Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ЦТЕ

Кафедра цифрових
технологій в енергетиці

ОПЕРАЦІЙНА СИСТЕМА UNIX

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	Перший (бакалаврський)
Галузь знань	12 Інформаційні технології
Спеціальність	122 Комп'ютерні науки
Освітня програма	Цифрові технології в енергетиці
Статус дисципліни	Нормативна
Форма навчання	очна(денна)
Рік підготовки, семестр	2 курс осінній семестр
Обсяг дисципліни	На засвоєння дисципліни передбачено 90 год / 3 кредити ЄКТС, з них - 18 год. лекції, 18 год. практичні, 18 год. лабораторні,36 год. самостійна робота
Семестровий контроль/ контрольні заходи	Залік, МКР
Розклад занять	Науково-педагогічний працівник
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор: д.т.н., професор, Левченко Лариса Олексіївна, levchenko_larisa@lll.kpi.ua, тел. 097-068-11-42 Практичні та лабораторні: д.т.н., професор, Левченко Лариса Олексіївна, levchenko_larisa@lll.kpi.ua, тел. 097-068-11-42
Розміщення курсу	Кампус

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

В цій дисципліні детально вивчається UNIX-подібна операційна система Linux, а саме, архітектура системного програмного забезпечення операційно системи (ОС) Linux. ОС - це окрема реалізація всіх основних принципів побудови багатозадачних, багатокористувацьких та універсальних ОС, які закладалися при створенні UNIX в далекі 60-х роки. Код ядра Linux не має нічого спільного з UNIX, але реалізує той же набір системних викликів з аналогічною специфікацією. На відміну від ОС UNIX, яка є комерційним продуктом, ОС Linux є повністю безкоштовним продуктом, який надійно працює на будь-якому обладнанні та максимально використовує усі апаратні можливості. Крім того, Linux підтримується і вдосконалюється величезним та відкритим співтовариством розробників, а найефективніші, передові та оригінальні ідеї яких реалізуються згодом впроваджуються у ядро.

На сьогоднішній день затребувані фахівці, які впроваджують новітні технології, в основі яких лежить саме Linux. Отримані знання дозволяють виконувати роботи з адміністрування операційної системи, застосування технології контейнерів для розгортання, поширення та функціонування програмного забезпечення, створення мікросервісів, а також як результат будуть сприяти кар'єрному зростанню, самореалізації, застосуванню отриманих знань для вирішення практичних завдань.

Метою дисципліни є опанування студентами теоретичних знань архітектури системного програмного забезпечення, побудови, функціонування, використання засобів операційної системи Linux, а також опанування технології контейнерів для реалізації упаковки, розгортання та функціонування програмного забезпечення.

Предмет дисципліни - вивчення принципів побудови, архітектури, основних функцій, режимів роботи, засобів ОС Linux, розроблення мікросервісів на основі технології Docker.

Завдання. В результаті вивчення дисципліни у студентів повинні сформуватися наступні компетентності:

загальні:

- здатність до абстрактного мислення, аналізу та синтезу (ЗК 1),
- здатність застосовувати знання у практичних ситуаціях (ЗК 2),
- здатність діяти на основі етичних міркувань (ЗК 13).

фахові:

- здатність до інтелектуального аналізу даних на основі методів обчислювального інтелекту включно з великими та погано структурованими даними, їхньої оперативної обробки та візуалізації результатів аналізу в процесі розв'язування прикладних задач (ФК 11);
- здатність забезпечити організацію обчислювальних процесів в інформаційних системах різного призначення з урахуванням архітектури, конфігурування, показників результативності і функціонування операційних систем і системного програмного забезпечення (ФК 12);
- здатність до розробки мережевого програмного забезпечення, що функціонує на основі різних топологій структурованих кабельних систем, використовує комп'ютерні системи і мережі передачі даних та аналізує якість роботи комп'ютерних мереж (ФК 13).

