

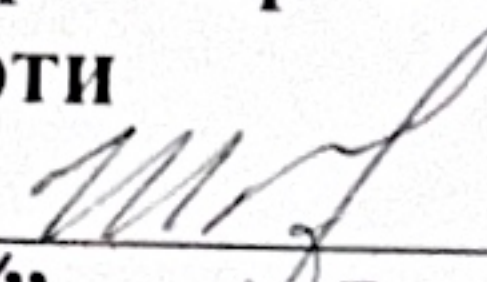
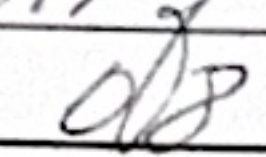
ЗАКЛАД ВИЩОЇ ОСВІТИ
«УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА»

Факультет суспільних і прикладних наук

Кафедра інформаційних технологій

ЗАТВЕРДЖУЮ

Проректор з навчально-методичної
роботи

 Ярослав ШТАНЬКО
“31”  2026 р.

Основи програмування

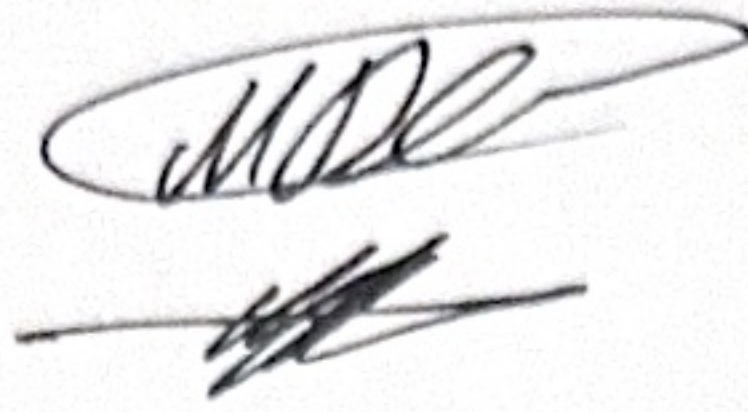
СИЛАБУС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Галузь знань:	F Інформаційні технології
Спеціальність:	F2 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
Семестр викладання	I

Івано-Франківськ
2026

РОЗРОБНИКИ:

доцент кафедри ІТ
доцент кафедри ІТ



Микола ДЕМЧИНА
Дмитро ЛУЦАК

ЗАТВЕРДЖЕНО:

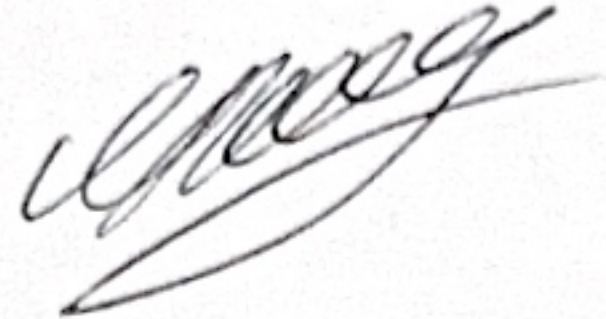
на засіданні кафедри інформаційних технологій, протокол № 1 від 27.08.2026 р.

Завідувач кафедри
УЗГОДЖЕНО:



Олександр ІВАНОВ

Гарант ОПП/ОНП



Володимир МАКОВИШИН

СХВАЛЕНО:

на засіданні Науково-методичної ради, протокол № 1 від 31.08.2026

e-mail	mykola.demchyna@ukd.edu.ua dmytro.l.lutsak@ukd.edu.ua
Номер аудиторії чи кафедри	206
Посилання на сайт	https://ukd.edu.ua
Сторінка курсу в СДО	https://online.ukd.edu.ua/

ВСТУП

Анотація навчальної дисципліни

Навчальна дисципліна «Основи програмування» формує базові алгоритмічні та практичні навички розроблення програм. Основною мовою є JavaScript, а середовищами виконання – браузер і Node.js. Курс охоплює алгоритми, типи даних, керувальні конструкції, функції, масиви, об'єкти, класи, асинхронність, налагодження, тестування та контроль версій.

У другому семестрі здобувачі застосовують засоби JavaScript у вебсередовищі: створюють семантичну HTML-структуру, оформлюють її засобами CSS, працюють із DOM, подіями, формами та локальним сховищем. Навчання побудовано навколо послідовного виконання практичних завдань і завершується інтегрованим вебпроектом.

