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

ЦТЕ

Кафедра цифрових технологій в енергетиці

СИСТЕМНЕ ПРОГРАМУВАННЯ

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 120 год / 4 кредити ЄКТС, з них - 36 год. лекції, 18 год. лабораторні, 66 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 Лабораторні: д.т.н., професор, Левченко Лариса Олексіївна, levchenko_larisa@lll.kpi.ua, тел. 097-068-11-42
Розміщення курсу	Кампус

• Програма навчальної дисципліни

• 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 застосовувати знання у практичних ситуаціях (ЗК 2);

фахові:

- здатність до системного мислення, застосування методології системного аналізу для дослідження складних проблем різної природи, методів формалізації та розв'язування системних задач, що мають суперечливі цілі, невизначеності та ризики (ФК 06);
- 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).

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

- володіти мовами системного програмування та методами розробки програм, що взаємодіють з компонентами комп'ютерних систем. Використовувати мережні технології, архітектури комп'ютерних мереж, мати практичні навички технології адміністрування комп'ютерних мереж та їх програмного забезпечення (ПРН 13).

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

Пререквізити дисципліни. Знання, отримані при вивченні дисциплін: Математичний аналіз» «Алгоритмізація і програмування», «Комп'ютерна схемотехніка та архітектура комп'ютера», «Операційні системи», «Програмування алгоритмічних структур».

Постреквізити дисципліни. Отримані знання при вивченні дисципліни «Системне програмування» формує базові знання для вивчення наступних дисциплін: «Геоінформаційні системи в енергетиці», «Комп'ютерні мережі», «Безпека інформаційних систем», «Технології розробки програмного забезпечення», які викладаються в наступних семестрах. Компетенції, отримані студентами в процесі вивчення цієї дисципліни, використовуються ними при виконанні дипломної роботи.

- **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, потоками.
Розділ 5. Управління пам'яттю.
Розділ 6. Застосування технології контейнерів.
Розділ 7. Операційні системи сімейства Windows.

- **Навчальні матеріали та ресурси**

Основна література

- 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
- Nemeth Evi, Snyder Garth, Trent R. Hein, Whaley Ben. Unix and Linux System Administration Handbook. Publisher: Addison-Wesley Professional. 2020. 1168 P.
- Kerrisk Michael. The Linux Programming Interface: A Linux and UNIX Programming Handbook. Publisher: No Starch Press. 2019. 1248 P.
- Love Robert. Linux Kernel Development. Publisher: Addison-Wesley. 2020. 480 P.
- Shotts William. The Linux Command Line: A Complete Introduction. Publisher: No Starch Press, Incorporated. 2020. 480 P.
- Mouat Andrian. Using Docker: Developing and Deploying Software with Containers. Publisher: O'Relly Media. 2017. 354 P.
- Kocher Parminder Singh. Microservices and Containers. Publisher: Addison-Wesley Professional 2019. 240 P.
- Miell Ian, Sayers Aidin Hobson. Docker in Practice. Publisher: Manning. 2020. 516 P.

Додаткова література

- Ward Brian. How Linux Works: What Every Superuser Shoud Know. Publisher: No Starch Press 2021. 467 P.
- Negus K., Kaen F. Ubuntu and Debian Linux for advanced more than 1000 essential commands. Publisher: Wiley. 2014. 384 P.
- Clinton David. Linux in Action. Publisher: Manning. 2018. 384 P.
- Blum R., Bresnahan Ch. Linux Essentials. Publisher : John Wiley & Sons; 2nd edition. 2017. 368 P.
- 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.
- Robbins A. Bash Pocket Reference: Help for Power Users and Sys Admins. Publisher : O'Reilly Media; 2 edition. 2016. 153 P.
- Sayers Aidan Hobson, Miell Ian. Docker in Practice. Publisher: Manning. 2019. 384 P.
- 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.
- 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.
- Richardson Ch. Microservices Patterns: With examples in Java. Publisher : Manning; 1st edition. 2018. 520 P.
- Horsdal HYPERLINK "https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Christian+Horsdal+Gammelgaard&text=Christian+Horsdal+Gammelgaard&sort=relevancerank&search-alias=books"_____ HYPERLINK "https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Christian+Horsdal+Gammelgaard&text=Christian+Horsdal+Gammelgaard&sort=relevancerank&search-alias=books"Ch HYPERLINK