Після засвоєння навчальної дисципліни студенти мають продемонструвати такі *програмні результати навчання:*

- застосовувати знання основних форм і законів абстрактно-логічного мислення, основ методології наукового пізнання, форм і методів вилучення, аналізу, обробки та синтезу інформації в предметній області комп'ютерних наук (ПР 1),
- використовувати мережні технології, архітектури комп'ютерних мереж, мати практичні навички технології адміністрування комп'ютерних мереж та їх програмного забезпечення (ПР 14).

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Пререквізити дисципліни. Знання, отримані при вивченні дисциплін: «Системи баз даних», «Програмування алгоритмічних структур», «Алгоритмізація і програмування», «Операційні системи».

Постреквізити дисципліни. Отримані знання при вивченні дисципліни «Операційна система UNIX» формує базові знання для вивчення наступних дисциплін: «Проектування інформаційних систем», «Комп'ютерні мережі», «Безпека інформаційних систем», «Комп'ютерна схемотехніка і архітектура комп'ютерів», які викладаються в наступних семестрах. Компетенції, отримані студентами в процесі вивчення цієї дисципліни, використовуються ними при виконанні дипломної роботи.

3. Зміст навчальної дисципліни

Розділ 1. Призначення, функції та архітектура операційної системи Linux.
Розділ 2. Управління Linux-пакетами
Розділ 3. Файлова система Linux.
Розділ 4. Управління процесами
Розділ 5. Управління пам'яттю
Розділ 6. Застосування технології контейнерів

4. Навчальні матеріали та ресурси

Основна література

1. Архітектура системного програмного забезпечення [Електронний ресурс] : підручник для студ. спеціальності 121 «Інженерія програмного забезпечення» / Л. О. Левченко, Н. Г. Кучук,

Ю.А. Тарнавський, В. П. Колумбет; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 6,6 Мбайт). Київ : КПІ ім. Ігоря Сікорського, 2022. – 499 с.
https://ela.kpi.ua/handle/123456789/49759
2. Nemeth Evi, Snyder Garth, Trent R. Hein, Whaley Ben. Unix and Linux System Administration Handbook. Publisher: Addison-Wesley Professional. 2020. 1168 P.
3. Kerrisk Michael. The Linux Programming Interface: A Linux and UNIX Programming Handbook. Publisher: No Starch Press. 2019. 1248 P.
4. Love Robert. Linux Kernel Development. Publisher: Addison-Wesley. 2020. 480 P.
5. Shotts William. The Linux Command Line: A Complete Introduction. Publisher: No Starch Press, Incorporated. 2020. 480 P.
6. Mouat Andrian. Using Docker: Developing and Deploying Software with Containers. Publisher: O'Relly Media. 2017. 354 P.
7. Kocher Parminder Singh. Microservices and Containers. Publisher: Addison-Wesley Professional 2019. 240 P.
8. Miell Ian, Sayers Aidin Hobson. Docker in Practice. Publisher: Manning. 2020. 516 P.

Додаткова література

9. Ward Brian. How Linux Works: What Every Superuser Shoud Know. Publisher: No Starch Press 2021. 467 P.
10. Negus K., Kaen F. Ubuntu and Debian Linux for advanced more than 1000 essential commands. Publisher: Wiley. 2014. 384 P.
11. Clinton David. Linux in Action. Publisher: Manning. 2018. 384 P.
12. Blum R., Bresnahan Ch. Linux Essentials. Publisher : John Wiley & Sons; 2nd edition. 2017. 368 P.
13. Taylor D., Perry P. Wicked Cool Shell Scripts: 101 Scripts for Linux, OS X, and Unix Systems Paperback. Publisher : No Starch Press; 2nd edition. 2015. 392 P.
14. Robbins A. Bash Pocket Reference: Help for Power Users and Sys Admins. Publisher : O'Reilly Media; 2 edition. 2016. 153 P.
15. Sayers Aidan Hobson, Miell Ian. Docker in Practice. Publisher: Manning. 2019. 384 P.
16. Sayfan G. Mastering Kubernetes: Level up your container orchestration skills with Kubernetes to build, run, secure, and observe large-scale distributed apps. Publisher : Packt Publishing; 3rd edition. 2020. 642 P.
17. Schenker Gabriel N. Learn Docker - Fundamentals of Docker 19.x: Build, test, ship, and run containers with Docker and Kubernetes, 2nd Edition. Publisher: Packt Publishing. 2020. 592 P.
18. Richardson Ch. Microservices Patterns: With examples in Java. Publisher : Manning; 1st edition. 2018. 520 P.
19. Horsdal Ch. Microservices in .NET. Publisher : Manning; 2nd edition. 2021. 328 P.