Мета навчальної дисципліни – сформувати здатність алгоритмічно мислити, проєктувати, реалізовувати, тестувати й документувати нескладні програмні рішення мовою JavaScript та застосовувати їх для створення інтерактивних вебсторінок.

Основні завдання навчальної дисципліни

- опанувати базові алгоритмічні конструкції, типи даних і структури даних;
- навчитися писати зрозумілий модульний код мовою JavaScript і працювати з помилками;
- застосовувати тестування, налагодження, документування та систему контролю версій Git;
- створювати доступні інтерактивні вебсторінки з використанням HTML, CSS, DOM і подій.

У результаті вивчення дисципліни здобувач повинен знати

- властивості алгоритмів, способи їх подання та базові структури керування;
- синтаксис, типи даних, оператори, функції, масиви, об'єкти, класи й асинхронну модель JavaScript;
- принципи модульності, налагодження, тестування, оброблення помилок і контролю версій;
- структуру HTML-документа, каскадність CSS, основи адаптивного компонування та модель DOM.

У результаті вивчення дисципліни здобувач повинен уміти

- аналізувати умову задачі, будувати алгоритм і добирати відповідні структури даних;
- реалізовувати та налагоджувати програми мовою JavaScript у браузері й Node.js;
- писати перевірки, опрацьовувати граничні випадки та пояснювати коректність рішення;
- вести репозиторій Git, працювати з гілками, злиттями та запитами на злиття;

- створювати й тестувати інтерактивні вебсторінки з HTML, CSS і JavaScript.

Компетентності та результати навчання

Шифри й формулювання наведено відповідно до ОПП «Розробка та тестування програмного забезпечення» у редакції 2026 року. Стовпці містять незалежні переліки компетентностей і програмних результатів навчання для ОК 10.

Шифр та назва компетентності	Шифр та назва програмних результатів навчання
ЗК1. Здатність до абстрактного мислення, аналізу та синтезу.	ПРН7. Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.
ЗК2. Здатність застосовувати знання у практичних ситуаціях.	ПРН13. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.
СК3. Здатність розробляти архітектури, модулі та компоненти програмних систем.	ПРН15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.
СК6. Здатність аналізувати, вибирати і застосовувати методи і засоби для забезпечення інформаційної безпеки (в тому числі кібербезпеки).	
СК7. Володіння знаннями про інформаційні моделі даних, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних.	
СК8. Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення.	

Шифр та назва компетентності	Шифр та назва програмних результатів навчання
СК10. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя.	
СК11. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.	
СК13. Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.	
СК14. Здатність до алгоритмічного та логічного мислення.	

ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Курс	1		
Семестр	1, 2		
Кількість кредитів ECTS	9		
Аудиторні навчальні заняття		денна форма	заочна форма
	лекції	0	0
	практичні	126	24
Самостійна робота		144	246
Форма підсумкового контролю	Залік (1 семестр), екзамен (2 семестр)		

Загальний обсяг дисципліни – 270 годин. Для денної форми передбачено 126 годин практичних занять і 144 години самостійної роботи; для заочної форми – 24 та 246 годин відповідно. У першому семестрі денна форма охоплює 42 практичні й 48 самостійних годин, у другому – 84 практичні й 96 самостійних годин.

Структурно-логічна схема вивчення навчальної дисципліни

Пререквізити	Постреквізити
Шкільний курс інформатики та математики; Базові навички роботи з операційною системою і браузером.	Алгоритми та структури даних; Об'єктно-орієнтоване програмування; Розробка веб-додатків, веб-дизайн та тестування.

Для занять потрібні персональний комп'ютер, сучасний браузер, редактор коду, Node.js, Git і доступ до СДО. Навчальні репозиторії не повинні містити паролів, токенів доступу, а також персональних або конфіденційних даних.

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Лекційні заняття

Лекційні заняття навчальним планом не передбачені. Необхідні теоретичні відомості подаються під час практичних занять, консультацій і керованої самостійної роботи.

Зміст практичних занять

1 семестр

Тема 1. Вступ до програмування та робоче середовище (2 год.)

Мови програмування, компіляція й інтерпретація; JavaScript у браузері та Node.js; запуск першої програми, консоль і базове налагодження.

