

<b>KARTA OPISU MODUŁU KSZTAŁCENIA</b>		
Nazwa modułu/przedmiotu <b>Programowanie systemowe i współbieżne</b>		Kod <b>1010514331010510192</b>
Kierunek studiów <b>Informatyka</b>	Profil kształcenia (ogólnoakademicki, praktyczny) <b>ogólnoakademicki</b>	Rok / Semestr <b>2 / 3</b>
Ścieżka obieralności/specjalność <b>-</b>	Przedmiot oferowany w języku: <b>polski</b>	Kurs (obligatoryjny/obieralny) <b>obligatoryjny</b>
Stopień studiów: <b>I stopień</b>	Forma studiów (stacjonarna/niestacjonarna) <b>niestacjonarna</b>	
Godziny Wykłady: <b>16</b> Ćwiczenia: <b>-</b> Laboratoria: <b>16</b> Projekty/seminaria: <b>-</b>		Liczba punktów <b>5</b>
Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny) <b>kierunkowy</b>		(ogólnouczelniany, z innego kierunku) <b>z danego kierunku</b>
Obszar(y) kształcenia i dziedzina(y) nauki i sztuki <b>nauki techniczne</b> <b>nauki techniczne</b>		Podział ECTS (liczba i %) <b>5 100%</b> <b>5 100%</b>
<b>Odpowiedzialny za przedmiot / wykładowca:</b>		
Prof. dr hab. inż. J. Brzeziński email: Jerzy.Brzezinski@cs.put.poznan.pl tel. tel. (0-61) 665-2903, fax: (0-61) 877 1525, Instytut Informatyki ul. Piotrowo 2, 60-965 Poznań		dr inż. Anna Kobusińska email: Anna.Kobusinska@cs.put.poznan.pl tel. tel. (0-61) 665-2964, fax: (0-61) 877 1525, Instytut Informatyki ul. Piotrowo 2, 60-965 Poznań
<b>Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:</b>		
1	<b>Wiedza:</b>	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z zakresu funkcjonowania systemów operacyjnych prezentowaną w ramach przedmiotu Systemy Operacyjne.
2	<b>Umiejętności:</b>	Powinien także posiadać umiejętności: programowania, definiowania niskopoziomowych struktur danych i rozwiązywania podstawowych problemów niskopoziomowego kodowania algorytmów, nabyte w ramach przedmiotu Programowanie niskopoziomowe. Student powinien posiadać także umiejętność pozyskiwania informacji ze wskazanych źródeł.
3	<b>Kompetencje społeczne</b>	Powinien również rozumieć konieczność poszerzania swoich kompetencji i 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.
<b>Cel przedmiotu:</b>		
1. Przekazanie studentom podstawowej wiedzy nt. elementów programowania współbieżnego oraz szczegółowej wiedzy z systemów operacyjnych w zakresie zarządzania procesami, mechanizmów synchronizacji i przeciwdziałania zakleszczeniom.		
2. Rozwijanie u studentów umiejętności rozwiązywania prostych problemów programowania współbieżnego oraz stosowania wybranych mechanizmów synchronizacji do rozwiązania klasycznych problemów synchronizacji.		
3. Kształtowanie u studentów umiejętności pracy zespołowej w trakcie realizacji projektu na zajęciach laboratoryjnych.		
<b>Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia</b>		
<b>Wiedza:</b>		
1. ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną w zakresie programowania systemowego i współbieżnego, oraz wiedzę szczegółową w zakresie funkcjonowania systemów operacyjnych - [K1st_W4]		
2. ma podstawową wiedzę o cyklu życia systemów operacyjnych, a w szczególności o zasadach zarządzania procesami, mechanizmach synchronizacji i przeciwdziałania zakleszczeniom - [K1st_W6]		
3. zna podstawowe techniki, metody oraz narzędzia wykorzystywane w procesie rozwiązywania zadań informatycznych, głównie o charakterze inżynierskim, z zakresu kluczowych zagadnień programowania systemowego i współbieżnego - [K1st_W7]		
<b>Umiejętności:</b>		

1. potrafi, formułując i rozwiązując zadania informatyczne, zastosować odpowiednio dobrane metody programowania systemowego i współbieżnego, w tym metody analityczne - [K1st\_U4]
2. potrafi ocenić złożoność obliczeniową algorytmów i problemów współbieżnych - [K1st\_U8]
3. potrafi - zgodnie z zadaną specyfikacją - zaprojektować (sformułować specyfikację funkcjonalną i wymagania pozafunkcyjne dla wybranych charakterystyk jakościowych) oraz zrealizować szeroko rozumiany system informatyczny, dobierając język programowania odpowiedni do danego zadania programistycznego oraz używając właściwych metod, technik i narzędzi programowania współbieżnego - [K1st\_U10]
4. ma umiejętność formułowania algorytmów współbieżnych i ich implementacji z użyciem przynajmniej jednego z popularnych narzędzi - [K1st\_U11]

#### Kompetencje społeczne:

1. rozumie, że w informatyce wiedza i umiejętności z zakresu programowania współbieżnego bardzo szybko stają się przestarzałe - [K1st\_K1]
2. ma świadomość znaczenia wiedzy z zakresu programowania systemowego i współbieżnego w rozwiązywaniu problemów inżynierskich oraz zna przykłady i rozumie przyczyny wadliwie działających systemów informatycznych, które doprowadziły do poważnych strat finansowych i społecznych - [K1st\_K2]

#### Sposoby sprawdzenia efektów kształcenia

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:

a)W zakresie wykładów sprawdzanie założonych efektów kształcenia realizowane jest przez:

- ocenę wiedzy i umiejętności wykazanych na egzaminie pisemnym o charakterze problemowym

