



Крос-платформне програмування

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	<i>Перший (бакалаврський)</i>
Галузь знань	<i>12 Інформаційні технології</i>
Спеціальність	<i>124 Системний аналіз</i>
Освітня програма	<i>Системний аналіз і управління</i>
Статус дисципліни	<i>Вибіркова</i>
Форма навчання	<i>очна(денна)</i>
Рік підготовки, семестр	<i>4 курс, осінній семестр</i>
Обсяг дисципліни	<i>4 кредити (лекції 36 год., практикум 18 год., СРС 66 год.)</i>
Семестровий контроль/ контрольні заходи	<i>Залік/МКР</i>
Розклад занять	<i>http://rozklad.kpi.ua/</i>
Мова викладання	<i>Українська</i>
Інформація про керівника курсу / викладачів	<i>Лектор: Титаренко Андрій Миколайович, tytarenko.andrii@iit.kpi.ua Практичні / Семінарські: Титаренко Андрій Миколайович</i>
Розміщення курсу	<i>Moodle, web-page.</i>

Програма навчальної дисципліни

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

Розробка сучасних програмних продуктів часто є складним процесом. Він включає в себе роботу десятків, а інколи сотень людей протягом місяців та років, що обертається великими затратами. Тому цілком природнім є бажання зберегти дорогоцінний час та кошти, якщо це можливо – бажано без суттєвих змін в якості результату. Крос-платформне програмування є одним з інструментів для досягнення цієї цілі, що дозволяє зберегти зусилля під час портування (адаптації) програмного продукту до інших платформ – операційних систем, архітектури центрального процесору або графічних прискорювачів, браузерів, мобільних пристроїв, вбудованих систем, тощо.

Предмет дисципліни: підходи та принципи крос-платформної розробки, платформи та їх відмінності, інструменти адаптації програмного забезпечення до виконання на декількох платформах.

Мета дисципліни: познайомити студентів із типами існуючих платформ, їх особливостей, причинами необхідності адаптації та відповідними інструментами. Не залежно від обраної спеціалізації – науки про дані, системного аналізу, розробки програмного забезпечення, аналітики, тощо – виникає необхідність взаємодіяти із різними платформами та вирішувати відповідні проблеми адаптації, часто навіть виключаючи необхідність змінювати код програми.

Метою кредитного модуля є формування у студентів наступних *компетентностей*:

- Вміння застосовувати інструменти для адаптації програмного забезпечення до різних платформ;
- Вміння аналізувати та проводити діагностику інструментів крос-платформної розробки;
- Вміння використовувати крос-платформні мови програмування – інтерпретовані та на базі Java Virtual Machine (Python, Java).

Програмні результати навчання:

- Досвід застосування інструментів для адаптації програмного забезпечення до різних платформ;
- Вміння використовувати крос-платформні мови програмування Python та Java;
- Вміння розрізняти існуючі платформи та визначати методи адаптації програмного забезпечення до них.

2. Пререквізити та постреквізити дисципліни

Пререквізити: вміння з програмування, базове розуміння операційних систем.

Постреквізити: вміння застосовувати крос-платформні мови програмування на базі Java Virtual Machine та CPython, володіння інструментами контейнеризації (Docker) та віртуальними машинами (QEMU), розуміння принципів роботи компіляторів, бінарних файлів та коду.

Дисципліна належить до вибіркових навчальних дисциплін. Під час вивчення дисципліни використовуються знання з попередніх модулів бакалаврської підготовки: програмування та алгоритмічні мови; операційні системи. Результати вивчення дисципліни застосовуються у подальшій практичній роботі та при підготовці бакалаврської дисертації.

3. Зміст навчальної дисципліни

Розділ 1. Вступ.

Тема 1.1. Платформи та їх види. Проблеми крос-платформної розробки.

Тема 1.2. Огляд і логістика курсу.

Розділ 2. Інтерпретовані мови

Тема 2.1. Інтерпретатори.

Тема 2.2. CPython та його архітектура. Байткод.

Тема 2.3. Мова програмування Python.

Розділ 3. Компільовані мови.

Тема 3.1. Компілятори. Машинний код.

Тема 3.2. Лінування. Бінарні файли.

Тема 3.3. LLVM та GCC.

Тема 3.4. POSIX та крос-платформні API. Принципи крос-платформності на C/C++.

Розділ 4. Мови на базі Java Virtual Machine.

Тема 4.1. Java Virtual Machine. Принципи роботи JVM. Байткод.

