KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Introduction to Computing	**1010514311010510189**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**1 / 1**
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: **16** Ćwiczenia: **-** Laboratoria: **16** 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 %)**

Odpowiedzialny za przedmiot / wykładowca:

dr hab. inż. J. Nawrocki, prof. nadzw. PP.
email: jerzy.nawrocki@put.poznan.pl
tel. 6652980
Instytut Informatyki
ul. Piotrowo 2, 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Wymagana jest wiedza na poziomie IV Polskiej Ramy Kwalifikacji (matura).
2	**Umiejętności:**	Wymaga się umiejętności logicznego myślenia i skupienia uwagi.
3	**Kompetencje społeczne**	Uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu:

Przedstawienie studentom ogólnego obrazu informatyki, w tym pokazanie związków między różnymi obszarami informatyki ze szczególnym uwzględnieniem z jednej strony aspektów praktycznych, a z drugiej ? perspektyw rozwoju informatyki.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. Ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną w zakresie sprzętu komputerowego wraz z jego alternatywnymi postaciami (maszyna Turinga, DNA computing); paradygmatem programowania imperatywnego (C i język asemblera) wraz z jego alternatywami (przetwarzanie regułowe, wyrażenia regularne); - [K1s_W4]

2. podstawowych strukturami danych (dynamiczny przydział pamięci, grafy, drzewa i listy, stos, kolejka priorytetowa); metod opisu złożoności obliczeniowej algorytmów i granic obliczalności (problem stopu); podstaw metod numerycznych (stabilność numeryczna, schemat Hornera, szeregi Maclaurina i ich zastosowanie, metoda stycznych Newtona); - [K1s_W4]

3. obliczeń współbieżnych i problemów ich synchronizacji (POSIX, semafory, problem producenta-konsumenta, problem czytelników i pisarzy); sieci komputerowych i cyberbezpieczeństwa; modelu relacyjnego danych i SQL; mikrokontrolerów i ich zastosowań; postaw inżynierii oprogramowania (fazy cyklu rozwoju oprogramowania, wybrane rodzaje diagramów języka UML i ich zastosowanie). - [K1s_W4]

4. Ma wiedzę o takich kierunkach rozwoju informatyki, jak alternatywne środowiska obliczeniowe (np. DNA computing), poza-imperatywne paradygmaty programowania (np. Board Programming), sieci komputerowe (perspektywy związane z syst.5G) i cyberbezpieczeństwo (szyfrowanie homomorficzne) oraz o związkach informatyki z automatyką i robotyką,elektroniką,matematyką,telekomunikacją i zarządzaniem. - [K1st_W5]

5. Ma podstawową wiedzę o cyklu życia oprogramowania (model wodospadowy i jego słabości oraz środki zaradcze, w tym metodyki zwinne). - [K1st_W6]

6. Ma podstawową wiedzę nt. zasad skutecznego działania wg Stephena Covey?ego, kodu etycznego ACM (ACM Code of Ethics), zagrożeń związanych z przestępczością elektroniczną (m.in. ransomwere) oraz specyfiki systemów krytycznych ze względu na bezpieczeństwo (m.in. pojęcie fail-safe). - [K1st_W8]

7. . Ma podstawową wiedzę nt. prawa autorskiego, patentów i ochrony danych osobowych (RODO) - [Kst1_W11]]

Umiejętności:

1. potrafi odpowiednio posługiwać się technikami informacyjno-komunikacyjnymi, znajdującymi zastosowanie na różnych etapach realizacji przedsięwzięć informatycznych - [K1st_U2]

2. Potrafi dyskutować o społecznych, prawnych i ekonomicznych aspektach informatyki - [K1st_U5]

3. Potrafi ocenić i opisać za pomocą notacji O(..) złożoność obliczeniową prostych algorytmów (m.in. heap sort) - [-]

4. Rozumie proste programy zapisane w języku assemblera, w języku C (także z wybranymi elementami API w standardzie POSIX) oraz proste zapytania w języku SQL - [-]

5. Rozumie działanie prostych systemów zbudowanych z bramek, dekoderów i multiplekserów, a także rozumie działanie mikrokontrolerów rodziny 8051, w tym aspekty czasowe - [-]

6. Potrafi pracować w małym (do 4 osób) zespole - [K1st_U18]

Kompetencje społeczne:

1. Rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe - [K1st_K1]

2. Ma świadomość społecznych i ekonomicznych konsekwencji, jakie mogą wynikać z wad systemów informatycznych - [K1st_K2]

Sposoby sprawdzenia efektów kształcenia

Osiągnięcie efektów kształcenia związanych z wiedzą W4, W5, W6, W8 i W11 oraz umiejętnościami U2, U5 i U18 zostanie sprawdzone za pomocą trzech testów śródsemestralnych indywidualnych, dwóch zawodów zespołowych (zespoły maksymalnie 4-osobowe) i egzaminu końcowego (indywidualnego). Umiejętność U5 oraz kompetencje społeczne K1 i K2 będą zweryfikowane w trakcie zajęć laboratoryjnych. Weryfikacją umiejętności U18 będą wspomniane zawody zespołowe.

