



Кафедра обчислювальної
техніки

Реквізити навчальної дисципліни

Рівень вищої освіти	<i>Перший (бакалаврський)</i>
Галузь знань	<i>12 Інформаційні технології</i>
Спеціальність	<i>121 Інженерія програмного забезпечення</i>
Освітня програма	<i>Інженерія програмного забезпечення комп'ютерних систем</i>
Статус дисципліни	<i>Нормативна</i>
Форма навчання	<i>очна(денна)</i>
Рік підготовки, семестр	<i>1 курс, весняний семестр</i>
Обсяг дисципліни	<i>5,5 кредитів ECTS (165 годин). Лекцій 36 годин, Лабораторні роботи (комп'ютерний практикум) - 54 годин, Самостійна робота студента - 75 годин</i>
Семестровий контроль/ контрольні заходи	<i>Екзамен</i>
Розклад занять	<i>http://rozklad.kpi.ua/</i>
Мова викладання	<i>Українська</i>
Інформація про керівника курсу / викладачів	<i>Керівник курсу: завідувач каф. ОТ, д. т. н., проф. Стіренко С. Г., sergii.stirenko@gmail.com</i> <i>Лектор: старший викладач кафедри обчислювальної техніки Шемсєдинов Т.Г. timur.shemsedinov@gmail.com</i> <i>Лабораторні: асистент кафедри обчислювальної техніки Кухар В.В. kukhar.vitalii@gmail.com</i>
Розміщення курсу	<i>https://github.com/HowProgrammingWorks</i>

Програма навчальної дисципліни

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

Метою навчальної дисципліни є формування у студентів здатностей (компетенцій) програмування, вільного володіння синтаксисом та методологією програмування, розуміння базових структур даних та парадигм. За результатами вивчення дисципліни студент має бути здатним вирішувати професійні завдання та володіти такими компетенціями:

- Здатність до абстрактного мислення, аналізу та синтезу(ЗК01)
- Здатність до пошуку, оброблення та аналізу інформації з різних джерел (ЗК06)
- Здатність аналізувати предметні області, формувати, класифікувати вимоги до програмного забезпечення (ФК01)

- Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування (ФК02)
- Здатність розробляти архітектури, модулі та компоненти програмних систем (ФК03)
- Володіння знаннями про інформаційні моделі даних, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних (ФК07)
- Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення (ФК08)
- Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя (ФК10)
- Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення (ФК11)
- Здатність до алгоритмічного та логічного мислення (ФК14)

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

- Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки (ПРН01)
- Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення (ПРН03)
- Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення (ПРН06)
- Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення (ПРН07)
- Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань (ПРН13)
- Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення (ПРН15)
- Знати та вміти застосовувати інформаційні технології обробки, зберігання та передачі даних (ПРН18)

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

Дисципліни, що передують: Основи програмування. Частина 1

Дисципліни, до яких готує цей курс: Компоненти програмної інженерії. Частина 1 та 2, Основи розробки програмного забезпечення на платформі Node.js, Системне програмування, Паралельні та розподілені обчислення, Гнучкі методи програмування, Моделювання програмного забезпечення.

3. Зміст навчальної дисципліни

- Тема 1. Стан додатків, структури даних та колекції
- Тема 2. Підходи до роботи зі станом: stateful and stateless
- Тема 3. Структури та записи
- Тема 4. Стек, черга, дек
- Тема 5. Дерева та графи
- Тема 6. Проекції та відображення наборів даних
- Тема 7. Оцінка обчислювальної складності
- Тема 8. Структура додатку: файли, модулі, компоненти
- Тема 9. Об'єкт, прототип та клас
- Тема 10. Залежності та бібліотеки
- Тема 11. Регулярні вирази
- Тема 12. Фабрики та пули
- Тема 13. I/O (введення-виведення) та файли
- Тема 14. Мономорфний та поліморфний код, інлайн-кеш, приховані класи
- Тема 15. Вимірювання продуктивності коду та оптимізація
- Тема 16. Асинхронне програмування на callback`ax
- Тема 17. Асинхронне програмування на promise`ax
- Тема 18. Асинхронні функції, async/await, thenable, обробка помилок
- Тема 19. Незмінні структури даних (immutable)
- Тема 20. Автоматне програмування: кінцеві автомати (машини станів)
- Тема 21. Шаблон Singleton (синглтон) JavaScript
- Тема 22. Функціональні об'єкти, функтори та монади
- Тема 23. Асинхронні генератори та асинхронні ітератори
- Тема 24. Перерахований тип (enum)

