

KARTA PRZEDMIOTU (SYLABUS)

Opis przedmiotu

Kod przedmiotu		Nazwa przedmiotu	PROGRAMOWANIE NISKOPOZIOMOWE	
IT/P/I/NST/B _I -16			LOW-LEVEL PROGRAMMING	
Język wykładowy		polski		
Rok akademicki		2019/2020		
Kierunek		Informatyka techniczna		
w zakresie				
Poziom studiów		studia pierwszego stopnia		
Profil studiów		praktyczny		
Forma studiów		studia niestacjonarne		
Semestr / semestry		drugi letni		
Przynależność do grupy zajęć		B1. Grupa zajęć kierunkowych - obowiązkowych		
Status przedmiotu		obowiązkowy		
Formy realizacji zajęć dydaktycznych, wymiar, punkty ECTS		Forma zajęć	Liczba godzin zajęć dydaktycznych	Liczba punktów ECTS
		Wykład	20 [h]	4 ECTS
		Ćwiczenia laboratoryjne	25 [h]	
		
Powiązanie przedmiotu	z profilem studiów	kształtuje umiejętności praktyczne		2 ECTS
	z uprawnieniami	służy do zdobywania przez studenta kompetencji inżynierskich		4 ECTS
	z dyscypliną	informatyka techniczna i telekomunikacja		4 ECTS
Forma nauczania		tradycyjna – zajęcia zorganizowane w Uczelni		
Wymagania wstępne		Wymagana znajomość przedmiotów: teoretyczne podstawy informatyki, podstawy programowania		
Jednostka prowadząca		Katedra Informatyki		
Koordynator		dr Artur Hermanowicz		
Osoby prowadzące		dr Artur Hermanowicz		
Adres strony internetowej pjo		www.wim.uniwersytetradom.pl		
Adres e-mail, telefon koordynatora		artur.hermanowicz@uthrad.pl		

EFEKTY UCZENIA SIĘ, TREŚCI PROGRAMOWE, REALIZACJA ZAJĘĆ DYDAKTYCZNYCH, WERYFIKACJA EFEKTÓW UCZENIA SIĘ

Cel kształcenia:	Poznanie metod przechowywania i przetwarzania informacji w bliskim związku ze sprzętem komputerowym, a w szczególności: poznanie języka assembler, programowanie elementów systemu komputerowego, efektywniejsze wykorzystanie komputera dzięki znajomości zasad jego działania
Treści programowe:	<p>Wykłady: Język maszynowy, komputerowa reprezentacja informacji. [2h] – W1 Język assembler: etapy tworzenia programu w języku assembler, budowa programu. Podstawowe rozkazy procesorów 80x86: rozkazy przesłań, rozkazy arytmetyczne i logiczne, skoki warunkowe i bezwarunkowe, tworzenie pętli programowych, tworzenie procedur. [6h] – W1 Korzystanie z funkcji systemu operacyjnego, przerywania i ich obsługa, bezpośrednie działanie na pamięci karty graficznej w trybie znakowym i graficznym. [4h] – W1, W2 Zalety i wady programowania niskopoziomowego, porównanie programów w assemblerze do programów w językach wysokiego poziomu. [4h] – W1 Łączenie assemblera z językami wysokiego poziomu, wstawki assemblerowe, wskaźniki w C i Pascalu, bezpośrednie działanie na pamięci. Obsługa urządzeń zewnętrznych. [4h] – W1, W2</p> <p>Ćwiczenia laboratoryjne: Zapoznanie z podstawowymi instrukcjami języka assembler dla procesorów x86. Współpraca assemblera z językami wysokiego poziomu – tworzenie wstawek assemblerowych. [2h] – U1, K1 Programowanie prostych wstawek assemblerowych wykorzystujących rozkazy przesłań, instrukcje arytmetyczne i logiczne, skoki, tworzenie pętli. [2h] – U1 Tworzenie całego programu w assemblerze. Wykorzystanie assemblera, linkera i debuggera. Implementacje prostych algorytmów assemblerze. [3h] – U1, U2 Tworzenie procedur. Implementacje programów wykorzystujących podprogramy. [3h] – U1, U2 Wykorzystanie przerwań. Zastosowanie przerwań do obsługi wejścia i wyjścia tekstowego. [3h] – U1, U2 Zastosowanie assemblera do bezpośredniego działania na pamięci karty graficznej. Implementacje prostych algorytmów dotyczących grafiki komputerowej, kreślenie linii, okręgu itp. [3h] – U1, U2 Programowanie koprocatora. Zastosowanie koprocatora do obliczeń zmiennoprzecinkowych. Współpraca procesora z koprocetorem. [2h] – U1, U2 Zastosowanie koprocatora do obliczeń związanych z generowaniem grafiki. Tworzenie animacji. [3h] – U1, U2, K1 Programowanie w assemblerze w systemie Windows. Wykorzystanie funkcji systemu operacyjnego. Korzystanie z API systemu Windows. [4h] – U1, U2, K1</p>
Metody dydaktyczne (kształcenia):	<p>Metody podające - wykład informacyjny – W1, W2 Metody praktyczne – ćwiczenia laboratoryjne - U1, U2, K1</p> <p>Wszystkie zastosowane metody umożliwiają rozpoznawanie i zaspokajanie indywidualnych potrzeb studentów, w tym studentów niepełnosprawnych oraz indywidualizację toku studiów.</p>
Rygor zaliczenia, kryteria oceny osiągniętych efektów uczenia się, sposób obliczania oceny końcowej:	<p>Warunkiem zaliczenia przedmiotu jest osiągnięcie wszystkich wymaganych efektów kształcenia określonych dla danego przedmiotu. Uzyskanie pozytywnych ocen ze wszystkich form zajęć wchodzących w skład danego przedmiotu jest równoznaczne z jego zaliczeniem i zdobyciem przez studenta liczby punktów ECTS przyporządkowanej temu przedmiotowi. Sposób obliczenia oceny końcowej z przedmiotu określa regulamin studiów. Sposób obliczania oceny z poszczególnych form zajęć przedstawia się następująco:</p> <p>Ćwiczenia laboratoryjne – warunkiem zaliczenia jest osiągnięcie wszystkich wymaganych efektów kształcenia dla tej formy zajęć i uzyskanie pozytywnych ocen za pomocą przyjętych dla przedmiotu metod oceniania. Ocena końcowa z ćwiczeń laboratoryjnych stanowi sumę ocen: 90 % sprawdziany praktyczne przy komputerze, 10% aktywność na zajęciach. Wykład – 100% ocena z testu zaliczeniowego.</p>