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Розділ 1. Призначення, функції та архітектура операційної системи Linux. Управління Linux-пакетами

Лекція 1. Концепції програмування в Linux. Завантаження і установка інструментарію для роботи з відкритим вихідним кодом

Історія виникнення Linux. Стандарт POSIX. Дистрибутиви Linux (Debian, Ubuntu, Red Hat, SUSE, Solaris, HP-UX и AIX) та їх характеристика. Огляд Linux, системного програмування. Ядро, конфігурування параметрів ядра, методи конфігурування ядра. Драйвери пристроїв. Файли і файлова система. Процеси. Користувачі і групи. Права доступу. Сигнали. Міжпроцесна взаємодія. Заголовки. Обробка помилок. Пошук інструментарію. Формати поширення. Архівні файли.

Менеджери пакетів та їх характеристика. Що краще: бінарні пакети чи пакети з початковими кодами? Диспетчер пакетів RPM (Red Hat Package Manager), команда rpm. Огляд пакетів RPM. Управління пакетами .deb в системі Ubuntu (команда dpkg). Оновлення пакетів: Інструмент Apt: Advanced Package Tool. Пакети tarball. Менеджер пакетів Aptitude. Synaptic: передовий GUI-інтерфейс для APT. Огляд пакетів RPM (Red Hat Package Manager). Інструмент Yum (Yellowdog Updater Modified). Перевірка автентичності. Система контролю версій проектів з відкритим вихідним кодом - GIT.

Розділ 2. Файлова система Linux

Лекція 2. Файлова система Linux, особливості роботи

Організація файлової системи. Стандарт FHS (Filesystem Hierarchy Standard). Стандартні підкаталоги та їх вміст. Абсолютні і відносні шляхи до файлу. Типи файлів та їх характеристики: звичайні (-), каталоги (d), файли байт-орієнтованих символьних пристроїв (c), файли блочно-орієнтованих пристроїв (b), локальні сокети (sokets), іменовані канали (pipe), символічні посилання. Атрибути файлів. Біти режиму. Отримання інформації про файл – системний виклик stat().Перенаправлення введення-виведення. Файловий менеджер mc (Midnight Commander).

Оболонка bash. Користувачі (UID) і групи (GID). Права доступу для файлів та для каталогів. Редагування прав доступу - команда chmod. Команда ls – перегляд атрибутів. Додаткові атрибути файлів (SUID, SGID). Робота з привілеями root – sudo. Зміна власника файлу і групи. Команда отримання відомостей про ім'я користувача (whoami). Команди для роботи з файлами (man, cat, touch, ln, cp, mv, head, tail, less) та каталогами (cd, ls, mkdir, rm, cp, tree). Система контролю версій програмного забезпечення Git.

Лекція 3. Фільтрація даних у файлі. Низькорівневе-введення виведення. Сценарії в оболонці bash

Утиліта grep – потужний інструмент системного адміністратора. Приклади роботи. Базові регулярні вирази. Розширений глобальний вираз – egrep. Модель введення-виведення в системах UNIX. Системні виклики (open(), read(), write(), close(), lseek()).

