МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики
Кафедра програмування

Затверджено
На засіданні кафедри програмування
факультету прикладної математики
Львівського національного університету
імені Івана Франка
(протокол № 1 від 29 серпня 2024 р.)

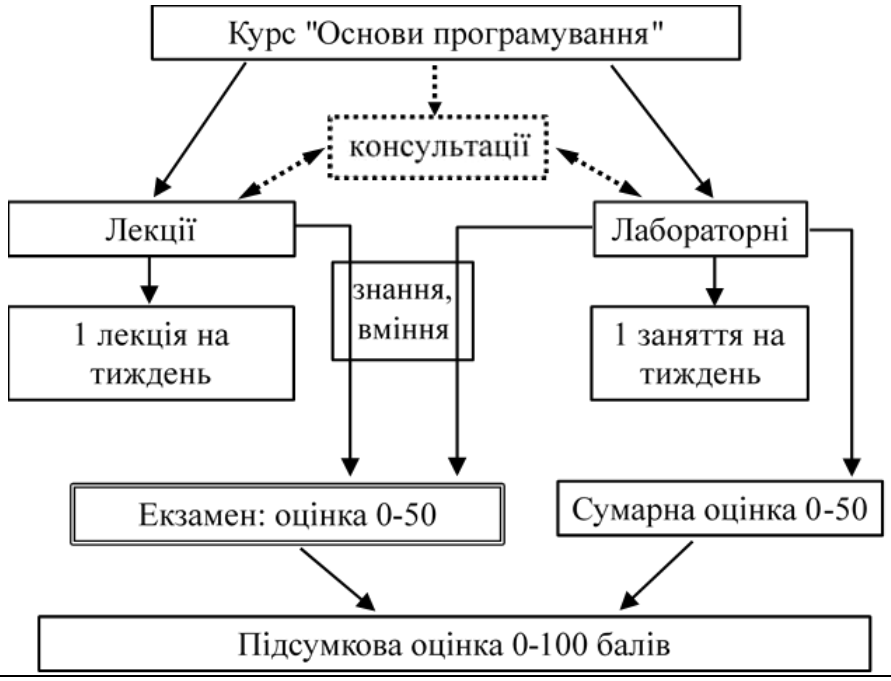
Зав. кафедри к. ф.-м. н., доц. Ярошко С. А.

Силабус навчальної дисципліни
«Основи програмування»,
що викладається в межах ОПП “Прикладна математика”
для здобувачів першого (бакалаврського) рівня вищої освіти
з предметної спеціальності 113 Прикладна математика
галузі знань 11 Математика та статистика

Львів 2024 р.