"https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Christian+Horsdal+Gammelgaard&text=Christian+Horsdal+Gammelgaard&sort=relevancerank&search-alias=books". HYPERLINK "https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=Christian+Horsdal+Gammelgaard&text=Christian+Horsdal+Gammelgaard&sort=relevancerank&search-alias=books"_Microservices in .NET. Publisher : Manning; 2nd edition. 2021. 328 P.

- **Навчальний контент**

- **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.

Лабораторна робота 1. Установка Ubuntu Linux на віртуальній машині VirtualBox.

Розділ 2. Управління Linux-пакетами

Лекція 2. Управління Linux-пакетами

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.

Лабораторна робота 2. Менеджери для роботи з пакетами програм Linux.

Розділ 3. Файлова система Linux

Лекція 3. Особливості файлової системи.

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).

Лабораторна робота 3. Робота в оболонці Bash, змінні середовища оточення і оболонки.

Лекція 4. Робота з файловою системою

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.

Лабораторна робота 4. Робота з файловою системою ОС Linux.

Лекція 5. Фільтрація даних у файлі. Низькорівневе-введення виведення.

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()).

Лабораторна робота 5. Робота з Bash-скриптами

Лекція 6. Сценарії в оболонці bash

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 4. Управління процесами, потоками

Лекція 7. Процеси, потоки, системні виклики

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.

Лабораторна робота 6. Робота з процесами ОС Linux

Лекція 8. Управління процесами в оболонці Bash

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

Лекція 9. Потоковість

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.

Лабораторна робота 7. Потоки, перенаправлення потоків

Розділ 5. Управління пам'яттю

Лекція 10. Управління пам’яттю

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 6. Застосування технології контейнерів

Лекція 11. 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.

Лабораторна робота 8. Установка Docker.

Лекція 12. . 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.

Лабораторна робота 9.1. Створення проекту для web-розробки, який складається з наборів контейнерів з використанням Docker Compose та Dockerfile.

Лекція 13. Життєвий цикл програмного забезпечення при використанні Docker. Мікросервіси і контейнери

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.

Лабораторна робота 9.2. Створення проекту, що складається з контейтерів Django+PostgreSQL, з використанням Docker Compose.

Лекція 14. Мікросервіси і контейнери

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

Розділ 7. Операційні системи сімейства Windows

Лекція 15. Операційні системи сімейства Windows

Історія Windows аж до Windows 11. Програмування в Windows: власний інтерфейс прикладного програмування NT. Інтерфейс прикладного програмування Win32. Реєстр Windows. Структура системи: структура операційної системи. Завантаження Windows.

Реалізація диспетчера об’єктів. Підсистема середовища, DLL і служби користувацького режиму. Регістри.

Лекція 16. Процеси і потоки в Windows

Базові поняття: процеси і потоки в сучасних ОС. Багатопотоковість. Складові елементи процесів і потоків. Стани процесів і потоків. Керуючі блоки процесів і потоків. Образи процесів і потоків. Організація контексту перемикання.

Процеси і потоки в Windows: фундаментальні концепції. Виклики API для управління завданнями, процесами, потоками і волокнами. Реалізація процесів і потоків.

Лекція 17. Управління пам’яттю в Windows

Організація оперативної пам’яті. Віртуальна пам’ять. Сторінкова організація пам’яті. Сегментна організація пам’яті. Сегментно-сторінкова організація пам’яті. Управління пам’яттю: фундаментальні концепції. Системні виклики управління пам’яттю. Реалізація управління пам’яттю. Кешування в Windows.

Лекція 18. Введення-виведення в Windows

Введення-виведення в Windows: фундаментальні концепції. Виклики інтерфейсу прикладного програмування вводу-виводу. Реалізація введення-виведення. Файлова система Windows NT: фундаментальні концепції. Реалізація файлової системи NTFS. Управління електроживленням в Windows. Безпека в Windows 8 (10): фундаментальні концепції. Виклики інтерфейсу прикладного програмування безпеки. Реалізація безпеки. Полегшення умов безпеки.