Питання для самостійного опрацювання: Порівняти парадигми програмування; установити й перевірити інструменти розробки. Література: [1, 5, 6].

Тема 2. Алгоритми та блок-схеми (4 год.)

Властивості алгоритмів, способи подання, базові структури; побудова блок-схем і трасування виконання.

Питання для самостійного опрацювання: Описати алгоритм словами, псевдокодом і блок-схемою; перевірити його на граничних даних. Література: [1, 2, 7].

Тема 3. Введення і виведення даних (2 год.)

Консоль, браузерні діалоги та модуль readline у Node.js; перетворення введених значень; повідомлення про помилки.

Питання для самостійного опрацювання: Порівняти введення у браузері й Node.js; дослідити об'єкти помилок. Література: [5, 6, 8].

Тема 4. Змінні, типи даних і арифметичні оператори (4 год.)

Оголошення let і const, іменування, примітивні та об'єктні значення, typeof, перетворення типів, арифметика й пріоритет операторів.

Питання для самостійного опрацювання: Дослідити точність чисел з рухомою комою, Math та відмінності строгого й нестрогого порівняння. Література: [1, 5, 7].

Тема 5. Рядки та шаблонні літерали (4 год.)

Створення рядків, індексація, властивості й методи, пошук, зрізи, форматування та шаблонні літерали.

Питання для самостійного опрацювання: Розробити набір функцій для перевірки й нормалізації текстових даних. Література: [1, 6, 7].

Тема 6. Лінійні програми (4 год.)

Декомпозиція обчислення на послідовність кроків, проміжні змінні, читабельні імена, прості формули та перевірка результату.

Питання для самостійного опрацювання: Проаналізувати область видимості й підняття оголошень; скласти тестові приклади. Література: [1, 5, 7].

Тема 7. Умовні конструкції (4 год.)

if, else if, else, switch і тернарний оператор; вкладені розгалуження; валідація введення.

Питання для самостійного опрацювання: Побудувати таблицю тестів для розгалуженого алгоритму. Література: [1, 5, 6].

Тема 8. Логічні оператори та складні умови (4 год.)

Оператори &&, ||, !, коротке замикання, truthy/falsy, оператори ?? і ?.; спрощення логічних виразів.

Питання для самостійного опрацювання: Побудувати таблиці істинності та перевірити еквівалентність умов. Література: [1, 5, 7].

Тема 9. Одновимірні масиви (6 год.)

Створення, індексація, перебір і зміна масивів; push, pop, shift, unshift, slice, splice; пошук і базові ітеративні методи.

Питання для самостійного опрацювання: Порівняти поверхневе й глибоке копіювання; розв'язати задачі на агрегацію даних. Література: [1, 2, 6].

Тема 10. Цикли (4 год.)

for, while, do...while, for...of, break і continue; вкладені цикли; умови завершення та запобігання нескінченним циклам.

Питання для самостійного опрацювання: Оцінити кількість ітерацій і перевірити цикли на порожніх та граничних наборах. Література: [1, 5, 7].

Тема 11. Двовимірні масиви (4 год.)

Подання таблиць і матриць, вкладені цикли, пошук, підсумки за рядками й стовпцями, транспонування.

Питання для самостійного опрацювання: Реалізувати й протестувати операції над прямокутними масивами. Література: [1, 2, 6].

2 семестр**Тема 12. Функції та область видимості (6 год.)**

Оголошення й виклик функцій, параметри, аргументи, значення за замовчуванням, return, локальна й зовнішня область видимості, чисті функції.

Питання для самостійного опрацювання: Декомпонувати програму на функції та сформулювати їхні контракти. Література: [1, 5, 6].

Тема 13. Стрілкові функції (4 год.)

Синтаксис стрілкових функцій, неявне повернення, лексичне `this`, застосування з методами масивів.

Питання для самостійного опрацювання: Порівняти звичайні й стрілкові функції на прикладах контексту виклику. Література: [1, 5, 7].

Тема 14. Функції зворотного виклику (4 год.)

Функції як значення, передавання поведінки, колбеки у методах масивів і власних функціях; читабельність вкладених викликів.

Питання для самостійного опрацювання: Створити функцію вищого порядку та перевірити її кількома колбеками. Література: [1, 2, 6].

Тема 15. Рекурсія (4 год.)