Скрипт. Створення сценарію. Перетворення сценарію у виконуваний файл. Змінні середовища та користувача. Підстановка команд. Математичні операції. Управляючі конструкції if-then/if-then-else. Оператори циклу while/until/for. Функції. Виклик функції з аргументами. Глобальні та локальні змінні в функціях. Передача параметрів у командному рядку. Передача функції масива в якості аргумента. Приклади сценаріїв.

Розділ 3. Управління процесами, потоками, пам’яттю

Лекція 4. Управління процесами, системні виклики. Управління процесами в оболонці Bash

Програми, процеси і потоки. Виділення ідентифікатора процесу. Ідентифікатор дочірнього процесу Ієрархія процесів. pid_t. Отримання ідентифікаторів батьківського процесу (PID) і дочірнього процесу PPID. Привілейовані процеси. Запуск нового процесу: Сімейство викликів exec. Системні виклики fork(). Завершення процесу: Інші способи завершення. atexit(), on_exit(). SIGCHLD. Очікування завершених дочірніх процесів: Очікування певного процесу. Ще більше гнучкості при очікуванні. Стиль BSD: wait3() і wait4(). Запуск і очікування нового процесу. Зомбі.

Системні процеси. Демони. Міжпроцесна взаємодія (канали, сигнали, сокети). Фонові процеси. Таблиця процесів. Моніторинг процесів. Управління сигналами.

Лекція 5. Потоковість. Управління пам'яттю

Бінарні модулі, процеси і потоки. Багатопоточність: витрати багатопоточності. Альтернативи багатопоточності. Поточні моделі: Поточність на рівні користувача. Комбінована поточність. Співпрограми і фібери. Шаблони поточності: потік на з'єднання. Потік, керований подією.

Конкурентність, паралелізм і гонки. Синхронізація: м’ютекси. Взаємні блокування. Р-потоки. Реалізація поточності в Linux. API для роботи з Р-потоками. Зв’язування Р-потоків. Створення потоків. Ідентифікатори потоків. Завершення потоків. Самозавершення. Завершення інших потоків. Приєднання і від’єднання потоків: Приєднання потоків. Від’єднання потоків. Приклад поточності. М’ютекси Р-потоків: Ініціалізація м’ютексів. Запирання м’ютексів. Відмикання м’ютексів. Приклад використання м’ютексів.

Адресний простір процесу: Сторінки та їх підкачка. Області пам’яті. Виділення динамічної пам’яті: Виділення масивів. Зміна розміру виділених областей. Звільнення динамічної пам’яті. Вирівнювання. Управління сегментом даних. Анонімні відображення в пам’яті: Створення анонімних відображень в пам’яті. Відображення /dev/zero. Розширене виділення пам’яті. Налагодження при операціях виділення пам’яті. Виділення пам'яті на основі стека: дублювання рядків в стеці. Масиви змінної довжини. Вибір механізму виділення пам’яті.

Управління пам’яттю: Установка байтів. Порівняння байтів. Переміщення байтів. Пошук байтів. Переклацування байтів. Блокування пам’яті: Блокування частини адресного простору. Блокування всього адресного простору. Розблокування пам’яті. Ліміти блокування. Чи знаходиться сторінка у фізичній пам’яті? Поступатися виділенням пам’яті.

Розділ 4. Застосування технології контейнерів

Лекція 6. Розробка та впровадження програмного забезпечення за допомогою технології контейнерів

Контейнери та їх призначення. Відмінності між віртуальними машинами та контейнерами. Docker і контейнери. Коротка історія Docker. Архітектура Docker. Базові технології Docker. Інструментальні засоби Docker. Установка Docker в ОС Linux.. Опції, основні команди управління, підкоманди Docker.

Лекція 7. Практична робота з Docker

Хостинг для Docker. Образи, реєстр образів. Команди для роботи з реєстром. Рівні образу контейнера. Команди управління контейнерами. Робота з образами. Приклади роботи з Docker. Базові образи Dockerfile, Docker Compose. Інструкції Dockerfile. Команди для роботи з Docker Compose. Встановлення зв’язку контейнерів із зовнішнім світом. Команда run. З’єднання між контейнерами.