4. Навчальні матеріали та ресурси

Базова:

1. Шемсєдинов Т.Г., Нечай Д.О., Кухар В.В., Орленко О.А., Голіков О.Г., Білочуб М.М., Духін В., Іванова Л.А., Чорненький А.Ю. та інші. Приклади коду та приклади проектів [Електронний ресурс] розміщено за адресою: [https://github.com/HowProgramming Works/](https://github.com/HowProgrammingWorks/)
2. Refactoring: Improving the Design of Existing Code // Martin Fowler
3. Clean Code: A Handbook of Agile Software Craftsmanship // Robert C. Martin
4. Introduction to Algorithms, 3rd Edition // Thomas H. Cormen
5. Design Patterns // Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides

Додаткова:

1. Шемсєдинов Т.Г., Нечай Д.О., Кухар В.В., Орленко О.А., Голіков О.Г., Білочуб М.М., Духін В., Іванова Л.А., Чорненький А.Ю. та інші. Технологічний стек Metarhia [Електронний ресурс] розміщено за адресою: <https://github.com/metarhia/>

2. Algorithms Unlocked // Thomas H. Cormen
3. The Art of Computer Programming // Donald Knuth
4. Code Complete // Steve McConnell
5. Designing Object Oriented C++ Applications Using The Booch Method // Robert C. Martin
6. Extreme Programming Explained // Kent Beck
7. Analysis Patterns: Reusable Object Models // Martin Fowler

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Основні завдання циклу лабораторних занять (комп'ютерного практикуму) - надбання практичних навичок використання структур даних та колекцій, написання коду, створення багатомодульних бібліотек та програм, оптимізація коду, відлагодження та декомпозиція коду, групова робота та використання систем контролю версій.

№ з/п	Назва лабораторної роботи (комп'ютерного практикуму)	Кількість ауд. годин
1	Реалізація зв'язаних списків	9
2	Побудова модулів та залежностей, структура додатку	9
3	Робота з файлами: дескриптори та файлові потоки	9
4	Обхід дерев та графів	9
5	Оптимізація коду під віртуальну машину V8	9
6	Використання рефлексії та інтроспекції	9
	Разом:	54

6. Самостійна робота студента/аспіранта

У процесі виконання індивідуальних завдань студенти повинні опрацьовувати знання, отримані під час лекцій та самостійної роботи, самостійно вивчати визначені теми, поглиблювати свої знання для подальшого навчання. Самостійна робота студентів полягає в наступному:

- підготовці до лекційних занять по вивченню попереднього лекційного матеріалу;
- розглядання та модифікація прикладів коду та проектів з наданих викладачем git-репозиторіїв;
- виконання лабораторних робіт з вивченням теорії та реалізацією наведеної теми у програмному коді;
- підготовка та участь у обговоренні тем на семінарах;
- пір-рев'ю роду колег студентів;

- підготовка та захищення коду на код-рев'ю викладача.

№ з/п	Назва теми, що виноситься на самостійне опрацювання	Кількість годин СРС
1	Контриб'ютшен у Open-Sourceпроекти	12
2	Налаштування системи тестування та середовища розробки IDE	12
3	Створення та налаштування профілю у Github	12

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Під час занять з навчальної дисципліни студенти повинні дотримуватись певних дисциплінарних правил:

- 1) так використовувати чат-групи, репозиторії та середовище виконання, щоб не створювати проблем з зайвими нотифікаціями іншим учасникам учбового процесу та викладачеві;
- 2) вчасно виконувати роботу та комітити все кожного дня у систему контролю версій, не створювати багато комітів у одному пул-реквесту, не створювати великих комітів, розділяти все на окремі тематичні коміти, чітко та зрозуміло описувати коміти та пул-реквести, додавати теги для лінування issues, PR, та комітів у репозиторіях;
- 3) не допускаються використання піратських копій середовищ розробки, операційних систем чи інших засобів розробки та розгортання як на своїх обчислювальних засобах, так і на хмарних;
- 4) спамити, постити забагато мемів та стікерів у групах та репозиторіях;
- 5) якщо виникає питання, потрібно спочатку шукати у матеріалах курсу, потім у інтернеті, потім запитати у асистентів та інших студентів, а потім вже, тільки коли рішення не знайдено чи воно незрозуміле, турбувати викладача;