Базовий випадок, рекурсивний крок, стек викликів, типові рекурсивні структури й ризик нескінченної рекурсії.

Питання для самостійного опрацювання: Порівняти рекурсивне та ітеративне розв'язання однієї задачі. Література: [1, 2, 6].

Тема 16. Система контролю версій Git (12 год.)

Репозиторій, staging area, коміти, історія, `.gitignore`, гілки, злиття, конфлікти, віддалені репозиторії, clone, fork і pull request; командна робота.

Питання для самостійного опрацювання: Створити навчальний репозиторій, організувати гілки, виконати злиття та задокументувати розв'язання конфлікту. Література: [3, 9].

Тема 17. Об'єкти та ітератори (4 год.)

Об'єктні літерали, властивості, методи, деструктуризація, вкладені об'єкти, `Object.keys/values/entries`, протокол ітерації.

Питання для самостійного опрацювання: Розробити модель предметних даних та обґрунтувати спосіб обходу. Література: [1, 5, 7].

Тема 18. Класи та наслідування (8 год.)

Класи, конструктор, екземпляри, методи, приватні поля, `extends`, `super`, перевизначення та зв'язок класів із прототипами.

Питання для самостійного опрацювання: Побудувати невелику ієрархію класів і перевірити інваріанти об'єктів. Література: [1, 5, 6].

Тема 19. Регулярні вирази (4 год.)

Літерали й конструктор `RegExp`, `test` і `match`, прапорці, класи символів, квантифікатори, групи та альтернативи.

Питання для самостійного опрацювання: Скласти й протестувати регулярні вирази для валідації кількох форматів даних. Література: [1, 5, 7].

Тема 20. Асинхронне програмування (8 год.)

Подієвий цикл, колбеки, `Promise`, `then/catch/finally`, `async/await`, оброблення помилок, `Promise.all` і робота з асинхронними операціями.

Питання для самостійного опрацювання: Побудувати часову діаграму виконання та порівняти послідовне й паралельне очікування. Література: [1, 5, 8].

Тема 21. Основи HTML (8 год.)

Структура документа, семантичні елементи, заголовки, текст, списки, посилання, зображення, таблиці, форми й базова доступність.

Питання для самостійного опрацювання: Створити валідну семантичну сторінку з навігацією, таблицею та формою. Література: [4, 6, 10].

Тема 22. Основи CSS (8 год.)

Каскад, специфічність, селектори, блокова модель, типографіка, кольори, Flexbox, Grid і засади адаптивного компонування.

Питання для самостійного опрацювання: Стилізувати сторінку й перевірити її на різних ширинах екрана. Література: [4, 6, 11].

Тема 23. Інтегрована фронтенд-розробка (14 год.)

Підключення JavaScript, DOM, пошук і зміна елементів, події, класи й атрибути, створення елементів, валідація форм, localStorage/sessionStorage, модульність, тестування та підсумковий проєкт.

Питання для самостійного опрацювання: Завершити інтерактивний вебзастосунок, перевірити основні сценарії, оформити репозиторій і підготувати захист. Література: [1, 3, 4, 6, 10, 11].

Разом практичних занять для денної форми – 126 годин: 42 години у першому семестрі та 84 години у другому.

За кожною темою здобувач подає вихідний код, контрольні приклади або тести, короткий висновок і посилання на репозиторій, якщо це передбачено завданням. Програма має запускатися за наданою інструкцією, а здобувач – пояснювати її алгоритм і ключові рішення.

Зміст самостійної роботи здобувачів

Розподіл годин, виділених на вивчення дисципліни

Найменування видів робіт	Розподіл годин	
	денна форма	заочна форма
Самостійна робота, год, у т.ч.:	144	246
Опрацювання теоретичного матеріалу, поданого в СДО та на практичних заняттях	23	39
Підготовка до практичних занять та контрольних заходів	36	62
Підготовка й оформлення звітів, коду та навчального репозиторію	23	39
Підготовка до поточного та підсумкового контролю	12	20
Опрацювання матеріалу, винесеного на самостійне вивчення, та виконання підсумкового проєкту	50	86

Самостійна робота включає роботу з документацією, додаткові алгоритмічні задачі, тести й звіти, ведення Git-репозиторію та інтегрований вебпроєкт. Конкретні завдання і строки за семестрами публікуються в СДО.

