KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Programowanie deklaratywne [S1Inf1>PDEK]

Przedmiot

Kierunek studiów
Informatyka

Rok/Semestr
1/2

Studia w zakresie (specjalność)
–

Profil studiów
ogólnoakademicki

Poziom studiów
pierwszego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
16

Laboratorium
16

Inne
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

2,00

Koordynatorzy

dr inż. Artur Michalski
artur.michalski@put.poznan.pl

Wykładowcy

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z logiki obliczeniowej oraz teorii mnogości. Powinien posiadać umiejętność rozwiązywania podstawowych problemów algorytmicznych opisanych za pomocą formalnego aparatu matematycznego oraz umiejętność pozyskiwania informacji na temat sposobów ich rozwiązywanie ze wskazanych źródeł.

Cel przedmiotu

Przekazanie studentom podstawowej wiedzy dotyczącej deklaratywnego paradygmatu programowania na przykładzie języka Prolog. Rozwijanie u studentów umiejętności rozwiązywania problemów algorytmicznych z wykorzystaniem języka programowania wysokiego poziomu opartego na mechanizmach rekurencji oraz przeszukiwaniu przestrzeni rozwiązań. Kształtowanie u studentów świadomości różnic między paradygmatem deklaratywnym a pozostałymi paradygmatami programowania.

Przedmiotowe efekty uczenia się

Wiedza:
1. ma uporządkowaną, podbudowaną teoretycznie wiedzę ogólną w zakresie języków i paradygmatów programowania oraz interpretacji proceduralnej i deklaratywnej kodu programu. [K1st_W4]

2. posiada wiedzę podstawową nt. funkcjonowania automatycznych mechanizmów sterowania wnioskowaniem w językach deklaratywnych opartych na logice obliczeniowej. [K1st_W6]
3. zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu przetwarzania rekurencyjnych struktur danych i ich implementacji w deklaratywnych językach programowania. [K1st_W7]
4. dysponuje wiedzą elementarną w zakresie form i metod metaprogramowania, przetwarzania symbolicznych form reprezentacji danych i wiedzy. [K1st_W7]

Umiejętności:
1. potrafi ocenić przydatność języków, metod i narzędzi służących do rozwiązywania zadań typowych dla informatyki, oraz wskazywać właściwe obszary zastosowań metod i narzędzi programistycznych paradygmatu deklaratywnego. [K1st_U10]
2. posiada umiejętność formułowania algorytmów i ich programowania w zakresie zadań o charakterze symbolicznym i tekstowym. [K1st_U11]
3. opracować i zaimplementować rozwiązanie problemu programistycznego w kategoriach paradygmatu deklaratywnego z zastosowaniem prostych i złożonych (rekurencyjnych) struktur danych. [K1st_U11]

Kompetencje społeczne:
1. rozumie potrzebę ciągłego pogłębiania swojej wiedzy i doskonalenia umiejętności w zakresie narzędzi programistycznych i rozwijających się paradygmatów programowania. [K1st_K1]
2. potrafi myśleć i działać w sposób kreatywny, pozostając otwartym na pozainformatyczne aspekty działalności inżyniera-informatyka związane z konstruowaniem oprogramowania. [K1st_K3]

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocenę wiedzy nabytej w ramach wykładu w formie testu wielokrotnego wyboru, składającego się z 20 pytań o łącznej wartości 20 punktów. Skala ocena zgodna ze zaktualizowanym regulaminem studiów. Ocenę wiedzy i umiejętności związanych z realizacją zadań laboratoryjnych poprzez kolokwium na zakończenie semestru, polegające na rozwiązaniu określonego problemu programistycznego; zadanie będące przedmiotem kolokwium ma charakter wieloetapowego algorytmu, a uzyskanie oceny pozytywnej możliwe jest po poprawnym zaimplementowaniu ponad połowy wszystkich jego kroków, składających się na kompletne rozwiązanie końcowe oraz wykazaniu zdolności do konstruowania własnych (oryginalnych) składowych rozwiązania; część kroków poprawnego rozwiązania obejmuje metody prezentowane w trakcie semestru, jednak student musi wykazać się przede wszystkim samodzielnością w rozwiązywaniu zadania, aby uzyskać końcowe zaliczenie; kryterium to jest traktowane jako kluczowe przy ocenie pracy zaliczeniowej.

