ЗАКЛАД ВИЩОЇ ОСВІТИ
«УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА»

Факультет суспільних і прикладних наук

Кафедра інформаційних технологій

ЗАТВЕРДЖУЮ
Проректор з методичної роботи
_____ Ярослав ШТАНЬКО
“ 19 ” _грудня__ 2026 р.

КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

СИЛАБУС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Галузь знань:	12 Інформаційні технології
Спеціальність:	121 Інженерія програмного забезпечення
Освітньо-професійна (освітньо-наукова) програма:	Розробка та тестування програмного забезпечення
Освітній рівень:	(перший) бакалаврський
Статус дисципліни:	обов’язкова
Мова викладання, навчання та оцінювання:	українська

Івано-Франківськ
2025

2

РОЗРОБНИК:

асистент кафедри ІТ Владислав ГОЛУБЦОВ

ЗАТВЕРДЖЕНО:

на засіданні кафедри ІТ
протокол № 5 від 18.12 2025 р.
PhD, завідувач кафедри Олександр ІВАНОВ

УЗГОДЖЕНО:

Гарант ОПП
к. ф-м.н., доцент кафедри Володимир МАКОВИШИН

СХВАЛЕНО:

на засіданні Науково-методичної ради, протокол № __ від 19.12 2025 р.

e-mail	vladyslav.o.holubtsov@ukd.edu.ua
Номер аудиторії чи кафедри	206
Посилання на сайт	https://ukd.edu.ua/dovidnyk/kafedra-informatsiynykh-tekhnolohiy
Сторінка курсу в СДО	https://online.ukd.edu.ua/course/

ВСТУП

Анотація навчальної дисципліни

Навчальна дисципліна "Конструювання програмного забезпечення" є складовою освітньо-професійної програми підготовки фахівців за освітнім ступенем "бакалавр" галузі знань 12 "Інформаційні технології" спеціальності 121 "Інженерія програмного забезпечення", освітньої програми “Розробка та тестування програмного забезпечення”.

Слухачі дисципліни повинні одержати теоретичні знання та практичні навички з використання методів та засобів конструювання програмного забезпечення в систематизованому вигляді для їх застосування на процесах розробки програмних систем на основі мови програмування Python, а також основам роботи в команді.

Мета навчальної дисципліни

Мета дисципліни – формування у студентів бази теоретичних знань та умінь щодо сучасних методів та засобів конструювання програмних систем які розвивають здатність розв'язувати складні спеціалізовані завдання і практичні проблеми інженерії програмного забезпечення.

В результаті вивчення дисципліни студент повинен **знати**:

- основи поводження в команді;
- інструменти для роботи в команді;
- синтаксис мови програмування Python;
- процеси та інструменти для відлагодження програм;
- як працювати з фреймворками;
- інструментальні засоби конструювання програмного забезпечення.

В результаті вивчення дисципліни студент повинен **вміти**:[1]

- вибирати мови програмування для створення програмного забезпечення;
- виконувати розгортання та збірку програм;
- керувати процесом конструювання програм;
- обирати методику конструювання;
- застосовувати набуті знання на практиці.

[1] поняття вміти і знати повинні співвідноситися з програмними результатами навчання

Компетентності та результати навчання, **яких набувають здобувачі освіти внаслідок вивчення навчальної дисципліни (шифри та зміст компетентностей та програмних результатів навчання вказано відповідно до ОПП "Розробка та тестування програмного забезпечення"**.

Шифр та назва компетентності	**Шифр та назва програмних результатів навчання**
К07 Здатність працювати в команді	ПР03 Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення, ПР06 Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення, ПР12 Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення, ПР13 Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань, ПР15 Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення, ПР17 Вміти застосовувати методи компонентної розробки програмного забезпечення
К14 Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування, К15 Здатність розробляти архітектури, модулі та компоненти програмних систем, К19 Володіння знаннями про інформаційні моделі даних та системи, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних, К22 Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження та визнання важливості навчання протягом всього життя, К23 Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення, К24 Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки цілісності, загальної функціональності і надійності програмного забезпечення, К25 Здатність обгрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення, К26 Здатність до алгоритмічного та логічного мислення	

ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Курс	**2**		
Семестр	**4**		
Кількість кредитів ECTS	**5**		
Аудиторні навчальні заняття		**денна форма**	**заочна форма**
	лекції	**30**	**6**
	практичні	**30**	**6**
Самостійна робота (в годинах)		**90**	**138**
Форма підсумкового контролю	**Екзамен**		

Структурно-логічна схема вивчення навчальної дисципліни[2]:

Пререквізити	**Постреквізити**
Інженерія програмного забезпечення	Паралельні та розподілені обчислення
	Менеджмент проектів програмного забезпечення
	Якість програмного забезпечення та тестування

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Перелік тем лекційного матеріалу

Змістовий модуль 1

Ознайомлення з фундаментальними основами програмування та роботи в команді

Тема 1. One day of Software Engineer (4 год.).

Вступ в предмет, мета, завдання. Опис робочого дня програміста. Організація робочого процесу: дошки, статуси, Kanban. GitHub Projects (automation rules). Поняття про meeting, task, release. Побудова проектної документації: README, Notion, Confluence.

Завдання для самостійної роботи. Опис вимог з використанням User Story. Створення Trello, перші кроки.

Тема 2. Використання системи контролю версій GIT. Розгортання додатків в Docker (2 год.)

Розглянемо базовий функціонал: git clone, pull-request, merge, rebase, захищені гілки, GitHub Actions. Зв’язування проекту та GitHub.

[2] тільки для обов’язкових дисциплін

Завдання для самостійної роботи. Створення акаунту в GitHub. Створення першого PR з README.md документацією.

Тема 3. Підготовка до розробки програмного забезпечення (4 год.).

Знайомство з інтегрованим середовищем розробки. Поняття про віртуальне середовище. Dockerfile + docker-compose для Node.js. Знайомство з npm workspaces / pnpm monorepo, environment-variables (.env, secrets).

Завдання для самостійної роботи. Створення другого PR з базовими файлами Docker. Оновлення README.md

Тема 4. Основи програмування на Node.js та JavaScript. Back-End. (4 год.).

Типи даних, оператори, умовні вирази, цикли, ітератори, функції, генератори, контекстні менеджери, робота з ОС та файлами.

Завдання для самостійної роботи. Створення третього PR з першим кодом на Node.js*. Оновлення README.md*

Тема 5. Основи програмування та JavaScript. Front-End. (4 год.).

Базовий синтаксис JavaScript у браузері. DOM-модель: вибір елементів, маніпуляції, події. Робота із запитами fetch(), JSON, async JS у браузері. Основи HTML, CSS (огляд) для JS-інтеракції.

Завдання для самостійної роботи. Створення четвертого PR з першим кодом на JS. Оновлення README.md

Тема 6. Веб-розробка: HTML, CSS, Bootstrap. Створення форм. (4 год.).

Розглянемо основи HTML5: структура документа, теги, семантика, базову стилізацію в CSS. Підключення Bootstrap 5. Надсилання даних форми на бекенд (fetch API).

Завдання для самостійної роботи. Створення п'ятого PR з HTML формою*. Оновлення README.md*

Тема 7. Express та робота з базою даних. Робота з MySQL через ORM. (8 год.)

Встановлення та запуск Express, ініціалізація ORM. Розширення роутингу для роботою з HMTL формою. Створення таблички та робота з даними в MySQL. Додавання MySQL в Docker.

Завдання для самостійної роботи. Створення шостого PR з оновленим Docker файлами та структурою проєкту. Оновлення README.md

Зміст практичних занять

Змістовий модуль 1

Тема 1. Організація роботи програміста. Дошки, задачі, документація (2 год.).

Тема 2. Підготовка середовища розробки. Контроль версій у GIT та перший Pull Request (4 год.).

Тема 3. Ініціалізація Docker, ініціалізація Node.js (2 год.).

Тема 4. Основи програмування на Node.js та JavaScript. Back-End. (4 год.).

Тема 5. Основи програмування та JavaScript. Front-End. (6 год.).

Тема 6. Веб-розробка: HTML, CSS, Bootstrap. Створення форм. (4 год.).

Тема 7. Express та робота з базою даних. Робота з MySQL через ORM. (8 год.).