Efekty uczenia się dla przedmiotu w odniesieniu do efektów kierunkowych i formy zajęć				Metody weryfikacji efektów uczenia się	
Numer	Opis efektów uczenia się dla przedmiotu (PEU)	Kierunkowy	Forma zajęć	Forma	Metody

efektu uczenia się	Student, który zaliczył przedmiot (W) zna i rozumie/ (U) potrafi /(K) jest gotów do:	efekt uczenia się (KEU)		weryfikacji (zaliczeń)	sprawdzania i oceny
W1	Ma uporządkowaną wiedzę w zakresie budowy procesora oraz zna metody programowania go na poziomie listy rozkazów.	K_WG06	wykład	zaliczenie na ocenę	kolokwium
W2	Ma uporządkowaną wiedzę w zakresie systemów operacyjnych. Zna funkcje systemu i metody wykorzystywania ich w programowaniu niskopoziomym.	K_WG06	wykład	zaliczenie na ocenę	kolokwium
U1	Potrafi zaprogramować niskopoziomowo poszczególne elementy systemu komputerowego tj. procesor, karta graficzna rozumiejąc ich budowę i zasady działania.	K_UW02	ćwiczenia laboratoryjne	zaliczenie na ocenę	kolokwium
U2	Potrafi sformułować algorytm i zaprogramować go w języku niskiego poziomu posługując się odpowiednimi narzędziami.	K_UW12	ćwiczenia laboratoryjne	zaliczenie na ocenę	kolokwium
K1	Ma świadomość ciągłego rozwoju technologii komputerowych i konieczność stałego aktualizowania i poszerzania swej wiedzy.	K_KK01	ćwiczenia laboratoryjne	zaliczenie na ocenę	aktywność na zajęciach

Stopień osiągnięcia kierunkowych efektów uczenia się: K_WG06+++, K_UW02+++, K_UW12+++, K_KK01++

Literatura podstawowa, literatura uzupełniająca, pomoce naukowe

Literatura podstawowa:

1. Gawrylczyk M.: *Efekty graficzne w assemblerze*, Helion, Gliwice 1996.
2. Irvine K.R.: *Assembler dla procesorów Intel*, Helion, Gliwice 2003.
3. Kowalczyk A.: *Assembler*, Croma, Wrocław 1999.
4. Kruk S.: *Assembler. Wykłady i ćwiczenia*, Mikom, Warszawa 2003.

Literatura uzupełniająca:

1. Błaszczak A.: *Win32ASM. Assembler w Windows*, Helion, Gliwice 2004.
2. Kopacz T.: *Karty graficzne VGA i SVGA*, Mikom, Warszawa 1995
3. Kruk S.: *Assembler – Podręcznik użytkownika*, Mikom, Warszawa 1999.
4. Michałek G.: *Co i jak w assemblerze*, Intersoftland, Warszawa 1995.
5. Syck G.: *Turbo Assembler. Biblia użytkownika*, LT&P, Warszawa 1994.

Naład pracy studenta potrzebny do osiągnięcia zakładanych efektów uczenia się – bilans punktów ECTS			
Udział w zajęciach, aktywność	Obciążenie studenta [h]		
	Inne godz. kontaktowe (IGK)	Zajęcia bez nauczyciela-praca własna studenta (ZBN)	Zajęcia dydaktyczne
Udział w wykładach	X	X	20 [h]
Samodzielne studiowanie tematyki wykładów	X	20 [h]	X
Udział w ćwiczeniach laboratoryjnych	X	X	25 [h]
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	X	25 [h]	X
Udział w konsultacjach	3 [h]	X	X
Przygotowanie do zaliczenia	X	5 [h]	X
Udział w zaliczeniu	2 [h]	X	X
Sumaryczne obciążenie pracą studenta	5 [h]/ 0,2 ECTS	50 [h]/2,0 ECTS	45 [h]/ 1,8 ECTS
Punkty ECTS za przedmiot	4 ECTS		

Informacje dodatkowe, uwagi
Terminy odbywania zajęć: zgodnie z planem zajęć.
Miejsce odbywania zajęć: UTH Radom, ul. Malczewskiego 20A