Тема 4.2. Мова програмування Java. Об'єкти та стандартна бібліотека.

Тема 4.3. Інтроспекція та рефлексія. Чому JVM є потужним інструментом.

Розділ 5. Контейнеризація.

Тема 5.1. Контейнеризація. LibContainer

Тема 5.2. Docker. Принципи роботи, інтерфейс, Dockerfile.

Тема 5.3. Docker Compose, Docker Hub.

Розділ 6. Віртуалізація.

Тема 6.1. Віртуалізація. Віртуальні машини та їх види.

Тема 6.2. QEMU. Емульовані архітектури.

Тема 6.3. Гіпервізор та WSL.

4. Навчальні матеріали та ресурси

1. Bryant, Randal E., and David Richard O'Hallaron. *Computer systems: a programmer's perspective*. Prentice Hall, 2011.
2. Ramalho, Luciano. *Fluent Python: Clear, concise, and effective programming*. " O'Reilly Media, Inc.", 2015.
3. Shaw, Anthony. *CPython Internals: Your Guide to the Python 3 Interpreter*. Real Python, 2021.
4. <https://llvm.org/>
5. Oaks, Scott. *Java performance: in-depth advice for tuning and programming Java 8, 11, and beyond*. O'Reilly Media, 2020.
6. <https://docs.docker.com/>
7. <https://www.qemu.org/>

Додаткова література

8. Arpaci-Dusseau, Remzi H., and Andrea C. Arpaci-Dusseau. "Operating systems: Three easy pieces." (2018): 15.
9. Chisnall, David. *The definitive guide to the xen hypervisor*. Pearson Education, 2008..

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Методи навчання включають.

1. Лекції: мультимедійні презентації з ілюстрацією прикладів на дошці.
2. Комп'ютерний практикум заняття: групові виконання завдань у навчальній лабораторії, консультації у відведені для цього години.

Перелік тем лекційних занять

№ заняття	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
1.	Лекція 1 Вступ. Платформи та їх види. Крос-платформне програмування та проблеми. Огляд і логістика курсу. Література: 1. Самостійна робота: види платформ.
2.	Лекція 2. Інтерпретовані мови. Інтерпретатори. CPython та його архітектура. Література: 1, 2, 3 Самостійна робота: інші інтерпретовані мови програмування.
3.	Лекція 3. CPython. CPython та його архітектура. Мова програмування Python. Література: 2, 3. Самостійна робота: Python та стандартна бібліотека.
4.	Лекція 4. Python. Мова програмування Python. Пакетний менеджер PyPi. Література: 2, 3. Самостійна робота: віртуальні середовища PyPi.
5.	Лекція 5. Компільовані мови.

№ заняття	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	Процес компіляції. Компільовані мови. Література: 1, 4. Самостійна робота: огляд етапів компіляції.
6.	Лекція 6. Машинний код. Компілятори. Машинний код. Лінкування. Література: 1, 4. Самостійна робота: вивчення процесу лінкування на Linux.
7.	Лекція 7. Інфраструктури компіляторів. LLVM. GCC. Література: 1, 4. Самостійна робота: LLVM та графічні прискорювачі.
8.	Лекція 8. C/C++. POSIX та крос-платформні API. Принципи крос-платформності на C/C++. Література: 1, 4, 8. Самостійна робота: історія крос-платформних API.
9.	Лекція 9. Java Virtual Machine. Java Virtual Machine. Принципи роботи JVM. Байткод. Література: 5. Самостійна робота: перетворення Java програми на байткод.
10.	Лекція 10. Java. Мова програмування Java. Об'єкти та стандартна бібліотека. Література: 5. Самостійна робота: Java робота з мережею.
11.	Лекція 11. Java та JVM. Інтроспекція та рефлексія. Чому JVM є потужним інструментом. Java SE. Література: 5. Самостійна робота: Java SE.
12.	Лекція 12. Контейнеризація. Контейнеризація. LibContainer. Література: 6. Самостійна робота: libcontainer та інтерфейси.
13.	Лекція 13. Docker. Docker. Принципи роботи, інтерфейс. Dockerfile. Література: 6. Самостійна робота: практика роботи з Docker.
14.	Лекція 14. Практичне застосування Docker. Docker Compose. Docker Hub.