- **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. Управління Linux-пакетами

Більш детальний огляд інструменту Yum (Yellowdog Updater Modified).

Розділ 3. Файлова система Linux

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

Розділ 4. Управління процесами, потоками, пам’яттю

Ідентифікація процесів. Моніторинг дочірніх процесів.

Розділ 5. Управління пам'яттю

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

Розділ 6. Застосування технології контейнерів

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

Розділ 7. Операційні системи сімейства Windows

Інтерфейс прикладного програмування Win 32.

• Політика та контроль

• 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).

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 в середині грудня),
- по закінченні навчального процесу складають залік.

• Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Система рейтингових (вагових) балів та критерії оцінювання

• Лабораторні роботи. Максимальна кількість балів за усі виконані лабораторні роботи дорівнює 90 балів. Розподіл балів серед лабораторних робіт наступний:

№ з/п	Назва лабораторної роботи	Кількість балів
1	Установка Ubuntu Linux на віртуальній машині VirtualBox	5
2	Менеджери для роботи з пакетами програм Linux	5
3	Робота в оболонці Bash, середовище оточення	10
4	Робота з файловою системою ОС Linux	10
5	Створенні сценаріїв в оболонці Bash	10
6	Робота з процесами ОС Linux	5
7	Потоки, перенаправлення потоків	17
8	Установка Docker	10
9	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	18
Всього:		90

• Модульна контрольна робота. Максимальна кількість балів модульну контрольну роботу дорівнює 10 балів.

Модульна контрольна робота оцінюється таким чином:

1. Коректність та повнота відповіді на 2 теоретичних питання – 6 балів (по 3 бали за кожне теоретичне питання).

2. Надання прикладу на компонент схеми з теоретичного завдання – 4 бали (по 2 бали за кожний приклад).

- Складання диференційованого заліку. Максимальний ваговий бал $r_{зал}$=100

Умови допуску до диференційованого заліку. Необхідною умовою допуску до диференційованого заліку є зарахування усіх лабораторних робіт та отримання не менше 60 балів.

Розрахунок шкали (R) рейтингу:

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

$$R = r_{лаб} + r_{мод} = 90+10 = 100 \text{ балів.}$$

4) Результати *виконання залікової контрольної роботи* оцінюються за 100-бальною шкалою.

Білет залікової контрольної роботи складається з двох теоретичних питань та одного практичного завдання. Ваговий бал кожного теоретичного питання – 30 балів, завдання – 40 балів.

Максимальна кількість балів за складання заліку дорівнює

30 балів *2 + 40 балів = 100 балів.

Теоретична частина оцінюється таким чином:

- правильна чітко викладена, повна відповідь – (не менше 90% потрібної інформації) – 27-30 балів;
- достатньо повна відповідь (не менше 75% потрібної інформації) – 23-26 балів;
- неповна відповідь (не менше 60% потрібної інформації) – 18-22 бали;
- незадовільна відповідь – 0 балів.

Практичне завдання оцінюється таким чином:

- повне, безпомилкове розв'язування завдання – 36-40 балів;
- повне, розв'язування завдання із несуттєвими невідповідностями – 30-35 балів;
- завдання виконане з певними недоліками – 24-29 балів;
- завдання не виконано – 0 балів.

5) Рейтингова оцінка за семестр за бажанням студента визначається одним з таких способів:

1) кількість балів, отриманих за стартовою шкалою, або

2) результат виконання залікової контрольної роботи (тоді не враховуються бали, отримані в семестрі, за умови, що їх кількість не менше 60).

Результат переводиться в оцінку за освітній компонент згідно з таблицею:

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 Ларисою Олексіївною

Ухвалено кафедрою ЦТЕ (протокол № 22 від 25.06.2025 р.)

Погоджено Методичною радою НН ІАТЕ КПІ ім. Ігоря Сікорського (протокол № 9 від 27.06.2025 р.)