Лекція 8. Життєвий цикл програмного забезпечення при використанні Docker.

Використання Docker в процесі розробки: Традиційне вітання світу. Життєвий цикл контейнера Автоматизація з використанням Compose. Порядок роботи Compose. Установка Docker Compose в Ubuntu 18.04. Створення простого веб-застосування: створення основної веб-сторінки. Приклад роботи Docker з БД Mongo.

Лекція 9. Мікросервіси і контейнери

Мікросервіси Оркестрація, кластеризація і управління: інструментальні засоби кластеризації і оркестрації. Swarm. Fleet. Kubernetes.

6. Самостійна робота студента

Розділ 1. Призначення, функції та архітектура операційної системи Linux

Історія створення Linux, принципи, філософія, стандарти та нормативна база Linux

Створення завантажувальної флешки UBUNTU у середовищі операційної системи Windows

Центр застосувань Gnome Software для роботи з Deb та Rpm пакетами. Робота з програмою APPGRID.

Встановлення ОС Linux на MAC OS.

Розділ 2. Файлова система Linux

Операції, які не вписуються у загальну модель універсального введення-виведення *iocl().* Архівація даних. Редагування файлів.

Розділ 3. Управління процесами, потоками, пам'яттю

Ідентифікація процесів. Моніторинг дочірніх процесів

Операції з віртуальною пам'яттю. Рівень блочного розподілу пам'яті.

Розділ 4. Застосування технології контейнерів

Мікросервіси. Оркестрація, кластеризація і управління: інструментальні засоби кластеризації і оркестрації. Swarm. Fleet. Kubernetes.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Відвідування лекційних та лабораторних занять є обов'язковим за винятком поважних причин (хвороби, форс-мажорних обставин).

В разі пропущення занять з поважних причин викладач надає можливість студенту виконати усі або деякі лабораторні завдання (винятком є виконання деяких завдань у зв'язку із закінченням навчального процесу).

В разі пропущення занять без поважних причин, а також через порушення граничного терміну виконання завдання (deadline) студент може отримати 80% від максимальної оцінки відповідне завдання.

Протягом семестру студенти:

- виконують та захищають лабораторні роботи у відповідні терміни (на кожну лабораторну роботу відводиться два тижні для здачі),
- пишуть модульну контрольну роботу,
- повинні позитивно закрити дві атестації (в кінці жовтня та на початку грудня),
- по закінченні навчального процесу складають залік.

8. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Система рейтингових (вагових) балів та критерії оцінювання

1) Робота на лекціях

На лекціях може бути проведено бліцопитування студентів. Такі опитування проводяться на довільних лекціях 5 разів протягом семестру, наприкінці лекції. Ваговий бал за вірну відповідь - 1. Максимальна кількість балів, що може отримати кожен студент за семестр - 5.

2) Максимальна кількість балів за усі виконані практичні та лабораторні роботи дорівнює 50 балів. Розподіл балів серед практичних та лабораторних робіт наступний:

Практичні роботи

№ з/п	Назва практичної роботи	Кількість балів
1	Установка Ubuntu Linux на віртуальній машині VirtualBox	4
2	Менеджери для роботи з пакетами програм Linux	5
3	Робота в оболонці Bash, середовище оточення	5
4	Робота з файловою системою ОС Linux	5
Всього:		19

3) Лабораторні роботи

№	Назва лабораторної роботи	Кількість

з/п		балів
1	Створенні сценаріїв в оболонці Bash	5
2	Робота з процесами ОС Linux	5
3	Потоки, перенаправлення потоків	6
4	Установка Docker	5
5	9.1. Створення проекту для web-розробки, який складається з наборів контейнерів з використанням Docker Compose та Dockerfile. 9.2. Створення проекту, що складається з контейтерів Django+PostgreSQL, з використанням Docker Compose та Dockerfile	10
Всього:		31