Назва дисципліни	Основи програмування
Адреса викладання дисципліни	Львівський національний університет імені Івана Франка, вул. Університетська 1, м. Львів, Україна, 79000
Факультет та кафедра, за якою закріплена дисципліна	Факультет прикладної математики та інформатики, кафедра програмування
Галузь знань, шифр та назва спеціальності	Галузь знань: 11 Математика та статистика Спеціальність: 113 Прикладна математика
Викладачі дисципліни	Черняхівський Володимир Вікторович, к. ф.-м. н., доц., доцент кафедри програмування
Контактна інформація викладачів	Електронна пошта: volodymyr.chernyakhivskyy@lnu.edu.ua веб-сторінка: https://ami.lnu.edu.ua/employee/cherniakhivskyi
Консультації з питань навчання по дисципліні відбуваються	Консультації проводять раз на тиждень згідно з оприлюдненим розкладом консультацій викладача. Можливі онлайн консультації через Zoom чи Microsoft Teams. Для погодження часу онлайн консультацій потрібно писати на електронну пошту викладача.
Сторінка курсу	https://ami.lnu.edu.ua/course/basics-of-programming-applied-mathematics-system-analysis
Інформація про дисципліну	Курс “Основи програмування” є нормативною дисципліною зі спеціальності 113 Прикладна математика для освітньої програми «Прикладна математика», яку викладають у другому семестрі в обсязі 4 кредитів (за Європейською кредитно-трансферною системою ECTS).
Коротка анотація дисципліни	Викладання дисципліни має на меті сформувати в студентів базову систему знань та навиків в області сучасних прийомів програмування на основі парадигм: узагальнене програмування; модульне програмування; імперативне програмування; процедурне програмування (функціонально-параметрична декомпозиція). Вивчають зміст і методи реалізації парадигм до будови моделей і алгоритмів програмування розв'язків задач на основі мови С++, і відповідні прийоми застосування С++ в обсязі тем курсу. Дисципліна є наступною після курсу "Вступ до програмування" і розглядає такі розділи програмування: вивчення базових засобів алгоритмічної мови С++; типи і структури даних мови С++ та їх зв'язок з технологіями і парадигмами; керування потоком виконання програми; статичні і динамічні об'єкти; файлова система мови С++; проєктування і застосування функцій; захист від помилок.
Мета та цілі дисципліни	Метою нормативної дисципліни «Основи програмування» є: ● вивчення алгоритмічної мови С++ в базовій частині в обсязі програми курсу; ● вивчення принципів проєктування і програмування задач прикладного характеру мовою С++, вибір методу і алгоритму розв'язування задач; ● отримання навиків проєктування, програмування, тестування і налагодження програм мовою С++; ● вивчення системи розробки програм С++ і прийомів роботи в ній; ● вивчення і застосування стандартних бібліотек функцій мови С++; ● отримання навиків роботи з документацією мови С++ і документацією системи розробки програм.
Література для вивчення дисципліни	*Основна література* 1. Мартін Роберт С. Чистий код. Створення, аналіз і рефакторинг: Пер. з англ. / Роберт С. Мартін // К.: Фабула. – 2019. – 368 с. [Електронний ресурс]. – Режим доступу: https://fabulabook.com/product/chystyj-kod/ 2. w3schools. Підручник С++ [Електронний ресурс]. – Режим доступу: https://www.w3schools.com/cpp/default.asp

	3. Документація Microsoft C++, C та Assembler [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/uk-ua/cpp/?view=msvc-170&viewFallbackFrom=vs-2019 4. Васильєв Олексій. Програмування на С++ в прикладах і задачах / Олексій Васильєв // Ліра-К. – 2017. – 382с. 5. Ярошко Сергій, Ярошко Оксана. Методи розробки алгоритмів. Програмування мовою С++: навч. посібник / С.А.Ярошко, О.С.Ярошко // Львів: ЛНУ імені Івана Франка. – 2022. – 248с. 6. Сайт до вивчення С++ [Електронний ресурс]. – Режим доступу: https://www.bestprog.net/uk/sitemap_ua/c/ 7. Дудзяний І.М. Програмування мовою С++. Частина 1: Парадигма процедурного програмування: навчальний посібник / І.М. Дудзяний // Львів: ЛНУ імені Івана Франка. – 2013. – 468 с. 8. Вивчіть С++. Веб-сайт, присвячений навчанню програмувати на С++ [Електронний ресурс]. – Режим доступу: https://www.learncpp.com/ *Додаткова література* 9. Ярошко С.А. Методи розробки алгоритмів. Навчальний посібник / С.А. Ярошко // Львів, ЛНУ імені Івана Франка. – 2019. [Електронне видання]. 10. Вікіпедія. С++ [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/C%2B%2B
Обсяг курсу	4 кредити ЄКТС – 120 годин. З них 32 годин лекцій, 32 годин лабораторних занять та 56 годин самостійної роботи.
Очікувані результати навчання	Після завершення цього курсу студент буде: *знати* ● алгоритмічну мову С++ разом з бібліотеками функцій в обсязі програми курсу; ● принципи розробки програм мовою С++ з використанням парадигм узагальненого, модульного, імперативного, процедурного програмування; ● методи програмування структурних елементів прикладних задач; ● проєктування програмованих функцій і шаблонів функцій; ● структуру і застосування стандартних бібліотек функцій мови С++; *вміти* ● проєктувати, кодувати, тестувати і налагоджувати програми С++ в обсязі всіх базових можливостей мови за програмою курсу; ● обирати обгрунтовану парадигму програмування відповідно до змісту задачі; ● застосовувати стандартні бібліотеки функцій мови С++; ● застосовувати для програмування складові частини мови С++: вбудовані скалярні і структурні дані, оператори керування потоком виконання, статичні і динамічні об'єкти, файлову систему, програмовані функції, виключні ситуації; ● складати шаблони функцій для різних типів даних відповідно до задач.
Компетентності	*Інтегральна:* Здатність застосовувати загальні принципи програмування практичних задач засобами мови С++ на професійному рівні. Здатність застосовувати сучасні методи програмування задач різного призначення, проектувати структури даних і алгоритмічні моделі на основі різних парадигм програмування. *Загальні (ЗК):* • ЗК01. Здатність учитися і оволодівати сучасними знаннями. • ЗК08. Знання та розуміння предметної області та розуміння професійної діяльності. *Спеціальні (фахові, предметні) компетентності (ФК):* • ФК04. Здатність розробляти алгоритми та структури даних, програмні засоби та програмну документацію.

