

SYLABUS**DOTYCZY CYKLU KSZTAŁCENIA 2020/2021 – 2023/2024***(skrajne daty)*

Rok akademicki 2021/2022

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	Systemy operacyjne i architektura komputerów
Kod przedmiotu*	
Nazwa jednostki prowadzącej kierunek	Kolegium Nauk Przyrodniczych
Nazwa jednostki realizującej przedmiot	Kolegium Nauk Przyrodniczych
Kierunek studiów	Mechatronika
Poziom studiów	studia I-go stopnia
Profil	praktyczny
Forma studiów	studia niestacjonarne
Rok i semestr/y studiów	rok II, semestr 3
Rodzaj przedmiotu	przedmiot kierunkowy
Język wykładowy	język polski, język angielski
Koordinator	dr Krzysztof Balicki
Imię i nazwisko osoby prowadzącej / osób prowadzących	dr Krzysztof Balicki

* -opcjonalnie, zgodnie z ustaleniami w Jednostce

1.1. Formy zajęć dydaktycznych, wymiar godzin i punktów ECTS

Semestr (nr)	Wykł.	Ćw.	Konw.	Lab.	Sem.	ZP	Prakt.	Inne (jakie?)	Liczba pkt. ECTS
3	9			18					2

1.2. Sposób realizacji zajęć

- ☒ zajęcia w formie tradycyjnej
- ☒ zajęcia realizowane z wykorzystaniem metod i technik kształcenia na odległość

1.3 Forma zaliczenia przedmiotu (z toku) (egzamin, zaliczenie z oceną, zaliczenie bez oceny)

Wykład – zaliczenie bez oceny.

Laboratoria – zaliczenie z oceną.

2. WYMAGANIA WSTĘPNE

Znajomość programowania w języku C.

3. CELE, EFEKTY UCZENIA SIĘ, TREŚCI PROGRAMOWE I STOSOWANE METODY DYDAKTYCZNE

3.1 Cele przedmiotu

C ₁	Zapoznanie studentów z zasadami działania i budowy współczesnych systemów operacyjnych (Windows, Linux), ich możliwości i funkcji oferowanych użytkownikom.
C ₂	Nabycie umiejętności korzystania z systemu Linux na poziomie powłoki oraz wiersza poleceń systemu Windows.
C ₃	Zapoznanie słuchaczy z architekturą współczesnych systemów komputerowych ze szczególnym uwzględnieniem procesora i koprocesora arytmetycznego.
C ₄	Nabycie umiejętności programowania niskopoziomowego procesora i koprocesora arytmetycznego.

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu Student:	Odniesienie do efektów kierunkowych ¹
EK_01	ma podstawową wiedzę z zakresu systemów operacyjnych, ich zadań oraz funkcji oferowanych użytkownikom	K_Wo8
EK_02	ma wiedzę z zakresu budowy procesorów rodziny Intel, koprocesora arytmetycznego i jednostek wektorowych; rozumie sprzętowe mechanizmy ochrony wykorzystywane we współczesnych systemach operacyjnych	K_Wo9
EK_03	potrafi pozyskiwać informacje z dokumentacji: Linuxa, procesora, koprocesora, jednostek wektorowych, bibliotek programistycznych oraz innych źródeł; potrafi integrować uzyskane informacje, dokonywać ich interpretacji, a także wyciągać wnioski oraz formułować i uzasadniać opinie; potrafi wykorzystać pozyskane informacje do rozwiązywania problemów programistycznych	K_Uo1
EK_04	potrafi ocenić przydatność metod i narzędzi służących do rozwiązania zadań programistycznych, w tym dostrzec ograniczenia tych metod i narzędzi	K_Uo8
EK_05	potrafi posłużyć się właściwie dobranymi metodami i narzędziami, aby zbadać sposób reprezentacji liczb w pamięci, ulokowanie kodu programu, zmiennych globalnych, sterty i stosu dla programu w języku C; potrafi narysować mapę pamięci dla procesu	K_U16
EK_06	potrafi stosować wzory matematyczne do wyznaczania reprezentacji liczb w pamięci, konwersji liczb między różnymi systemami liczbowymi; potrafi rozwiązywać równania z arytmetyką modularną	K_U19

¹ W przypadku ścieżki kształcenia prowadzącej do uzyskania kwalifikacji nauczycielskich uwzględnić również efekty uczenia się ze standardów kształcenia przygotowującego do wykonywania zawodu nauczyciela.

EK_07	rozumie potrzebę i możliwości ciągłego dokształcania się (studia drugiego i trzeciego stopnia, studia podyplomowe, kursy) – podnoszenia kompetencji zawodowych, osobistych i społecznych	K_Ko1
-------	--	-------

3.3 Treści programowe

A. Problematyka wykładu

Treści merytoryczne
Ogólna zasada działania systemu operacyjnego.
Przykładowe polecenia powłoki systemowej Linux.
Procesy, zasoby i wątki. Ochrona pamięci.
Tryb jądra i użytkownika. Wywołania systemowe.
Wprowadzenie – system komputerowy, klasyfikacja architektur komputerowych, hierarchia pamięci. Maszyna von Neumanna, architektury Harvard, Princeton, Harvard-Princeton.
Dane – typy, reprezentacje, organizacja i adresowanie pamięci. Porządek Big-Endian i Little-Endian. Wyrównanie naturalne. Dane wektorowe.
Budowa modelu programowego — rejestry, tryby adresowania, model operacji warunkowych, lista instrukcji. Porównanie modelu programowego w podejściu CISC i RISC.
Model programowy procesorów 16, 32 i 64 bitowych rodziny Intel.
Arytmetyka stała i zmiennopozycyjna. Jednostki zmiennopozycyjne i wektorowe.
Wybrane konwencje wywołań dla kodu 32-bitowego.
Konwencje wywołań dla kodu 64-bitowego.
Analiza stanu stosu procesora i koprocesora arytmetycznego.
Analiza programów rekurencyjnych i nierekurencyjnych w języku C i assemblerze.
Mierzenie czasu wykonania procedur i funkcji.
Łączenie kodu w assemblerze z programami w języku C.
Assembler NASM, asmloder, disassembler, debugger GDB, linker.

