KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu **Języki formalne i kompilatory**		Kod **1010334531010330115**
Kierunek studiów **Informatyka**	Profil kształcenia (ogólnoakademicki, praktyczny) **(brak)**	Rok / Semestr **2 / 3**
Ścieżka obieralności/specjalność **-**	Przedmiot oferowany w języku: **polski**	Kurs (obligatoryjny/obieralny) **obligatoryjny**
Stopień studiów: **I stopień**	**Forma studiów** (stacjonarna/niestacjonarna) **niestacjonarna**	
Godziny Wykłady: **12** Ćwiczenia: **8** Laboratoria: **8** Projekty/seminaria: **-**		Liczba punktów **4**
Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny) **(brak)**	(ogólnouczelniany, z innego kierunku) **(brak)**	
Obszar(y) kształcenia i dziedzina(y) nauki i sztuki **nauki techniczne**		**Podział ECTS (liczba i %)** **4 100%**

Odpowiedzialny za przedmiot / wykładowca:

dr inż. Jolanta Cybulka
email: jolanta.cybulka@put.poznan.pl
tel. 0-61 6653724
Wydział Elektryczny
ul. Piotrowo 3A 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	1) ma podstawową wiedzę w zakresie matematyki, obejmującą algebrę, analizę, logikę, probabilistykę oraz elementy matematyki dyskretnej i stosowanej 2)ma uporządkowaną i podbudowaną teoretycznie wiedzę w zakresie podstawowych algorytmów i ich analizy, technik projektowania algorytmów, abstrakcyjnych struktur danych i ich implementacji, problemów obliczeniowo trudnych
2	**Umiejętności:**	1) potrafi pozyskiwać informacje z literatury, baz danych i innych źródeł; potrafi integrować uzyskane informacje, dokonywać ich interpretacji, a także wyciągać wnioski oraz formułować i uzasadniać opinie 2)potrafi posłużyć się środowiskami i platformami programistycznymi do pisania, wykonywania i testowania prostych programów kodowanych w językach programowania imperatywnego, obiektowego i deklaratywnego
3	**Kompetencje społeczne**	ma świadomość odpowiedzialności za pracę własną oraz gotowość podporządkowania się zasadom pracy w zespole i ponoszenia odpowiedzialności za wspólnie realizowane zadania

Cel przedmiotu:

Zapoznanie słuchaczy z elementami teorii języków formalnych i teorii translacji oraz zaznajomienie z narzędziami translacji sterowanej składnią w celu wykształcenia umiejętności samodzielnego tworzenia prostych systemów przetwarzania języków formalnych.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma uporządkowaną i podbudowaną teoretycznie wiedzę w zakresie podst. konstrukcji programistycznych, implementacji algorytmów, paradygmatów i stylów programowania, metod weryfikacji poprawności programów, języków formalnych, kompilatorów, platform - [K_W05]

2. ma uporządkowaną i podbudowaną teoretycznie wiedzę w zakresie podstawowych algorytmów i ich analizy, technik projektowania algorytmów, abstrakcyjnych struktur danych i ich implementacji, problemów obliczeniowo trudnych - [K_W04]

Umiejętności:

1. potrafi konstruować algorytmy z wykorzystaniem podstawowych technik algorytmicznych i dokonać analizy ich złożoności - [K_U09]

2. potrafi ocenić przydatność rutynowych metod i narzędzi służących do rozwiązywania prostych zadań inżynierskich typowych dla informatyki oraz wybierać i stosować właściwe technologie - [K_U22]

Kompetencje społeczne:

1. ma świadomość ważności dokładnego wykonania projektu, zachowania standardów notacyjnych, przestrzegania poprawności językowej i terminowego oddania prac - [K_K07]

Sposoby sprawdzenia efektów kształcenia

Wykład i ćwiczenia audytoryjne: sprawdzian pisemny z punktowanymi zadaniami (sprawdzenie wiedzy z zakresu teorii języków formalnych oraz teorii i technik translacji), kryterium zaliczenia od 50,1% punktów.