ПОЛІТИКА КУРСУ

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 здобувачів освіти». Ознайомитись з документом можна за [ПОКЛИКАННЯМ](#).



Здобувачі мають право оскаржувати результати поточного оцінювання, а також інформувати про факти конфліктних ситуацій в академічній групі чи з викладачем у відповідності до «Положення про політику та врегулювання конфліктних ситуацій». Ознайомитись з документом можна за [ПОКЛИКАННЯМ](#).



До семестрового контролю допускається здобувач, який виконав усі види робіт, передбачені навчальним планом і силабусом, відпрацював пропущені заняття та набрав не менше 35 балів поточного контролю. Ознайомитись з документом можна за [ПОКЛИКАННЯМ](#).



Вимоги щодо дотримання академічної доброчесності, види її порушень і порядок притягнення до академічної відповідальності визначено Положенням про систему забезпечення академічної доброчесності в Університеті Короля Данила. Ознайомитись з документом можна за [ПОКЛИКАННЯМ](#).



Використання систем штучного інтелекту під час виконання навчальних завдань допускається відповідно до “Політики використання штучного інтелекту в Університеті Короля Данила”. Здобувач зобов’язаний критично перевіряти створений ШІ зміст, нести відповідальність за подану роботу та зазначати факт і спосіб використання таких систем. Ознайомитись з документом можна за [ПОКЛИКАННЯМ](#).



Генеративний ШІ можна використовувати для пояснення понять, пошуку альтернативних підходів, налагодження та створення додаткових тестів, якщо це не заборонено умовою завдання. У звіті потрібно зазначити сервіс, спосіб використання та внесені здобувачем зміни. Заборонено подавати згенерований код без перевірки й розуміння, використовувати ШІ під час контрольних заходів без дозволу або передавати сервісам паролі, токени, персональні чи конфіденційні дані. Здобувач відповідає за коректність і авторство поданої роботи.

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 надсилення 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 анулюється без права перескладання. Ознайомитись з відповідним документом можна за [посиланням](#).



Комунікація відбувається через mykola.demchyna@ukd.edu.ua та сторінку дисципліни в СДО Moodle. Навчальні матеріали, завдання, строки виконання й оголошення розміщуються у СДО.

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



Для визнання результатів неформальної та/або інформальної освіти здобувач подає докази опанування відповідних тем і виконує практичне завдання або контрольну перевірку. Зміст, обсяг і результати навчання зіставляються із силабусом за процедурою університету.

МЕТОДИ НАВЧАННЯ

Навчання здійснюється через короткі пояснення й демонстрації, кероване програмування, поступове ускладнення задач, аналіз помилок, взаємне рецензування коду та проєктну роботу. На практичних заняттях здобувачі планують алгоритм, реалізують його, перевіряють за допомогою контрольних даних, налагоджують і пояснюють результат.

Результат навчання	Метод навчання	Метод оцінювання
ПРН7. Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.	Демонстрація мовних конструкцій, кероване програмування, налагодження у браузері й Node.js, робота з офіційною документацією.	Практичні задачі, пояснення коду, тести з основ мови та інструментів.
ПРН13. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.	Алгоритмічні вправи, трасування, декомпозиція, робота з масивами й об'єктами, тестування граничних випадків.	Перевірка алгоритмів і коду, захист практичних робіт, контрольні задачі.
ПРН15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.	Порівняння інструментів і підходів, Git-практикум, код-рев'ю, інтегрований вебпроєкт і рефлексія щодо вибору технологій.	Репозиторій і історія змін, проєкт, усний захист, комп'ютерне тестування.

ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Оцінювання здійснюється відповідно до «Положення про систему поточного і підсумкового контролю, оцінювання знань та визначення рейтингу здобувачів освіти» у редакції від 01.07.2026. [Положення УКД](#)

Вид	Зміст	% від загальної оцінки	Бал	
			min	max
Поточні контрольні заходи	всього	60	35	60
Підсумкові контрольні заходи	залік / екзамен	40	25	40
Всього:	-	100	60	100

Поточне оцінювання фіксується в електронному журналі за шкалою «2», «3», «4», «5»; відсутність позначається «н». Підсумковий бал за поточну роботу обчислюється як середнє арифметичне поточних оцінок, включно з «0» за невідпрацьовані пропуски, помножене на 12. Максимум – 60 балів.