Зміст самостійної роботи здобувачів

Розподіл годин, виділених на вивчення дисципліни:

Найменування видів робіт	Розподіл годин за формами навчання	
	денна	заочна
Самостійна робота, год, у т.ч.:	90	138
Опрацювання матеріалу, викладеного на лекціях	14	22
Підготовка до практичних занять та контрольних заходів	23	35
Підготовка звітів з практичних робіт	14	22
Підготовка до поточного контролю	7	11
Опрацювання матеріалу, винесеного на самостійне вивчення	32	48

ПОЛІТИКА КУРСУ

Коротко, з покликанням на відповідну нормативну базу УКД, висвітлити питання:[3]

1) щодо системи поточного і підсумкового контролю

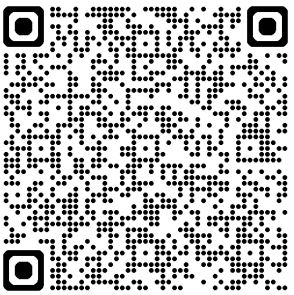
Організація поточного та підсумкового семестрового контролю знань студентів, проведення практик та атестації, переведення показників академічної успішності за 100-бальною шкалою в систему оцінок за національною шкалою здійснюється згідно з "Положенням про систему поточного і підсумкового контролю, оцінювання знань та визначення рейтингу здобувачів освіти". Ознайомитись з документом можна за покликанням.

2) щодо оскарження результатів контрольних заходів

[3] зміст пунктів може редагуватись з огляду на особливості курсу

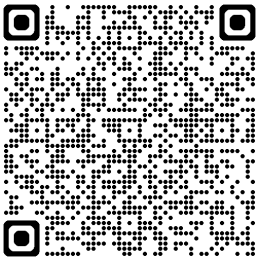
Здобувачі вищої освіти мають право на оскарження оцінки з дисципліни отриманої під час контрольних заходів. Апеляція здійснюється відповідно до «Положення про політику та врегулювання конфліктних ситуацій». Ознайомитись з документом можна за покликанням.

3) щодо відпрацювання пропущених занять



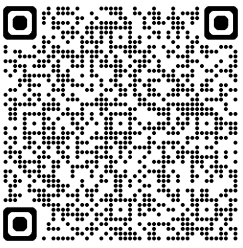
Згідно "Положення про організацію освітнього процесу" здобувач допускається до семестрового контролю з конкретної навчальної дисципліни (семестрового екзамену, диференційованого заліку), якщо він виконав усі види робіт, передбачені на семестр навчальним планом та силабусом/робочою програмою навчальної дисципліни, підтвердив опанування на мінімальному рівні результатів навчання (отримав ≥35 бали), відпрацював визначені індивідуальним навчальним планом всі лекційні, практичні, семінарські та лабораторні заняття, на яких він був відсутній. Ознайомитись з документом можна за покликанням.

4) щодо дотримання академічної доброчесності



"Положення про академічну доброчесність" закріплює моральні принципи, норми та правила етичної поведінки, позитивного, сприятливого, доброчесного освітнього і наукового середовища, професійної діяльності та професійного спілкування спільноти Університету, викладання та провадження наукової (творчої) діяльності з метою забезпечення довіри до результатів навчання. Ознайомитись з документом можна за покликанням.

5) щодо використання штучного інтелекту



"Положення про академічну доброчесність" визначає політику щодо використання технічних засобів на основі штучного інтелекту в освітньому процесі. Ознайомитись з документом можна за покликанням.[4] *"Положення про систему запобігання та виявлення академічного плагіату, самоплагіату, фабрикації та фальсифікації академічних творів" містить рекомендації щодо використання в академічних текстах генераторів на основі штучного інтелекту. Ознайомитись з документом можна за покликанням.*

[4] визначається політика використання ШІ в навчальній дисципліні - дозволене/заборонене, правила використання

6) щодо використання технічних засобів в аудиторії та правила комунікації

Використання мобільних телефонів, планшетів та інших гаджетів під час лекційних та практичних занять дозволяється виключно у навчальних цілях (для уточнення певних даних, перевірки правопису, отримання довідкової інформації тощо). На гаджетах повинен бути активований режим «без звуку» до початку заняття. Під час занять заборонено надсилання текстових повідомлень, прослуховування музики, перевірка електронної пошти, соціальних мереж тощо. Під час виконання заходів контролю використання гаджетів заборонено (за винятком, коли це передбачено умовами його проведення). У разі порушення цієї заборони результат анульовується без права перескладання.