W trakcie egzaminu student rozwiązuje pytania problemowe lub testowe; Za każde zadanie problemowe można uzyskać 10 punktów, za zadanie testowe 2 punkty; do zaliczenia egzaminu wymaga się uzyskania minimum 40% możliwych do zdobycia punktów.

b)W zakresie laboratoriów 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 ciągle, na każdych zajęciach (odpowiedzi ustne),

- ocenę sprawozdania przygotowywanego częściowo w trakcie zajęć, a częściowo po ich zakończeniu; ocena ta obejmuje także umiejętność pracy w zespole,

- ocenę wiedzy i umiejętności związanych z realizacją zadań laboratoryjnych poprzez kolokwia,

- ocenę wiedzy i umiejętności związanych z realizacją zadania projektowego poprzez realizację projektu w semestrze, realizowanego przez studenta jako praca domowa

Możliwe jest uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za:

- omówienia dodatkowych aspektów zagadnienia,

- efektywność zastosowania zdobytej wiedzy podczas rozwiązywania zadanego problemu

#### Treści programowe

W ramach wykładu przedstawiane są następujące zagadnienia:

1)Wprowadzane są elementy programowania współbieżnego (grafy przepływu procesów, oraz notacje "and", "fork-join-quit", "parbegin-parend")

2)Omawiany jest problem wzajemnego wykluczania oraz prezentowane i analizowane są przykładowe programowe sposoby jego rozwiązania, obejmujące m. in, algorytmy: Dekkera, Dijkstry, Petersona dla dwóch i n procesów, Lamporta

3)Definiowane są mechanizmy synchronizacji: sprzętowe (instrukcje test-and-set, aktywne czekanie, blokowanie systemu przerwań), systemowe (semafony binarne i ogólne, operacje lock i unlock, operacje enq i deq, operacja wait i post, operacje block i wakeup, liczniki zdarzeń), programowe (regiony krytyczne, warunkowe regiony krytyczne, monitory, implementacje programowych mechanizmów synchronizacji) i komunikacyjne (synchroniczne i asynchroniczne operacje wymiany komunikatów send i receive).

4)Przedstawiane są zastosowania wybranych mechanizmów synchronizacji do rozwiązywania klasycznych problemów synchronizacji (wzajemnego wykluczania, problemu producenta-konsumenta, problemu czytelników-pisarzy, problemu pięciu filozofów).

5)Omawiane jest zarządzanie procesami: pojęcie procesu, graf stanów procesów, problem szeregowania zadań w ujęciu probabilistycznym i deterministycznym (kryteria oceny uszeregowania), algorytmy szeregowania.

6)Wprowadzana jest definicja zakleszczenia, warunki konieczne i dostateczne zakleszczenia, przeciwdziałanie zakleszczeniom (podejście zapobiegania, unikania oraz detekcji i likwidacji).

Na zajęciach laboratoryjnych studenci implementują mechanizmy oferowane przez jądro systemu UNIX, ponadto konfrontują uzyskane w czasie wykładów wiadomości z praktyczną implementacją algorytmów i mechanizmów synchronizacji. W ramach laboratoriów omawiane są następujące zagadnienia:

1)Operacje na plikach zwykłych i przykłady zastosowania operacji plikowych z wykorzystaniem funkcji systemowych systemu UNIX

2)Obsługa procesów: tworzenie i usuwanie procesów, uruchamianie programów, przekierowania standardowych strumieni: wejścia, wyjścia i wyjścia diagnostycznego; przykłady użycia systemowych funkcji obsługi procesów

3)Tworzenie i obsługa łączy nazwanych i nienazwanych, przykłady błędów w synchronizacji procesów korzystających z łączy

4)Mechanizmy IPC: dostęp do pamięci współdzielonej, obsługa semaforów i kolejek komunikatów. Wykorzystanie poznanych mechanizmów do synchronizacji procesów; implementacja algorytmów poznanych na wykładzie z użyciem wybranych mechanizmów IPC

5)Obsługa i zarządzanie wątkami

Część wyżej wymienionych treści programowych jest realizowana w ramach pracy własnej studenta.

#### Literatura podstawowa:

1. Podstawy systemów operacyjnych, A. Silberschatz, J.L. Peterson, G. Gagne, WNT, W-wa, 2006
2. Podstawy programowania współbieżnego i rozproszonego, M. Ben-Ari, WNT, W-wa, 2009
3. Programowanie w systemie Unix dla zaawansowanych, Marc J. Rochkind, WNT, 2007

#### Literatura uzupełniająca:

1. Operating Systems. A Modern Perspective, G. Nutt, Addison Wesley Longman, Inc., 2002
2. Programowanie współbieżne i rozproszone w przykładach i zadaniach, Z. Weiss, T. Gruzlewski, WNT, W-wa, 1993
3. Programowanie współbieżne, W. Iszkowski, M. Maniecki, WNT, W-wa, 1982

#### Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. udział w zajęciach laboratoryjnych:	16
2. przygotowanie do ćwiczeń laboratoryjnych:	12
3. dokończenie (w ramach pracy własnej) sprawozdań z ćwiczeń laboratoryjnych:	12
4. udział w konsultacjach związanych z realizacją procesu kształcenia, w szczególności ćwiczeń laboratoryjnych	2
5. napisanie programu / programów, uruchomienie i weryfikacja (czas poza zajęciami laboratoryjnymi)	12
6. przygotowanie do kolokwium	16
7. udział w wykładach	25
8. zapoznanie się ze wskazaną literaturą / materiałami dydaktycznymi (10 stron tekstu naukowego = 1 godz.), 250 stron	10
9. przygotowanie do egzaminu i obecność na egzaminie: 8 godz. + 2 godz.	

#### Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	117	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	36	1
Zajęcia o charakterze praktycznym	52	2