Ocena formująca:

Każdy test indywidualny będzie się składał z kilku par zadań podobnych. Wystarczy rozwiązać jedno zadanie z pary, aby zaliczyć daną parę zadań. Żeby zaliczyć test należy zaliczyć co najmniej n-1 par zadań, gdzie n jest liczbą zadań w danym teście. Rozwiązując test można mieć do dyspozycji notatki zapisane na jednej kartce formatu A4 (czcionka dowolna). Żadne inne pomoce (kalkulatory, smartfony itp.) NIE są dozwolone. Niech T1, T2, T3 będą zmiennymi o wartościach logicznych (prawda, fałsz) określającymi, czy dana osoba zaliczyła dany test.

Każde zawody zespołowe będą się składały z kilku (ew. kilkunastu) zadań i będą oceniane w skali 0-100% (pojedyncze zadanie będzie oceniane w skali 0-10 i uzyskana suma punktów będzie dzielona przez maksymalną możliwą do otrzymania sumę punktów). W trakcie zawodów można mieć dowolne pomoce papierowe (materiały wykładowe, słowniki, książki, notatki itd.), ale nie żadne inne. Niech Z1, Z2 oznaczają punktację procentową uzyskaną przez dany zespół w odpowiednio pierwszych i drugich zawodach. Na podstawie sumy Z1+Z2 tworzy się ranking zespołów. Ocena zespołu, G, zależy od pozycji zespołu w rankingu i tak najwyższe 10% daje ocenę 5.0 (A), następne 20% ocenę 4.5 (B), kolejne 30% ocenę 4.0 (C) i następne 20% ocenę 3.5 (D). Ostatnie 20% zespołów biorących udział w zawodach ma ocenę nieokreśloną N/A. Ocena wypracowana przez zespół przechodzi na każdego członka zespołu. Osoba nie biorąca udział w zawodach dostaje ocenę N/A.

Egzamin zwykły i poprawkowy będzie się składał z pytań o charakterze testowym i zadań o podobnym stopniu trudności jak te na zawodach zespołowych. Niech Tz i Tp oznaczają wynik testu na egzaminie odpowiednio zwykłym i poprawkowym (są to wartości logiczne), natomiast Gz i Gp ocenę zadań na egzaminie zwykłym i poprawkowym (w skali 5.0, 4.5, 4.0, 3.5, 30, 2.0).

Ocena podsumowująca:

T1 and T2 and T3 => max{ 3.0, G, Gz }

not (T1 and T2 and T3) => (Tz => max{3.0, Gz})

not (T1 and T2 and T3) and not Tz (Tp => max{3.0, Gp})

not (T1 and T2 and T3) and not Tz and not Tp => 2.0 (F)

Treści programowe

Zasady skutecznego działania wg Stephena Covey?ego; podstawowe instrukcje imperatywne (wejście-wyjście, warunek, pętla). Układy techniki cyfrowej (bramki, dekodery, multipleksery, sumator, przerzutnik R-S i D, rejestr, układ adresowania). Koncepcja von Neumanna. Reprezentacja liczb w uzupełnieniu do 2. Podstawowe instrukcje języka asemblera (arytmetyczne, instrukcje skoku warunkowego i bezwarunkowego). Maszyna Turinga. DNA computing. Struktury danych na poziomie sprzętu (pamięć, stos) i na poziomie języka imperatywnego (tablice, rekordy). Przekazywanie parametrów do podprogramów. Wskaźniki i dynamiczny przydział pamięci. Listy, drzewa i grafy. Heap i jego wykorzystanie do sortowania. Opis złożoności algorytmów za pomocą notacji O(..). Granice obliczalności: funkcje szybko rosnące i problem stopu. Reprezentacja liczba rzeczywistych w komputerze i problem stabilności numerycznej. Schemat Hornera. Szeregi Maclaurina i ich zastosowanie. Metoda stycznych i obliczanie pierwiastka kwadratowego. Standard POSIX, obliczenia współbieżne i semafory. Problem producenta-konsumenta oraz problem czytelników i pisarzy. Protokół TCP/IP i programowanie na poziomie gniazd (sockets). Cyberbezpieczeństwo. Wprowadzenie do kryptografii (algorytmy szyfrowania DSA i RSA, szyfry homomorficzne, klucze publiczne i prywatne). Mikrokontrolery rodziny 8051: budowa i podstawy ich programowania z uwzględnieniem aspektów czasowych. Systemy krytyczne (mission-critical systems) Relacyjny model danych i podstawy języka SQL. Koncepcja uczenia maszynowego. Wyrażenia regularne. Wprowadzenie do inżynierii oprogramowania (cykl życia i jego podstawowe fazy, wymagania funkcjonalne i przypadki użycia, architektura, testowanie, wybrane diagramy języka UML). Aspekty etyczne (kodeks ACM) i prawne (prawo autorskie, patenty, ochrona danych osobowych).

Ćwiczenia laboratoryjne są skorelowane z wykładami. Studenci analizują układy elektroniczne, kody programów (i uruchamiają je) oraz dyskutują aspekty poruszane na wykładzie.

Literatura podstawowa:		
Literatura uzupełniająca:		
Bilans nakładu pracy przeciętnego studenta		
Czynność		**Czas (godz.)**
1. Udział w wykładach		30
2. Udział w zajęciach laboratoryjnych		30
3. Bezpośrednie przygotowanie do ćwiczeń laboratoryjnych		10
4. Studia literaturowe i praca własna		50
5. Przygotowanie do zaliczenia przedmiotu i udział w egzaminie 2 godz.		10
Obciążenie pracą studenta		
forma aktywności	**godzin**	**ECTS**
Łączny nakład pracy	130	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	62	2
Zajęcia o charakterze praktycznym	40	2