KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Języki formalne i kompilatory	**1010514361010510088**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**3 / 6**

Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
-	**polski**	**obieralny**

Stopień studiów:	**Forma studiów** (stacjonarna/niestacjonarna)
I stopień	**niestacjonarna**

Godziny	Liczba punktów
Wykłady: **18** Ćwiczenia: **-** Laboratoria: **18** Projekty/seminaria: **-**	**5**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
kierunkowy	**z danego kierunku**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**
nauki techniczne	**5 100%**
nauki techniczne	**5 100%**

Odpowiedzialny za przedmiot / wykładowca:

dr inż. W. Complak
email: Wojciech.Complak@cs.put.poznan.pl
tel. (0-61) 665-2983
Instytut Informatyki
ul. Piotrowo 3, 60-965 Poznań

Odpowiedzialny za przedmiot / wykładowca:

dr hab. inż. Jerzy Nawrocki
email: Jerzy.Nawrocki@cs.put.poznan.pl
tel. (0-61) 665-3422
Instytut Informatyki
ul. Piotrowo 3, 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z algorytmiki i programowania w językach imperatywnych.
2	**Umiejętności:**	Powinien posiadać umiejętność rozwiązywania podstawowych problemów z zakresu zaprojektowania, sprawdzenia poprawności i zaimplementowania algorytmów w języku C oraz umiejętność pozyskiwania informacji ze wskazanych źródeł.
3	**Kompetencje społeczne**	Powinien również rozumieć konieczność poszerzania swoich kompetencji / mieć gotowość 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:

1. Przekazanie studentom podstawowej wiedzy z zakresu praktycznych aspektów teorii języku formalnych oraz budowy translatorów i środowisk czasu wykonania w zakresie zasad, technik i narzędzi wykorzystywanych współcześnie do budowy kompilatorów i innych narzędzi do automatycznego przetwarzania tekstu, takich jak: edytory tekstu, systemy wyszukiwania informacji, systemy składu elektronicznego i weryfikatory programów.

2. Rozwijanie u studentów umiejętności rozwiązywania prostych problemów za pomocą języków programowania ogólnego przeznaczenia jak i z wykorzystaniem do tego celu specjalistycznych narzędzi. Poszerzenie wiedzy na temat wcześniej wykorzystywanych środowisk programistycznych i języków programowania w wyniku spojrzeniu na nie z punktu widzenia projektanta i implementatora a nie tylko użytkownika.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną w zakresie algorytmów, architektury systemów komputerowych, systemów operacyjnych, języków i paradygmatów programowania - [K1st_W4]

2. zna podstawowe techniki, metody i narzędzia wykorzystywane w procesie rozwiązywania zadań informatycznych w zakresie analizy algorytmów, architektury systemów komputerowych, systemów operacyjnych i implementacji języków programowania - [K1st_W7]

3. zna zasady doboru narzędzi programistycznych i metod przetwarzania tekstu - [K1st_W7]

Umiejętności:

1. potrafi właściwie zaplanować i wykonać testy funkcjonalne i pozafunkcjonalne oprogramowania - [K1st_U3]

2. potrafi zaprojektować oraz zrealizować zadanie informatyczne stosując odpowiednio dobrane metody analityczne i eksperymentalne - [K1st_U4]

3. ma umiejętność specyfikowania i implementowania analizatorów z wykorzystaniem poznanych narzędzi - [K1st_U11]

Kompetencje społeczne:

1. ma świadomość znaczenia wiedzy w rozwiązywaniu problemów inżynierskich oraz zna przykłady i rozumie przyczyny wadliwie działających systemów informatycznych, które doprowadziły do poważnych strat finansowych, społecznych lub też do poważnej utraty zdrowia, a nawet życia - [K1st_K2]

Sposoby sprawdzenia efektów kształcenia

Efekty kształcenia 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 ćwiczeń:

- na podstawie oceny bieżącego postępu realizacji zadań,

Ocena podsumowująca:

Sprawdzanie założonych efektów kształcenia realizowane jest przez:

- ocenę przygotowania studenta do poszczególnych sesji zajęć laboratoryjnych (sprawdzian "wejściowy") oraz ocenę umiejętności związanych z realizacją ćwiczeń laboratoryjnych,

- ocenianie ciągłe, na każdych zajęciach (odpowiedzi ustne) - premiowanie przyrostu umiejętności posługiwania się poznanymi zasadami i metodami,

- ocenę wiedzy i umiejętności związanych z realizacją zadań projektowych/laboratoryjnych poprzez kolokwium pod koniec semestru, kolokwium obejmuje 11 zadań o charakterze praktycznym dotyczących poszczególnych narzędzi i zagadnień omawianych w ramach przedmiotu (2 x AWK, 2 x lex, 2 x LLgen, 2 x yacc, 3 x SLR); zadania mają zarówno charakter konstrukcyjny (np. napisz program) jak i analityczny (np. jaka będzie odpowiedź danego programu)

- ocenę i "obronę" przez studenta sprawozdania z realizacji projektu,

Uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za:

- omówienia dodatkowych aspektów zagadnienia,

- efektywność zastosowania zdobytej wiedzy podczas rozwiązywania zadanego problemu,

- uwagi związane z udoskonaleniem materiałów dydaktycznych,

- wskazywanie trudności percepcyjnych studentów umożliwiające bieżące doskonalenia procesu dydaktycznego.

Treści programowe

Pierwszy wykład poświęcony jest omówieniu organizacji zajęć (zakresu przedmiotu, środowiska i narzędzi, literatury i zasad zaliczania) oraz wprowadzeniu do tematyki przetwarzania tekstu na przykładzie języka AWK.