Поточні оцінки виставляються за розв’язання й захист практичних задач, коректність алгоритму, якість коду, тестів і звітів, роботу з Git та виконання вебпроєкту. У кожному семестрі здобувач повинен мати оцінки щонайменше з 50 % практичних занять, виконати визначені обов’язкові роботи й підтвердити досягнення ПРН7, ПРН13 і ПРН15. Для допуску до семестрового контролю потрібно набрати не менше 35 балів.

Поточний і підсумковий контроль оцінюються окремо в кожному семестрі. Залік у першому семестрі та екзамен у другому оцінюються максимум у 40 балів. Мінімальний позитивний результат підсумкового контролю – 25 балів; 24 бали і менше означають, що контроль не складено. Підсумок за семестр є сумою балів поточного й підсумкового контролю; позитивна семестрова оцінка становить не менше 60 балів.

Критерії оцінювання практичних робіт і проєкту

Оцінка	Критерії
«2» – незадовільно	Рішення не запускається або не відповідає умові; алгоритм відсутній чи принципово помилковий; немає перевірок; здобувач не може пояснити код.
«3» – задовільно	Рішення працює для типових даних за зразком; структура коду й перевірки часткові; є помилки стилю або граничних випадків; пояснення потребує допомоги.
«4» – добре	Алгоритм коректний у стандартних і більшості граничних випадків; код читабельний та декомпонований; надано тести, звіт і змістовну історію змін.

Оцінка	Критерії
«5» – відмінно	Рішення самостійне, коректне й обґрунтоване; опрацьовано граничні та помилкові дані; код модульний, перевірений і задокументований; здобувач упевнено захищає вибір підходу.

Порядок відпрацювання та підсумкового контролю

Пропущені заняття й негативні оцінки необхідно відпрацювати впродовж трьох наступних тижнів. За тривалої відсутності з поважних причин, підтверджених деканатом, строк відраховується з моменту повернення. За недотримання строку замість «н» у журналі виставляється «0» без права перездачі цього пропуску; відпрацьована лекція позначається «в».

Підсумковий контроль проводиться в СДО та/або комп'ютерній аудиторії. Залік у першому семестрі охоплює базові алгоритми, типи даних, розгалуження, цикли й масиви. Екзамен у другому семестрі охоплює функції, об'єкти, класи, асинхронність, Git і веброзробку та може містити комп'ютерне тестування й практичне завдання з написання або аналізу коду. Тривалість, дозволені засоби й критерії оприлюднюються в СДО до контролю.

Неявка на залік або екзамен без поважної причини прирівнюється до незадовільного результату. Перескладання, ліквідація академічної заборгованості й оскарження результату відбуваються відповідно до положень університету.

Шкала оцінювання знань за ЄКТС

Оцінка за національною шкалою	Рівень досягнень, бали	Шкала ECTS
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	-

РЕКОМЕНДОВАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ

Основна література

1. Flanagan D. JavaScript: The Definitive Guide. 7th ed. O'Reilly Media, 2020.
2. Haverbeke M. Eloquent JavaScript. 4th ed. No Starch Press, 2024.
3. Simpson K. You Don't Know JS Yet: Get Started. 2nd ed. O'Reilly Media, 2020.
4. Бублик В. В. Об'єктно-орієнтоване програмування. Київ: АртЕк, 2022.

Електронні інформаційні ресурси

5. Ecma International. ECMAScript® 2026 Language Specification. URL: <https://tc39.es/ecma262/2026/multipage/> (дата звернення: 15.09.2026).
6. MDN Web Docs. JavaScript Guide. URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide> (дата звернення: 15.09.2026).
7. Сучасний підручник з JavaScript. Українська версія. URL: <https://uk.javascript.info/> (дата звернення: 15.09.2026).
8. OpenJS Foundation. Node.js API Documentation. URL: <https://nodejs.org/api/> (дата звернення: 15.09.2026).
9. Git Project. Git Reference. URL: <https://git-scm.com/docs> (дата звернення: 15.09.2026).
10. WHATWG. HTML Living Standard: Edition for Web Developers. URL: <https://html.spec.whatwg.org/dev/> (дата звернення: 15.09.2026).
11. MDN Web Docs. CSS Layout. URL: https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout (дата звернення: 15.09.2026).