№ заняття	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	Література: 6. Самостійна робота: розгортання за допомогою Docker Compose.
15.	Лекція 15. Віртуалізація. Віртуалізація. Віртуальні машини та їх види. Література: 7. Самостійна робота: віртуальні машини VirtualBox.
16.	Лекція 16. Емулятори та QEMU. QEMU. Емульовані архітектури. Література: 7. Самостійна робота: застосування QEMU.
17.	Лекція 17. Гіпервізор та WSL. Що таке гіпервізор. Windows Subsystem for Linux. Література: 1, 9. Самостійна робота: принципи роботи гіпервізора.
18.	Лекція 18. Заключення. Огляд матеріалу. Практичні питання крос-платформної розробки. Огляд подальших тем. Що далі. Література: 1. Самостійна робота: Flutter та інші інструменти крос-платформної розробки.

Комп'ютерний практикум використовуються для отримання та закріплення знань і навичок, висвітлених в лекційній частині курсу та під час самостійного вивчення студентами першоджерел. Основні завдання комп'ютерного практикуму – отримання досвіду роботи із інструментами крос-платформної розробки та крос-платформними мовами програмування

Перелік тем і завдань комп'ютерного практикуму

№ заняття	Назва практичної роботи
1, 2	CPython. Програмування на Python. PyPi.
3, 4	Компіляція та етапи компіляції. LLVM та GCC.
5, 6	JVM. Програмування на Java
7, 8	Docker. Контейнерізація. Docker Compose.
9	QEMU. Віртуальні машини.

У порядку чи тематиці занять можуть бути зміни, викликані наперед невизначеними обставинами. Слідкуйте за оголошеннями викладачів курсу.

6. Самостійна робота студента

У відповідності до робочого навчального плану передбачено 66 годин самостійної роботи студентів, з яких 8 годин - на підготовку до заліку і 58 годин на підготовку до аудиторних занять, опрацювання матеріалів лекцій, самостійний розв'язок додаткових задач та ознайомлення із навчальною літературою відповідно до структури дисципліни. Робота направлена на засвоєння та поглиблення вивченого матеріалу та на підготовку до занять та семестрового контролю. Самостійна робота студентів передбачає:

- закріплення знань, отриманих під час вивчення дисципліни;
- здобуття навичок самостійного вивчення матеріалу.

Перед кожним лекційним заняттям, окрім першого, студент переглядає надані матеріали та рекомендовані джерела, що стосуються теми лекції.

Перед кожним практичним заняттям студент ознайомлюється з темою та завданнями, запланованими на заняття, повторює теоретичний матеріал.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Правила відвідування занять

Студентам рекомендується відвідувати аудиторні заняття. Проте, в разі необхідності студент може пропустити заняття без попереднього повідомлення викладача. Лекційні заняття будуть записуватися для подальшої можливості перегляду онлайн.

Правила поведінки на заняттях

На заняттях передбачається активність студентів, пов'язана з навчальним процесом, включення в інтерактивні форми та методи навчання.

Під час комп'ютерного практикуму, окрім наявного в лабораторії обладнання, студент може користуватися власним ноутбуком.

Правила виконання і захисту практичних робіт

Навчальна лабораторія обладнана персональними комп'ютерами для виконання практичних завдань. До їх використання допускаються лише студенти, які пройшли відповідний інструктаж з техніки безпеки та правил використання обладнання.

Певні завдання комп'ютерного практикуму виконуються бригадами з кількох студентів. Проте за замовчуванням завдання є індивідуальними.

До захисту роботи допускаються студенти, які правильно виконали практичне завдання та мають підтвердження у вигляді завіреного викладачем протоколу. Захист проходить у форматі індивідуального спілкування студента з викладачем по тематиці роботи та виконання практичного завдання.

Правила призначення заохочувальних та штрафних балів

За узгодженням з викладачем студент може виконувати додаткові завдання чи брати участь у модернізації методичного забезпечення дисципліни. Заохоченням до подібної успішної роботи є додаткові рейтингові бали загальним обсягом до 10% від максимального рейтингового балу шкали оцінювання. Кількість та правила нарахування балів узгоджується викладачем у кожному окремому випадку.

Захист практичної роботи проводиться на занятті із комп'ютерного практикуму, на якому видане завдання, або на одному з наступних календарних практичних заняттях, що проходять до зазначеного дедлайну. Деякі завдання мають автоматизовану систему оцінювання і захисту.

Політика дедлайнів та перескладань

Студент, який з будь-яких причин не зміг вчасно виконати та захистити практичну роботу, може це зробити на наступних практичних заняттях за умови доступності обладнання та часу у викладачів. Під час виконання та захисту практичних робіт пріоритет надається студентам, які виконують завдання згідно календарного плану. Виконання та/або захист практичної роботи після відведеного на неї строку призводить до зниження максимального балу, який студент може отримати за цей вид роботи.