Na drugim wykładzie przedstawiany jest model analiza-synteza translatora, podział procesu translacji na etapy oraz faza analizy leksykalnej i zasady prowadzenie jej z wykorzystaniem generatora analizatorów leksykalnych lex.

Pierwsze zajęcia laboratoryjne poświęcone są zagadnieniom organizacyjnym: zaznajomieniu się ze środowiskiem i narzędziami, uruchamianiem skryptów do kompilacji oraz nauce wykorzystywania języka AWK do przetwarzania tekstu.

Trzeci wykład, otwierający cykl poświęcony analizie składniowej, zawiera omówienie ogólnych zasad prowadzenia analizy składniowej i pojęć związanych z gramatykami bezkontekstowymi (takich jak: terminale i nieterminale, produkcje, wywody, typy rekurencji, niejednoznaczność i równoważność gramatyk) oraz wstęp do metody wstępującej. Wykład obejmuje charakterystykę generatora yacc, składnię specyfikacji analizatora składniowego, zasady współpracy z analizatorem leksykalnym oraz wykrywania i obsługi błędów składniowych.

W trakcie kolejnego wykładu przedstawiana jest koncepcja translacji sterowanej składnią. Przedstawiane są pojęcia atrybutów, definicji sterowanych składnią, schematów translacji oraz definicji S-atrybutowych i L-atrybutowych.

Omawiane są także zasady implementacji translacji sterowanej składnią w metodzie wstępującej w generatorze yacc (atrybuty syntetyzowane i dziedziczone, typy atrybutów, akcje wielokrotne).

W ramach zajęć laboratoryjnych studenci przechodzą do zapoznawania się z projektowaniem i implementowaniem prostych filtrów tekstu z wykorzystaniem generatora analizatorów leksykalnych lex.

Kolejny wykład z cyklu dotyczącego analizy składniowej poświęcony jest posługiwaniu się gramatykami niejednoznacznymi w metodzie wstępującej w generatorze yacc. Przedstawiane są zalety i typowe, praktyczne przykłady gramatyk niejednoznacznych oraz zasady wykorzystywania ich w generatorze yacc.

W ramach szóstego wykładu przedstawiana jest analiza semantyczna: różne typy kontroli zależności kontekstowych, takie jak: sprawdzenie przepływu sterowania, unikalności deklaracji nazw, powtórzeń nazw oraz kontrola typów.

Na zajęciach laboratoryjnych studenci rozwiązują zadania dotyczące generatora yacc.

Wykład kończący cykl dotyczący analizy składniowej poświęcony jest porównaniu wad i zalet różnych metod tworzenia translatorów działających w oparciu o metodę wstępującą oraz demonstracji sposobu generowania kodu parsera.

Kolejny wykład poświęcony jest etapowi syntezy kodu pośredniego: omawiane są różne rodzaje kodów pośrednich i maszyny wirtualne, a jako przykład konkretnej implementacji, szczegółowo przedstawiany jest kod trójadresowy. W ramach wykładu dokonywany jest również przegląd zagadnień dotyczących generacji kodu wynikowego i jego optymalizacji oraz budowy środowiska wykonawczego, takich jak dostęp do nazw nielokalnych, dynamiczny przydział pamięci i przekazywanie parametrów do podprogramów.

Na laboratoriach studenci implementują w yaccu i lexie translatora tłumaczącego kod pomiędzy wybranymi, znanymi im językami imperatywnymi.

Ostatni wykład poświęcony jest podsumowaniu całości przedstawionych w ciągu semestru zagadnień oraz sprawdzeniu wiedzy studentów w formie kolokwium zaliczeniowego.

Na zajęciach laboratoryjnych studenci rozliczają wykonanie projektów.

Metody dydaktyczne:

1. wykład: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, rozwiązywanie zadań, demonstracja narzędzi programistycznych,
2. ćwiczenia laboratoryjne: rozwiązywanie zadań, dyskusja.

Literatura podstawowa:

1. Kompilatory. Reguły, metody i narzędzia, A. V. Aho, R. Sethi, J. D. Ullman, WNT, Warszawa, 2002
2. Automatyczne przetwarzanie tekstów, J. Cybulka, B. Jankowska, J. Nawrocki, Nakom, Poznań, 2002
3. Wprowadzenie do przetwarzania tekstów w języku AWK, J. Nawrocki, W. Complak, Nakom (Pro Dialog), Poznań, 1994
4. Wprowadzenie do generatora Lex, J. Nawrocki, A. Czajka, Nakom (Pro Dialog), Poznań, 1998

Literatura uzupełniająca:

1. lex & yacc, 2nd Edition, D. Brown, J. Levine, T. Mason, O?Reilly Media, 1992
2. The Definitive ANTLR Reference: Building Domain-Specific Languages, T. Parr, The Pragmatic Bookshelf, 2007
3. Compilers: Principles, Techniques, and Tools, 2. Ed., A.V. Aho, M. S. Lam, R. Sethi, J.D. Ullman, Addison-Wesley, 2007

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)

1. udział w zajęciach laboratoryjnych / ćwiczeniach:	18
2. przygotowanie do ćwiczeń laboratoryjnych:.	18
3. udział w konsultacjach związanych z realizacją procesu kształcenia, w szczególności ćwiczeń laboratoryjnych / projektu	2
4. napisanie programu / programów, uruchomienie I weryfikacja (czas poza zajęciami laboratoryjnymi)	10
5. przygotowanie do sprawdzianów / kolokwium	10
6. udział w wykładach	18
7. zapoznanie się ze wskazaną literaturą / materiałami dydaktycznymi (10 stron tekstu naukowego = 1 godz.), 50 stron	5
8. przygotowanie do zaliczenia wykładów i udział w kolokwium zaliczeniowym	15

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	96	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	38	3
Zajęcia o charakterze praktycznym	56	3