KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Programowanie systemowe i współbieżne	**1010511331010510192**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**2 / 3**
Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
-	**polski**	**obligatoryjny**

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

Godziny	Liczba punktów
Wykłady: **30** Ćwiczenia: **-** Laboratoria: **30** 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:

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

Odpowiedzialny za przedmiot / wykładowca:

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

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z zakresu funkcjonowania systemów operacyjnych prezentowaną w ramach przedmiotu Systemy Operacyjne.
2	**Umiejętności:**	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	**Kompetencje społeczne**	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.

Cel przedmiotu:

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.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną w zakresie programowania systemowego i współbieżnego, oraz wiedzę szczegółową w zakresie zakresu 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]

Umiejętności:

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 pozafunkcjonalne 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 współpbieznego 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 ciągłe, 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 (semafory 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

Literatura podstawowa:

1. Operating Systems: Design and Implem., Tanenbaum A., Prentice-Hall Intern. Ed., 2008
2. Podstawy systemów operacyjnych, Silberschatz A., Galvin P.B., WNT, 2006
3. Operating System Concepts, 8th, Update Edition, Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Wiley&Sons, 2011
4. Program. w systemie Unix dla zaawansowanych, Marc J. Rochkind, WNT, 2008
5. System operacyjny LINUX, Cezary Sobaniec, Nakom, 2002
6. Unix i Linux. Przewodnik administratora systemów. Wydanie IV, E. Nemeth, i inni, WNT, 2011

Literatura uzupełniająca:

1. Operating Systems - A Modern Perspective, 3rd Edition , Nutt, G.J, Addison-Wesley Pub, 2003
2. Operating Systems, 3/E, Deitel I inni, Prentice Hall Intern, 2004
3. The Linux Programming Interface, Michael Kerrisk, No Starch Press, 2010
4. Distributed Content Dissemination with a Rank Function, A.Kobusinska, J Brzeziński, J Aftowicz, G Grzelachowski, CIT 2016
5. Advanced Programming in the Unix Environment (3rd Edition), R.Stevens, S.Rago, O'Reilly, 2013
6. Linux System Programming: Talking Directly to the Kernel and C Library, R. Love, O'Reilly, 2007
7. Linux Kernel Development, R. Love, Addison-Wesley, 2010
8. Operating Systems: Internals and Design Principles (8th Edition), Stallings W., Prentice Hall Intern, 2018

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. udział w zajęciach laboratoryjnych:	30
2. przygotowanie do ćwiczeń laboratoryjnych:	10
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)	10
6. przygotowanie do kolokwiów	10
7. udział w wykładach	30
8. zapoznanie się ze wskazaną literaturą / materiałami dydaktycznymi (10 stron tekstu naukowego = 1 godz.), 100 stron	10
9. przygotowanie do egzaminu i obecność na egzaminie: 8 godz. + 2 godz.	10

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	120	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	64	3
Zajęcia o charakterze praktycznym	62	2