Допуск до перескладання заходів семестрового контролю та самі перескладання здійснюються згідно правил Університету у терміни, визначені Університетом.

Політика щодо академічної доброчесності

Політика та принципи академічної доброчесності визначені у розділі 3 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: <https://kpi.ua/code>.

У разі виявлення порушення академічної доброчесності результати роботи студента, які стосуються недоброчесності, анулюються.

Норми етичної поведінки

Норми етичної поведінки студентів і працівників визначені у розділі 2 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: <https://kpi.ua/code>.

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

Рейтинг студента з дисципліни складається з балів, які він отримує за:

- 1) контрольну роботу (20 балів);
- 2) комп'ютерний практикум (80 балів).

Система рейтингових балів

1. Контрольна робота (максимум 20 балів):

- a. творче розкриття завдання (не менше 90% потрібної інформації) – 18...20 балів;
- b. глибоке розкриття завдання (не менше 75% потрібної інформації), незначні неточності або неповні відповіді – 15...18 балів;
- c. достатнє розкриття завдання (не менше 60% потрібної інформації) або часткова наявність помилкової інформації – 11...15 балів;
- d. відповідь не розкриває завдання або містить помилкову інформацію – 0 балів.

2. Комп'ютерний практикум (максимум 20 бали за кожну роботу).

Максимальна кількість балів за виконання циклу комп'ютерного практикуму дорівнює $20 \times 4 = 80$ балів. Комп'ютерний практикум проводиться в якості захисту практичних робіт. Критерії оцінювання:

- a. 18...20 балів – повна відповідь (не менше 90% потрібної інформації);
- b. 15...18 балів – достатньо повна відповідь (не менше 75% потрібної інформації);
- c. 11...15 балів – неповна відповідь (не менше 60% потрібної інформації);
- d. 0 балів – незадовільна відповідь.

Штрафні та заохочувальні бали:

- якщо завдання практичних робіт здається невчасно (пізніше встановленого строку) без поважної причини, то нараховується 2 штрафних бали (знімається 2 бали від максимальної оцінки);
- за активність на лекційних заняттях надається від 1 до 5 заохочувальних балів.

Сума штрафних та заохочувальних балів не може перевищувати 5 балів.

Умови позитивного календарного контролю (атестації)

Для отримання «атестовано» з першого календарного контролю (першої атестації) студент повинен мати не менше ніж 15 балів та виконання всіх завдань практичних робіт (на час атестації). Умовою другого календарного контролю – отримання не менше 30 балів, виконання всіх завдань практичних робіт (на час атестації).

Умови атестації

Умовою атестації студента є виконання усіх поточних контрольних заходів та рейтинг більший за 30 балів. На останньому практичному занятті (за розкладом) проводиться семестрова атестація. Студенти, які отримали за рейтингом позитивну оцінку (набрали протягом семестру не менше ніж 60 балів ($RD \geq 60$)), можуть бути атестовані за цими балами без написання контрольної роботи. Студенти, які отримали менше 60 балів, а також ті, хто хоче підвищити оцінку, виконують контрольну

роботу і захищають її у вигляді співбесіди. Якщо контрольна робота оцінюється нижче, то сумарна рейтингова оцінка залишається незмінною.

Максимальна кількість балів за залікову контрольну роботу – 100.

- вичерпна відповідь – 90...100 балів;
- відповідь з незначними помилками – 80...89 балів;
- неповна відповідь та незначні помилки – 70...79 балів;
- грубі помилки – 60...69 балів.
- незадовільна відповідь – 0 балів.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

<i>Кількість балів</i>	<i>Оцінка</i>
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

9. Додаткова інформація з дисципліни (освітнього компонента)

У рамках опанування дисципліни «Крос-платформне програмування» допускається можливість зарахування сертифікатів проходження дистанційних чи онлайн курсів за відповідною тематикою (за попереднім узгодженням викладачем). Кількість балів визначається після аналізу програми, тривалості та результатів курсу, вказаного у сертифікаті.

Робочу програму навчальної дисципліни (силабус):

Складено Асистентом кафедри математичних методів системного аналізу ІПСА Титаренком Андрієм Миколайовичем.

Ухвалено кафедрою ММСА (протокол № 13 від 05.06.2024)

Погоджено Методичною комісією НН ІПСА (протокол № 10 від 24.06.2024)
