**Nazwa przedmiotu:**

Systemy operacyjne 1

**Koordynator przedmiotu:**

Dr hab. inż. Leszek Jan Opalski, prof. PW, Mgr inż. Marcin Borkowski

**Status przedmiotu:**

Obowiązkowy

**Poziom kształcenia:**

Studia I stopnia

**Program:**

Informatyka i Systemy Informacyjne

**Grupa przedmiotów:**

Wspólne

**Kod przedmiotu:**

1030-IN000-ISP-0236

**Semestr nominalny:**

3 / rok ak. 2019/2020

**Liczba punktów ECTS:**

2

**Liczba godzin pracy studenta związanych z osiągnięciem efektów uczenia się:**

**Liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich:**

**Język prowadzenia zajęć:**

polski

**Liczba punktów ECTS, którą student uzyskuje w ramach zajęć o charakterze praktycznym:**

**Formy zajęć i ich wymiar w semestrze:**

|  |  |
| --- | --- |
| Wykład:  | 15h |
| Ćwiczenia:  | 0h |
| Laboratorium:  | 15h |
| Projekt:  | 0h |
| Lekcje komputerowe:  | 0h |

**Wymagania wstępne:**

Programowanie 1 – strukturalne
Programowanie 2 – obiektowe
Podstawy systemu UNIX

**Limit liczby studentów:**

Laboratorium (ćwiczenia komputerowe) – 15-24 os. /grupa

**Cel przedmiotu:**

Celem przedmiotu jest zapoznanie z budową i działaniem współczesnych systemów operacyjnych, a także kształtowanie umiejętności wykorzystania mechanizmów systemowych do poprawnej realizacji prostych aplikacji wieloprocesowych/wielowątkowych. Po ukończeniu kursu studenci:
1. posiadają wiedzę na temat:
- zadań, właściwości i budowy systemów operacyjnych,
- podstawowych modeli i sposobów realizacji współbieżności przez system komputerowy,
- planowania przydziału czasu procesora/procesorów,
- budowy i własności podstawowych typów systemów plików,
- podstawowych mechanizmów synchronizacji i komunikacji międzyprocesowej;
2. umieją zaprojektować, napisać w języku C i przetestować stworzoną przez siebie prostą aplikację o zadanej funkcjonalności, wykorzystując mechanizmy i funkcje systemowe (API POSIX).

**Treści kształcenia:**

Celem pierwszej części modułu "Systemy operacyjne" jest zapoznanie studentów z podstawami działania i konstrukcji współczesnych systemów operacyjnych, a także kształtowanie umiejętności wykorzystania funkcji systemowych do poprawnej realizacji prostych aplikacji wieloprocesowych/wielowątkowych, wykorzystujących standardowe kanały wejścia/wyjścia i dostęp do plików.
Wykład:
Wprowadzenie. System operacyjny, a system komputerowy. Zadania i interfejs systemów operacyjnych. Struktury systemów operacyjnych. Działanie systemu komputerowego.
Procesy i wątki. Koncepcje procesu, procesu lekkiego, włókna i wątku; cykl życia i interfejs programisty. Realizacja współbieżności procesów i wątków. Współprogramy (coroutines) i kontynuacje. Sygnały i wyjątki oraz ich obsługa. Funkcje systemowe POSIX związane z obsługą procesów, sygnałów i wątków.
Modele i interfejsy systemów plików. Plik, organizacja systemu plików. Operacje na systemie plików i interfejs programisty. Dostęp i ochrona plików, tryby dostępu do plików. Funkcje systemowe (POSIX) związane z obsługą synchroniczną i asynchroniczną plików i katalogów. Funkcje obsługi strumieni wejścia/wyjścia standardowej biblioteki języka C.
Laboratoria:
Wprowadzenie (1g). Środowisko wykonania programu POSIX (3g). Procesy i sygnały (3g). Wątki i muteksy (3g). Asynchroniczne operacje wejścia/wyjścia (3g). Indywidualna poprawa jednego ćwiczenia (2g).

