) w zakresie laboratoriów weryfikowanie założonych efektów kształcenia realizowane jest przez:
• w trakcie realizacji zajęć laboratoryjnych studenci mają do napisania do dwóch kolokwiów, dotyczących zagadnień programistycznych omawianych na wykładzie. Dodatkowo muszą samodzielnie napisać programy zaliczeniowe zgodnie ze specyfikacją omówioną na zajęciach.
• Studentka/student otrzymują ocenę pozytywną z laboratorium jeżeli napiszą pozytywnie oba kolokwia oraz poprawnie zaimplementują programy zaliczeniowe, przy wystawieniu oceny końcowej uwzględnia się wszystkie wymienione elementy oraz zadania domowe.
Aktywność podczas zajęć premiowana jest dodatkowymi punktami, w szczególności za:
• efektywność zastosowania zdobytej wiedzy podczas pisania zadanego programu w sposób wykraczający poza minimum określone w specyfikacji.

Treści programowe

Program wykładu obejmuje zagadnienia dotyczące szeroko rozumianego programowania strukturalnego na bazie języka C / C++. W ramach pierwszego wykładu studenci są dodatkowo zapoznawani z najbardziej podstawowymi zagadnieniami z informatyki niezbędnymi w programowaniu w ramach jakiegokolwiek języka (jak na przykład organizacja pamięci na najniższych poziomie teoretycznym – bity, bajty, słowa, różnice między algorytmem a programem, wstęp opisujący popularne środowiska programistyczne jak Eclipse, Visual Studio, itp.). Od drugiego spotkania zaczyna się ciąg wykładów poświęconych podstawowym elementom języka C / C++ czy także dowolnego innego, jak pętle, instrukcje warunkowe, zmienne, funkcje/procedury/metody, struktury i obiekty, itp. Po omówieniu podstaw niezbędnych do programowania następuje ciąg wykładów poświęconych bardziej teoretycznym aspektom algorytmiki.

Wykłady te dotyczą takich zagadnień, jak grafy, struktury danych jak listy, kolejki czy drzewa binarne, różne algorytmy sortowania oraz ich porównanie z punktu widzenia złożoności czasowej oraz pamięciowej. Ćwiczenia laboratoryjne prowadzone są w formie piętnastu dwugodzinnych zajęć odbywających się w laboratorium komputerowym. Pierwsze zajęcia przeznaczone są na zapoznanie studentów z zasadami użytkowania laboratorium i zaliczania ćwiczeń. Program zajęć laboratoryjnych obejmuje następujące zagadnienia:
- ćwiczenie i utrwalanie wiedzy z wykładów dotyczących różnych elementów języka C/C++,
- pisanie samodzielnych, prostych programów na bazie przećwiczonej wiedzy,
- rozwijanie w ramach zajęć laboratoryjnych większego programu służącego ilustracji takich pojęć, jak przejrzystość kodu, spójny styl pisania, podział programu na różne jednostki funkcjonalne, itp.,
- ćwiczenia dotyczące trudniejszych zagadnień niezbędnych do opanowania przed samodzielnym napisaniem programów zaliczeniowych.

Metody dydaktyczne

1. Wykład: prezentacja multimedialna oraz dodatkowe przykłady podawane na tablicy w miarę potrzeb.
2. Ćwiczenia laboratoryjne: rozwiązywanie zadań, ćwiczenia praktyczne, praca w zespole.

Literatura

Podstawowa
1. B.W.Kernighan, D.M. Ritchie, Język ANSI C, Wydawnictwa Naukowo-Techniczne, seria Klasyka Informatyki
2. C.L. Tondo, S.E. Gimpel, Język ANSI C – ćwiczenia i rozwiązania, Wydawnictwa Naukowo-Techniczne, seria Klasyka Informatyki
Uzupełniająca
1. http://pl.wikibooks.org/wiki/C - darmowa książka elektroniczna na licencji GNU GPL

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	150	6,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	80	3,00
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	70	3,00