Критерії оцінювання: *Виконання практичних та лабораторних робіт:*

- виконана своєчасно (протягом двох тижнів з моменту видачі), у повному обсязі – відповідний бал згідно номеру практичної або лабораторної роботи з урахуванням вимог до їх оформлення;
- виконаний із запізненням – знімається 10 – 30% від максимальної кількості балів в залежності від терміну запізнення;
- виконана не самостійно, із запізненням – знімається 50% від максимальної кількості балів;
- невиконана протягом відведеного часу – 0 балів.

4) Модульна контрольна робота

Максимальна кількість балів за модульну контрольну роботу дорівнює 10 балів.

Якість виконання роботи:

- усі відповіді вірні та повні –10 балів,
- у відповідях допущені несуттєві неточності – 8 балів,
- половина відповідей вірна – 5 балів,
- відповіді з суттєвими неточностями, але без критичних помилок – 2 бали,
- менше половини відповідей вірна – 0 балів.

Штрафні та заохочувальні бали за:

- активність на практичних або лабораторних заняттях + 2 бали
- виконання практичної або лабораторної роботи з використанням власного оптимального алгоритму + 1 бали
- відсутність на занятті без поважної причини – 2 бали
- несвоєчасна здача практичної або лабораторної роботи (пізніше ніж за тиждень) – 0,5 балів;

5) Складання заліку

Максимальний ваговий бал $r_{зал}$=35

Умови позитивної проміжної атестації.

Для отримання „зараховано” з першої проміжної атестації студент матиме не менше ніж 11 балів (за умови, що за 8 тижнів згідно з календарним планом контрольних заходів „ідеальний” студент має отримати 5+5+5+5 = 20 балів).

Для отримання „зараховано” з другої проміжної атестації студент матиме не менше ніж 25 балів (за умови, що за 14 тижнів згідно з календарним планом контрольних заходів „ідеальний” студент має отримати 20 + 5+5+8+6+6 =50 балів).

Умови допуску до заліку.

Необхідною умовою допуску до диференційованого заліку є зарахування усіх практичних та лабораторних робіт, виконання модульної контрольної роботи, а також стартовий рейтинг (R_c) не менше 40 балів. Для отримання заліку з кредитного модуля "автоматом" потрібно мати рейтинг не менш ніж 60 балів, а також зараховане виконання всіх завдань лабораторних робіт.

Розрахунок шкали (R) рейтингу:

Сума вагових балів контрольних заходів протягом семестру (шкала рейтингу) складає:

$$R = r_{лек} + r_{прак} + r_{мод} + r_{зал} = 5+50 + 10 + 35 = 100 \text{ балів.}$$

Стартовий рейтинг становить $R_c = r_{лек} + r_{лаб} + r_{мод} = 65$ балів.

Рейтинг заліку дорівнює 35 балів.

Таким чином, рейтингова шкала з кредитного модуля складає

$$R = 65+35=100 \text{ балів.}$$

Для отримання студентом відповідних оцінок рейтингова оцінка студента переводиться згідно таблиці:

Бали	**Оцінка**
95 - 100	Відмінно
85 - 94	Дуже добре
75 - 84	Добре
65 - 74	Задовільно
60 - 64	Достатньо
Менше 60	Незадовільно
R < 40 є незараховані роботи комп'ютерного практикуму або не виконані інші умови допуску до екзамену	Не допущено

Робочу програму навчальної дисципліни (силабус):

Складено професор, д.т.н., професор, Левченко Лариса Олексіївна

Ухвалено кафедрою ЦТЕ (протокол № _20__ від _10.05.2023 р.)

Погоджено Методичною комісією ННІАТЕ КПІ ім. Ігоря Сікорського (протокол № 9 від _26.05.2023 р.)