Комунікація відбувається через електронну пошту і сторінку дисципліни в Moodle.

7) щодо зарахування результатів навчання, здобутих шляхом формальної/інформальної освіти

Процедури визнання результатів навчання, здобутих шляхом формальної/інформальної освіти визначаються «Положенням про порядок визнання результатів навчання, здобутих шляхом неформальної та / або інформальної освіти». Ознайомитись з документом можна за покликанням.[5]

МЕТОДИ НАВЧАННЯ

При вивченні дисципліни застосовується комплекс методів для організації навчання студентів з метою розвитку їх логічного та абстрактного мислення, творчих здібностей, підвищення мотивації до навчання та формування особистості майбутнього фахівця.

Програмний результат навчання[6]	**Метод навчання**	**Метод оцінювання**
ПР03 Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення, ПР06	лекція, вправи, практичні роботи, кейс-метод, узагальнення лекція, вправи, лабораторні роботи,	іспит, письмовий контроль, тестовий контроль іспит, письмовий контроль, тестовий контроль

[5] визначається перелік електронних та інших ресурсів та умови перезарахування
[6] для вибіркових навчальних дисциплін вказується результат навчання

Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення, ПР12 Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення, ПР13 Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань, ПР15 Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення, ПР17 Вміти застосовувати методи компонентної розробки програмного забезпечення	кейс-метод, рольові і ділові ігри лекція, практичні роботи, порівняння, узагальнення, рольові і ділові ігри лекція, практичні роботи, порівняння, узагальнення, кейс-метод лекція, практичні роботи, порівняння, узагальнення, кейс-метод лекція, практичні роботи, порівняння, узагальнення, рольові і ділові ігри	іспит, письмовий контроль, тестовий контроль іспит, письмовий контроль, тестовий контроль іспит, письмовий контроль, тестовий контроль іспит, письмовий контроль, тестовий контроль

ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Контрольні заходи[7]

(в разі потреби - розділити за семестрами)

Вид	***Зміст[8]***	***% від загальної оцінки***	***Бал***	
			min	***max***
Поточні контрольні заходи	всього	**60**	**35**	**60**
Підсумкові контрольні заходи	проект	**40**	**25**	**40**

[7] зміст редагується залежно від наповнення дисципліни

[8] у випадку наявності видів роботи, які оцінюються окремо (проєкт, завдання тощо) прописується в окремому рядку; за відсутності - одним рядком визначається вся сукупність аудиторної роботи (опитування, поточні контрольні тощо) та визначається стандартне значення балів (35/60)

Всього:		**100**	**60**	**100**

Процедура проведення контрольних заходів, а саме поточного контролю знань протягом семестру та підсумкового семестрового контролю, регулюється «Положенням про систему поточного та підсумкового контролю оцінювання знань та визначення рейтингу студентів».

Фіксація **поточного** контролю здійснюється в “Електронному журналі обліку успішності академічної групи” на підставі чотирибальної шкали - “2”; “3”; “4”; “5”. У разі відсутності студента на занятті виставляється “н”. За результатами поточного контролю у Журналі, автоматично визначається підсумкова оцінка, здійснюється підрахунок пропущених занять.

Усі пропущені заняття, а також негативні оцінки студенти зобов'язані відпрацювати впродовж трьох наступних тижнів. У випадку недотримання цієї норми, замість “н” в журналі буде виставлено “0” (нуль балів), без права перездачі. Відпрацьоване лекційне заняття в електронному журналі позначається літерою «в».[9]

Критерії оцінювання[10] (за необхідності, поточного та/або підсумкового контролю)

До підсумкового контролю допускаються студенти які за результатами поточного контролю отримали не менше 35 балів. Усі студенти, що отримали 34 балів і менше, не допускаються до складання підсумкового контролю і на підставі укладання додаткового договору, здійснюють повторне вивчення дисципліни впродовж наступного навчального семестру. За результатами підсумкового контролю (диференційований залік/екзамен) студент може отримати 40 балів. Студенти, які під час підсумкового контролю отримали 24 бали і менше, вважаються такими, що не здали екзамен/диференційований залік і повинні йти на перездачу. Підсумковим контролем вивчення дисципліни є проєкт. Проєкт виконується згідно тематик та вимог.

