

STUDIA I STOPNIA

Nazwa	Podstawy programowania
Nazwa w j. ang.	Introduction to programming

Koordynator	Dr Dawid Nałęcz	Zespół dydaktyczny
		Dr Dawid Nałęcz
Punktacja ECTS*	3	

Opis kursu (cele kształcenia)

Celem kursu jest zapoznanie studentów z zasadami programowania imperatywnego o cechach strukturalnych i proceduralnych, metodami projektowania i zapisu algorytmów oraz nieskomplikowanych programów.
Po zakończeniu kursu student powinien znać podstawy programowania w języku C i umieć je stosować.
Przedmiot prowadzony jest w języku polskim.

Warunki wstępne

Wiedza	Student zna podstawowe pojęcia związane z programowaniem.
Umiejętności	Student posiada umiejętność opisanie prostych algorytmów.
Kursy	Wstępne kursy nie są wymagane.

Efekty uczenia się

Wiedza	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
--------	-----------------------------	-------------------------------------

	Po zakończeniu kursu student:	
	W 01: zna podstawy programowania, w tym, rozumie ideę programowania strukturalnego i proceduralnego w języku programowania wysokiego poziomu (C).	K_W05, K_W06, K_W10
	W 02 wie na czym polega działanie kompilatora języka.	K_W05,K_W06
	W 03 zna wybrane klasyczne algorytmy (w tym wybrane algorytmy wyszukiwania i porządkowania).	K_W05,K_W06
	W 04 student zna konstrukcje nieskomplikowanych struktur danych	KW_05,KW_06
	w 05 rozumie sposoby reprezentacji danych w komputerze.	K_W05,K_W06

	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
Umiejętności	Po zakończeniu kursu student:	
	U01: projektuje i zapisuje proste algorytmy, analizuje ich działanie oraz umie wykorzystać klasyczne algorytmy w sytuacjach problemowych.	K_U05,K_U07,K_U09,
	U02: samodzielnie pisze proste programy w języku C (z zachowaniem stylu gwarantującego czytelność kodu), w których: -poprawnie definiuje i wykorzystuje proste i złożone struktury danych takie jak: tablice jednowymiarowe, tablice dwuwymiarowe, struktury, -poprawnie deklaruje, definiuje i wywołuje funkcje, w tym również z parametrami, którymi są wskaźniki, tablice oraz struktury.	K_U05,K_U07,K_U09,
	U03: umie kompilować i uruchamiać programy w języku C (stosując zestaw kompilatorów projektu GNU - GCC) oraz śledzić ich wykonywanie, analizować i poprawiać błędy składniowe oraz błędy wykonania.	K_U05,K_U07,K_U09

	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
Kompetencje społeczne	Po zakończeniu kursu student:	
	K01: potrafi korzystać z różnych źródeł informacji (w tym zasobów sieci Internet) do poszerzania własnej wiedzy i zdobywania nowych umiejętności z zakresu programowania,	K_K01,
	K02: potrafi przekazywać wiedzę informatyczną w sposób zrozumiały dla innych.	K_K06

Organizacja

Forma zajęć	Wykład (W)	Ćwiczenia w grupach											
		A		K		L		S		P		E	
Liczba godzin													

Opis metod prowadzenia zajęć

Podczas zajęć laboratoryjnych studenci rozwiązują problemy zdefiniowane przez prowadzącego zajęcia.

Ćwiczenia nieukończone w trakcie zajęć studenci kończą samodzielnie (zdalnie na dedykowanym serwerze) oraz przesyłają na wskazany adres e-maili przed kolejnymi zajęciami.

Właściwa praca laboratoryjna poprzedzona jest przedstawieniem potrzebnych zagadnień i przykładów.

Formy sprawdzania efektów uczenia się

	E – learning	Gry dydaktyczne	Ćwiczenia w szkole	Zajęcia terenowe	Praca laboratoryjna	Projekt indywidualny	Projekt grupowy	Udział w dyskusji	Referat	Praca pisemna (esej)	Egzamin ustny	Egzamin pisemny	Inne
W01					X	X							
W02					X	X							
W03					X	X							
W04					X	X							
W05					X	X							
U01					X	X							
U02					X	X							
U03					X	X							
K01					X	X							
K02					X	X							

Kryteria oceny	Ocenę dobrą lub bardzo dobrą może uzyskać student, który wykaże się dobrą lub bardzo dobrą znajomością języka C oraz umiejętnością samodzielnego konstruowania programów. Ocena końcowa jest średnią ocen za poszczególne zadania wykonywane w trakcie ćwiczeń laboratoryjnych.
----------------	---

Uwagi	
-------	--

Treści merytoryczne (wykaz tematów)

1. Algorytmy oraz ich reprezentacje, struktury danych, program komputerowy, kod źródłowy, systemy liczbowe i reprezentacja danych w komputerze.
2. Obsługa terminala, zdalny dostęp do zasobów serwerowych.
3. Budowa programu w języku C. Formatowanie wejścia i wyjścia.
4. Typy danych. Instrukcje wyboru oraz pętli.
5. Implementacja prostych algorytmów.
6. Tablice jednowymiarowe i wielowymiarowe.
7. Funkcje.
8. Zmienne lokalne i globalne. Zasięg zmiennych.
9. Wskaźniki. Związek wskaźników z tablicami.
10. Ciągi znaków.
11. Struktury. Wskaźniki do struktur.
12. Dynamiczny przydział pamięci.
13. Obsługa plików.
14. Biblioteka standardowa języka C.

Wykaz literatury podstawowej

The GNU C Reference Manual <http://www.gnu.org/software/gnu-c-manual/> (eng.)
 King K. N., Język C. Nowoczesne programowanie. Wydanie II, Helion 2011
 Prata S.: Język C. Szkoła programowania, Wydanie V, Helion 2006

Wykaz literatury uzupełniającej

Kernighan B.W., Ritchie D.M.: Język ANSI C. Programowanie. Wydanie II, Helion 2010
 Kotowski M., Wysokie C, Lupus 1994
 Cormen T.H., Leiserson C.E., Rivest R.L., Stein C., Wprowadzenie do algorytmów, PWN 2013
 Hyde R., Profesjonalne programowanie. Część 1, Helion 2005
 Reese R., Wskaźniki w języku C. Przewodnik, Helion 2013

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta)

liczba godzin w kontakcie z prowadzącymi	Wykład	0
	Konwersatorium (ćwiczenia, laboratorium itd.)	30
	Pozostałe godziny kontaktu studenta z prowadzącym	15
liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	15
	Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu	15
	Przygotowanie projektu lub prezentacji na podany temat (praca w grupie)	
	Przygotowanie do egzaminu/zaliczenia	15

Ogółem bilans czasu pracy	90
Liczba punktów ECTS w zależności od przyjętego przelicznika	3