Laboratoria: test pisemny z punktowanymi zadaniami punktów sprawdzające umiejętność tworzenia prostych przetworników tekstów w języku Lex; kryterium zaliczenia od 50,1%

Treści programowe

Wykłady:

Pojęcie symbolicznego języka formalnego. Alfabet, składnia i semantyka języka. Podejście generacyjne i akceptorowe do definiowania składni. Klasyfikacja Chomsky?ego języków formalnych. Języki regularne: automaty skończenie stanowe, wyrażenia regularne. Przetwarzanie języków regularnych za pomocą systemu Lex. Języki bezkontekstowe: automaty ze stosem, gramatyki bezkontekstowe. Języki kontekstowe i obliczalne i ich akceptory. Wzmianka o formalnych metodach definiowania semantyki języków programowania. Pojęcie translacji. Interpretacja a kompilacja. Fazy i przebiegi kompilacji. Etap analizy w procesie kompilacji wyrażony jako translacja sterowana składnią: analiza leksykalna i składniowa, analiza zależności kontekstowych. Etap syntezy kodu: języki pośrednie i generacja kodu pośredniego. Elementy systemu wykonawczego: zarządzanie pamięcią i realizacja dostępu do nazw nielokalnych.

Ćwiczenia audytoryjne. Rozwiązywanie zadań związanych z formalnym definiowaniem języków oraz specyfikowaniem ich przetworników.

1. Wyrażenia regularne (modyfikacja 2017: definiowanie analizatora leksykalnego prostego języka programowania PJP).
2. Automaty skończenie stanowe.
3. Sprawdzian.

Laboratoria. Implementowanie prostych języków formalnych i ich przetworników w środowisku systemu Lex na platformie Linux.

1. Środowisko wykonawcze + Lex.
2. Programowanie przetworników ogólnego przeznaczenia w języku Lex.
3. (Modyfikacja 2017) Programowanie analizatora leksykalnego PJP w języku Lex.
4. Test z języka Lex.

Zastosowane metody kształcenia:

a) wykłady ilustrowane slajdami i przykładami uruchomień programów

b) ćwiczenia audytoryjne: rozwiązywanie zadań na tablicy przez studentów z przedyskutowaniem cech powstałych rozwiązań (dodatkowe oceny punktowe za aktywność w tym zakresie)

c) ćwiczenia laboratoryjne: samodzielne rozwiązywanie zadań przez studentów (programowanie przetworników tekstu) w celu przygotowania się do testu pisemnego.

Literatura podstawowa:

1. Cybulka J., Jankowska B., Nawrocki J. R.: Automatyczne przetwarzanie tekstów. AWK, Lex i YACC, Wyd. NAKOM, Poznań, 2002.
2. Hopcroft J.E., Ullman J.D.: Wprowadzenie do teorii automatów, języków i obliczeń, PWN, Warszawa, 1994.
3. Aho A.V., Sethi R., Ullman J.: Kompilatory. Reguły, metody I narzędzia. WNT, Warszawa 2002.

Literatura uzupełniająca:

1. Dembiński P., Małuszyński J.: Matematyczne metody definiowania języków programowania, WNT, Warszawa 1981.
2. Kernighan B.W., Ritchie D.M.: Język ANSI C, WNT, 1994.

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. wykłady	12
2. ćwiczenia audytoryjne	8
3. ćwiczenia laboratoryjne	8
4. przygotowanie do ćwiczeń laboratoryjnych	22
5. przygotowanie do ćwiczeń audytoryjych	22
6. przygotowanie do sprawdzianu z: wykład+ćwiczenia audyt.	13
7. przygotowanie do sprawdzianów z ćwiczeń lab.	13
8. sprawdziany	2

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	100	4
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	30	1
Zajęcia o charakterze praktycznym	60	2