Загальна семестрова оцінка з дисципліни, яка виставляється в екзаменаційних відомостях оцінюється в балах (згідно з **Шкалою оцінювання знань за ЄКТС**) і є сумою балів отриманих під час поточного та підсумкового контролю.

Підсумковий контроль з дисципліни Конструювання програмного забезпечення проводиться у вигляді тестового екзамену.

Шкала оцінювання знань за ЄКТС:

Оцінка за національною шкалою	**Рівень досягнень, %**	**Шкала ECTS**

[9] можна вказати теми чи завдання, які є обов’язковими до виконання, а також особисті підходи до оцінювання рівня знань здобувачів під час аудиторної роботи

[10] критерії вказуються згідно з особливостями дисципліни.

Національна диференційована шкала		
Відмінно	90 – 100	A
Добре	83 – 89	B
	75 – 82	C
Задовільно	67 – 74	D
	60 – 66	E
Незадовільно	35 – 59	FX
	0 – 34	F
Національна недиференційована шкала		
Зараховано	60 – 100	-
Не зараховано	0 – 59	-

Студенти, які не з'явилися на заліки/екзамени без поважних причин, вважаються такими, що одержали незадовільну оцінку.

РЕКОМЕНДОВАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ[11]

Основна література

1. Фаулер М. Рефакторинг: вдосконалення існуючого коду. Львів: Видавництво Старого Лева, 2022. 544 с.
2. Мартін Р. Чистий код. Створення, аналіз та рефакторинг. Київ: Fabula, 2019. 416 с.
3. Flanagan D. JavaScript: The Definitive Guide. Sebastopol: O'Reilly Media, 2020. 1096 p.
4. Tilkov S., Vinoski S. Node.js: Patterns and Best Practices. Birmingham: Packt Publishing, 2020. 352 p.
5. Sequelize ORM Documentation: офіційний довідник. 2024. URL: https://sequelize.org (дата звернення: 20.01.2026).
6. Docker Documentation: довідник із Dockerfile та Docker-Compose. 2024. URL: https://docs.docker.com (дата звернення: 20.01.2026).
7. MySQL 8 Developer Guide. Oracle Corp, офіційна документація. URL: https://dev.mysql.com/doc (дата звернення: 20.01.2026).

Електронні інформаційні ресурси

[11] обов'язково: враховувати вимоги ДСТУ 8302:2015 (відповідно до Наказу № 65, від 4.03. 2016), рекомендації Національного агентства з забезпечення якості вищої освіти, використовувати літературу за останні 5-7 років, наводити власні публікації за змістом навчальної дисципліни.

1. Документація Node.js (офіційний сайт) URL: https://nodejs.org/en/docs (Дата звернення: 25.12.2025) (Дата звернення: 24.12.2025).
2. Документація Express.js URL: https://expressjs.com/ (Дата звернення: 24.12.2025)
3. MySQL Developer Documentation URL: https://dev.mysql.com/doc/ (Дата звернення: 25.12.2025)
4. Sequelize ORM (офіційна документація) URL: https://sequelize.org/ (Дата звернення: 25.12.2025)
5. Docker Documentation URL: https://docs.docker.com/ (Дата звернення: 26.12.2025)
6. Bootstrap 5 Documentation URL: https://getbootstrap.com/docs/5.3/getting-started/introduction/ (Дата звернення: 26.12.2025)
7. Git Book — Pro Git (безкоштовна онлайн-книга) URL: https://git-scm.com/book/en/v2 (Дата звернення: 27.12.2025)
8. JavaScript MDN Web Docs (Mozilla Developer Network) URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript (Дата звернення: 27.12.2025)
9. W3Schools — навчальні матеріали з фронтенду та Node.js URL: https://www.w3schools.com/nodejs/ (Дата звернення: 28.12.2025)