Лабораторні роботи здаються тільки через Github у відкритому коді (до коду студенти додають ліцензію MIT), а репозиторій повинен мати історію розробки, щоб викладач міг простежити послідовність написання коду та авторство. Розробка відбувається у feature-гілках git, після чого перевірка коду проходить через PR та включає рев'ю, яке викладач робить у пул-реквесті. Історія рев'ю залишається у Github аккаунті студента.

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

Види контролю з навчальної дисципліни «Основи програмування 2. Методології програмування» включають:

Лабораторні роботи

Заплановано самостійне виконання 6 лабораторних робіт.

Теми лабораторних робіт узгоджені у часі та за змістом з темами лекцій

Поточний контроль:

Передбачено 2 попередні рев'юкода за курс, які повністю охоплюють розглянуту на лекціях тематику навчальної дисципліни. До рев'ю коду викладач може додавати теоретичні питання, які розкривають розуміння теми студентом.

Семестровий контроль

Остаточний рев'ю коду робиться за декілька підходів, але по дедлайну пул-реквест починається, та фіксуються усі зауваження, що виправлені чи не виправлені.

Екзамен

проводиться у вигляді співбесіди зі студентом для об'єктивного визначення рівня знань, умінь та практичних навичок, отриманих за семестр чи парного лайвкодингу за викладачем чи асистентом, парного кодингу з іншим студентом чи повернення до написаного за семестр коду та усунення його недоліків чи обговорення його особливостей.

Рейтинг студента з кредитного модуля складається з балів, які він отримує за види робіт відповідно до таблиці 4.

Таблиця 4

Оцінювання окремих видів навчальної роботи студента

Розділ 1,2		Розділ 3,4	
Вид навчальної роботи	Максимальна кількість балів	Вид навчальної роботи	Максимальна кількість балів
Виконання та захист лабораторної роботи № 1	5	Виконання та захист лабораторної роботи № 4	5
Виконання та захист лабораторної роботи № 2	5	Виконання та захист лабораторної роботи № 5	5
Виконання та захист лабораторної роботи № 3	5	Виконання та захист лабораторної роботи № 6	5
Поточний код-рев'ю №1	5	Поточний код-рев'ю №2	5
		Фінальний код-рев'ю	10
Екзамен			50
Усього за семестр			100

Разом за лабораторні роботи (максимальна кількість балів) – 30

Індивідуальний семестровий рейтинг студента (підсумкова семестрова рейтингова оцінка **RD**) є сумою балів, отриманих студентом протягом семестру за участі у семінарах, рев'ю коду, обговореннях.

Усі студенти, незалежно від того, чи вони виконали всі умови допуску до семестрової атестації з кредитного модуля та мають рейтингову оцінку не менше 60 балів, проходять співбесіду з викладачем.

Необхідною умовою допуску студента до екзамену є його індивідуальний семестровий рейтинг (**RD**) не менший, ніж 30% від максимуму балів, тобто 30 балів, здані 4 лабораторні роботи та наявність однієї позитивної атестації в семестрі. При невиконанні хоча б однієї зі згаданих умов студент до екзамену не допускається.

Сума підсумкової семестрової (**RD**) та екзаменаційної рейтингових оцінок у балах становить підсумкову семестрову рейтингову оцінку, яка перераховується в оцінки за національною шкалою та шкалою ECTS (табл. 5).

Таблиця 5

Відповідність 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

Робочу програму навчальної дисципліни (силабус):

Складено старший викладач кафедри обчислювальної техніки Шемсєдинов Т.Г.

Ухвалено кафедрою обчислювальної техніки (протокол № 10 від 25.05.2022)

Погоджено Методичною комісією факультету (протокол № 10 від 09.06.2022)