B. Problematyka laboratoriów

Treści merytoryczne
Podstawowe komendy wiersza poleceń systemu Windows.
Obsługa i konfiguracja programu Oracle VM VirtualBox.
Instalacja i konfiguracja systemu Linux w maszynie wirtualnej.
Podstawowe polecenia powłoki Unix/Linux.
Operacje na plikach i katalogach.
Potoki i filtry.
Koncepcja bezpieczeństwa w systemach Unixowych.
Praca z edytorem VI.
Obsługa procesów.
Konwersje liczb między różnymi systemami liczbowymi.
Reprezentacja danych, konwencje little-endian i big-endian.
Mapa pamięci procesu w języku C.
Obsługa programów: NASM, asmloder, linker, disassembler, debugger DBG.
Pseudoinstrukcje assemblera NASM.
Operacje przesłań.

Tryby adresowania.
Operacje arytmetyczne, logiczne, porównań, bitowe, przesunięcia i rotacje bitów.
Instrukcje skoków, skoków warunkowych, wywołania procedur.
Analiza stanu stosu procesora i koprocesora arytmetycznego.
Analiza programów rekurencyjnych i nierekurencyjnych w języku C i assemblerze.

3.4 Metody dydaktyczne

Wykład: wykład problemowy, wykład z prezentacją multimedialną.

Laboratoria: rozwiązywanie zadań, dyskusja.

4. METODY I KRYTERIA OCENY

4.1 Sposoby weryfikacji efektów uczenia się

Symbol efektu	Metody oceny efektów uczenia się (np.: kolokwium, egzamin ustny, egzamin pisemny, projekt, sprawozdanie, obserwacja w trakcie zajęć)	Forma zajęć dydaktycznych (w, ćw, ...)
EK_01	kolokwium	lab.
EK_02	obserwacja w trakcie zajęć, dyskusja	lab.
EK_03	kolokwium, projekt	lab.
EK_04	obserwacja w trakcie zajęć, projekt	lab.
EK_05	obserwacja w trakcie zajęć	lab.
EK_06	obserwacja w trakcie zajęć	lab.
EK_07	obserwacja w trakcie zajęć	lab.

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

Wykład

Zaliczenie bez oceny.

Zaliczenie wykładu na podstawie pozytywnej oceny z zajęć laboratoryjnych. Uzyskanie pozytywnej oceny z zajęć laboratoryjnych uwzględnia sprawdzenie wiedzy wykładowej - efekty EK_01, EK_02 weryfikowane są na laboratoriach.

Laboratoria

Warunkiem zaliczenia laboratoriów jest zaliczenie kolokwium z systemów operacyjnych oraz kolokwium z architektury komputerów, a także wykonanie projektu programistycznego.

Ocena końcowa jest średnią ocen z dwóch kolokwiów i projektu programistycznego.

Oceny z kolokwiów przyznawane są proporcjonalnie do liczby zdobytych punktów.

Pod uwagę brana jest również aktywność na zajęciach, która może obniżyć lub podwyższyć ocenę końcową o pół stopnia.

5. CAŁKOWITY NAKŁAD PRACY STUDENTA POTRZEBNY DO OSIĄGNIĘCIA ZAŁOŻONYCH EFEKTÓW W GODZINACH ORAZ PUNKTACH ECTS

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe wynikające z harmonogramu studiów	27
Inne z udziałem nauczyciela akademickiego	2

(udział w konsultacjach, egzaminie)	
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	31
SUMA GODZIN	60
SUMARYCZNA LICZBA PUNKTÓW ECTS	2

** Należy uwzględnić, że 1 pkt ECTS odpowiada 25-30 godzin całkowitego nakładu pracy studenta.*

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	Nie dotyczy
zasady i formy odbywania praktyk	Nie dotyczy

7. LITERATURA

Literatura podstawowa: [1] Abraham Silberschatz, Peter B. Galvin, Greg Gagne: Podstawy Systemów Operacyjnych. wyd. 10, PWN 2021. [2] The Linux man-pages project - www.kernel.org/doc/man-pages [3] Abraham Silberschatz: Podstawy Systemów Operacyjnych. wyd. 7, WNT 2006. [4] D. Patterson, J. Hennessy: Computer Organization and design. Elsevier 2005. [5] J. Biernat: Architektura komputerów, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2005.
Literatura uzupełniająca: [1] Andrew S. Tanenbaum, Herbert Bos: Systemy operacyjne. Wyd IV, Helion, 2016. [2] Christopher Negus: Linux. Biblia. Ubuntu, Fedora, Debian i 15 innych dystrybucji, Helion, 2011. [3] P. Metzger, M. Siemieniecki: Anatomia PC: architektura komputerów zgodnych z IBM PC. Helion, Gliwice 2003. [4] Materiały firmowe - dokumenty techniczne dostępne w sieci WWW – MIPS, Intel, AMD. [5] Specyfikacje: Application Binary Interface.

Akceptacja Kierownika Jednostki lub osoby upoważnionej