**Metody oceny:**

Wykład:
Wykłady mają wprowadzić studentów w tematykę przedmiotu, w szczególności w tematykę laboratorium. Dlatego spotkania w blokach 2-godzinnych odbywają się w pierwszej części semestru. Nie ma sprawdzianów audytoryjnych (wykładowych). Treści wykładowe będą się jednak pojawiać na wejściówkach laboratoryjnych.
Laboratoria:
Do dyspozycji studenta są zajęcia wprowadzające (L0), 4 ćwiczenia oceniane (L1, L2, L3, L4) oraz zajęcia (L5) poświęcone poprawie oceny z laboratorium. Ćwiczenia L1, L2, L3 mają układ: wejściówka sprawdzająca wiedzę techniczna wymaganą na zajęciach (z wykładu i man'a), tutorial ukazujący praktyczne aspekty użycia wiedzy technicznej związanej z tematyką zajęć i na koniec zadanie programistyczne ("łatwe"). Sprawdzana jest poprawność działania opracowanego programu na podstawie osiągania przez studentów z góry wyznaczonych punktów kontrolnych. Student okazuje efekt działania programu na ekranie monitora, prowadzący może zażądać pokazania również kodu. Maksymalna liczba punktów za każde z ćwiczeń: 25.
Ćwiczenie L4 ma układ: wejściówka sprawdzająca poprawność rozumowania koncepcyjnego związanego z następującym zadaniem programistycznym ("trudnym"). W trakcie zajęć sprawdzane jest zaawansowanie wykonania zadania poprzez osiąganie poszczególnych punktów kontrolnych. Sprawdzeniu podlega nie tylko poprawność działania opracowanego programu ale i jakość kodu, który jest oceniany po zajęciach przez prowadzących. Maksymalna liczba punktów: 25.
Szczegółowy sposób wyliczenia punktów oceny będzie podany dla każdego zadania oddzielnie.
Rozwiązania wszystkich zadań (kod) podlegających ocenie muszą zostać przekazane prowadzącym w wymaganej formie.
Zajęcia L5 poświęcone są na poprawianie oceny. Student może wybrać jeden temat, który chciałby poprawić lub nadrobić. Forma zadania będzie zgodna z tematem, który poprawia student. Nie ma możliwości poprawiania ani nadrabiania więcej niż jednego tematu w semestrze.
Zajęcia odbywają się (bez przerw) wg harmonogramu. Zawartość grup laboratoryjnych oraz przydział grup do terminów harmonogramu zostanie uzgodniona z reprezentacją przed pierwszymi zajęciami.
Uwagi ogólne:
Wszystkie oceniane prace muszą być wykonywane samodzielnie. Niesamodzielność pracy, bądź korzystanie przez studenta z niedozwolonych materiałów powoduje uzyskanie z danej pracy/sprawdzianu 0p.
W czasie wykonywania ocenianych prac pisemnych (jak wejściówki laboratoryjne) nie można korzystać z żadnych materiałów pisanych, nagrań dźwiękowych, środków komunikacji elektronicznej.
W czasie realizacji tutoriali dozwolona (a wręcz zalecana) jest komunikacja studentów z prowadzącymi, a także pomiędzy sobą - jednak tak, by nie przeszkadzać osobom postronnym.
W czasie rozwiązywania ocenianych programistycznych zadań laboratoryjnych można korzystać z tutoriala, własnych materiałów, rozwiązań zadań przykładowych, dokumentacji systemowej oraz Internetu jednak tak, by oceniana praca była dziełem samodzielnym.
Punkty z wszystkich ocenianych elementów sumują się do 100. Do zaliczenia wymagane jest 50p, oceny wyższe są wyznaczane co 10p metodą proporcjonalną.

**Egzamin:**

nie

**Literatura:**

Podstawowa:
1. A. Silberschatz, P.B. Galvin, G. Gagne, Podstawy systemów operacyjnych, wyd. 6 zm. i rozsz., WNT 2005.
2. The GNU C Library Manual, http://www.gnu.org/software/libc/manual/.
Uzupełniająca:
1. W. Stallings, Systemy operacyjne. Struktura i zasady budowy, Wyd. Naukowe PWN SA, 2006.
2. K. Stencel, Systemy operacyjne, Wyd. PJWSTK, 2004.
3. U. Vahalia, Jądro systemu UNIX. Nowe horyzonty, WNT 2001.
4. W.R. Stevens, Programowanie w środowisku systemu UNIX, WNT 2002.
5. M.K. Johnson, E.W. Troan, Oprogramowanie użytkowe w systemie Linux, WNT 2000.
6. K. Haviland, D. Gray, B. Salama, Unix. Programowanie systemowe, Wyd. RM, Warszawa 1999.
7. K. Wall, Linux, programowanie w przykładach, MIKOM 2000.

**Witryna www przedmiotu:**

e.mini.pw.edu.pl

**Uwagi:**

## Charakterystyki przedmiotowe

### Profil ogólnoakademicki - wiedza

**Charakterystyka W01:**

Ma uporządkowaną, podbudowaną teoretycznie wiedzę nt. zadań, właściwości i budowy systemów operacyjnych

Weryfikacja:

wejściówki lab.

**Powiązane charakterystyki kierunkowe:** K\_W05

**Powiązane charakterystyki obszarowe:**

**Charakterystyka W02:**

Zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu prostych zadań z zakresu środowiska wykonania, procesów i sygnałów, wątków i operacji asynchronicznych

Weryfikacja:

wejściówki i zadania lab.

**Powiązane charakterystyki kierunkowe:** K\_W11, K\_W13

**Powiązane charakterystyki obszarowe:**

### Profil ogólnoakademicki - umiejętności

**Charakterystyka U01:**

Potrafi opisać architekturę co najmniej dwóch systemów dostępowych w sieciach teleinformatycznych

Weryfikacja:

zaliczenie zadania lab.

**Powiązane charakterystyki kierunkowe:** K\_U05, K\_U07

**Powiązane charakterystyki obszarowe:**

**Charakterystyka U02:**

Potrafi zrealizować w języku C proste aplikacje o zadanej funkcjonalności, wykorzystując narzędzia linii poleceń systemu Linux

Weryfikacja:

zaliczenie zadania lab.

**Powiązane charakterystyki kierunkowe:** K\_U11

**Powiązane charakterystyki obszarowe:**

**Charakterystyka U03:**

Potrafi wykorzystać nabytą wiedzę o własnościach SO oraz znajomość API POSIX do realizacji prostych aplikacji wieloprocesowych / wielowątkowych o zadanej funkcjonalności

Weryfikacja:

zaliczenie zadania lab.

**Powiązane charakterystyki kierunkowe:** K\_U11, K\_U15, K\_U18

**Powiązane charakterystyki obszarowe:**

### Profil ogólnoakademicki - kompetencje społeczne

**Charakterystyka K01:**

Rozumie konieczność ciągłego śledzenia zmian w obszarze systemów operacyjnych

Weryfikacja:

wejściówki lab.

**Powiązane charakterystyki kierunkowe:** K\_K01

**Powiązane charakterystyki obszarowe:**