Treści programowe

Deklaratywna i proceduralna interpretacja programu prologowego
Rekurencyjna definicja reguły
Reprezentacja danych w Prologu
Mechanizm uzgadniania termów
Związki języka Prolog z logiką formalną
Reprezentacja list w Prologu
Operacje arytmetyczne w Prologu
Sterowanie mechanizmem nawrotów
Negacja przez niepowodzenie
Metapredykaty w Prologu
Efektywność programów prologowych

Tematyka zajęć

Wykład 1: Wprowadzenie do języka Prolog
- Przykład prostego programu w języku Prolog: Relacje pokrewieństwa
- Rozszerzanie programu przez wprowadzanie reguł prologowych
- Rekurencyjna definicja reguły prologowej
- Zasady generowania odpowiedzi na postawione pytania
- Deklaratywna i proceduralna interpretacja programu prologowego

Wykład 2: Budowa składniowa i interpretacja programów prologowych
- Reprezentacja danych w Prologu
- Mechanizm uzgadniania termów
- Formalna interpretacja deklaratywna programu prologowego
- Formalna interpretacja proceduralna programu prologowego
- Przykład interpretacji programu: Problem małpy i banana
- Porządek kauzul prologowych i celów
- Związki języka Prolog z logiką formalną
Wykład 3: Listy, operatory i operacje arytmetyczne
- Reprezentacja list w Prologu
- Wybrane operacje na listach w Prologu
- Notacja operatorów
- Operacje arytmetyczne w Prologu
Wykład 4: Sterowanie mechanizmem nawrotów
- Mechanizm odcięć (ang. cuts)
- Przykłady wykorzystywania odcięć w programie prologowym
- Negacja przez niepowodzenie
- Problemy związane z zastosowaniem odcięć i negacji w Prologu
Wykład 5: Predefiniowane (systemowe) predykaty prologowe - metapredykaty
- Sprawdzanie typu termów
- Kompozycja i dekompozycja termów: =.., functor, arg, name
- Różne rodzaje operacji równości w Prologu
- Manipulacja bazą danych w Prologu
- Manipulowanie przepływem sterowania w Prologu
- Predykaty: bagof, setof i findall
Wykład 6: Operacje wejścia/wyjścia w Prolog
- Operacje na plikach
- Przetwarzanie plików termów
- Manipulowanie danymi znakowymi
- Kompozycja i dekompozycja atomów
- Wczytywanie programów prologowych: consult i reconsult
Wykład 7: Styl i technika programowania w Prologu
- Ogólne zasady poprawnego programowania w Prologu
- Jak interpretować programy prologowe?
- Styl programowania
- Efektywność programów prologowych - sposoby poprawy
Program ćwiczeń laboratoryjnych odpowiada tematycznie programowi wykładu z wyjątkiem operacji wejścia/wyjścia.

Metody dydaktyczne

1. Wykład: prezentacja multimedialna w tym także przykłady rozwiązań zadań
2. Ćwiczenia laboratoryjne: realizacja praktycznych zadań programistycznych o rosnącym stopniu trudności

Literatura

Podstawowa
1. Prolog. Programowanie, W.F. Clocksin, C.S. Mellish, Helion, Gliwice, 2003
2. Logika w rozwiązywaniu zadań, R.A. Kowalski, WNT, Warszawa, 1989
Uzupełniająca
1. Prolog - programming for AI, I. Bratko, Addison-Wesley, 1990
2. Micro-Prolog, K.L. Clark, F.G. McGabe, WNT, Warszawa, 1985
3. Prolog, F. Kluźniak, S. Szpakowicz, WNT, Warszawa, 1983

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	60	2,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	32	1,00
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	28	1,00