	• ФК08. Здатність використовувати сучасні технології програмування та тестування програмного забезпечення. • ФК21. Здатність аналізувати можливості наявних технологій розробки програмного забезпечення, обирати та застосовувати найбільш ефективний інструментарій відповідно до специфіки завдання.
Програмні результати навчання	• РН09. Будувати ефективні щодо точності обчислень, стійкості, швидкодії та витрат системних ресурсів алгоритми для чисельного дослідження математичних моделей та розв'язання практичних задач. • РН11. Вміти застосовувати сучасні технології програмування та розроблення програмного забезпечення, програмної реалізації чисельних і символьних алгоритмів. • РН14. Виявляти здатність до самонавчання та продовження професійного розвитку.
Ключові слова	Процесор, пам'ять, компіляція, виконання програми, змінна величина, тип даних, операція, операнд, структура програми, функція, формат запису, локальна змінна, глобальна змінна, константа, присвоєння, вираз, перетворення типу, приведення типу, потік введення, потік виведення, консоль, умовний оператор, тернарний оператор, оператор перемикання, оператор for, параметр циклу, цикл while, цикл do-while, масив, індексування, ініціалізація масиву, матриця, пошук, сортування, сортування перестановками, сортування вибором, вказівник, посилання, динамічний об'єкт, адресна арифметика, динамічний масив, подвійне адресування, символьний тип, символьна функція, символьний рядок, тип структури, тип перелічення, оператор typedef, потік, файл, форматування, маніпулятор формата, файлова операція, послідовний доступ, прямий доступ, програмована функція, параметр функції, прототип функції, принцип локалізації, вбудована функція, перевантаження функції, шаблон функції, декомпозиція задачі, рекурсія, вказівник на функцію, виключна ситуація, генерування помилки, опрацювання/перехоплення помилки, складність алгоритму, алгоритм поліноміальний, алгоритм експоненціальний, оцінка складності.
Формат курсу	Очний: проведення лекцій, лабораторних робіт та консультацій в приміщеннях університету, а в умовах форсмажорних обставин – онлайновий на платформі Microsoft Teams. Зв'язки елементів курсу: 

Теми

Тиж-день	Тема, план, короткі тези	Форма заняття	Трива-лість год	Термін виконання
1	Структурна схема сучасного комп'ютера. Адресована пам'ять. Компіляція і виконання програм. Змінні величини і типи даних. Основні операції та їх особливості. Загальна характеристика мови С++.	Лекція	2	
	Середовище програмування для С++ (Visual Studio). Будова середовища. Засоби розробки програми. Основні прийоми роботи.	Лабораторна робота	2	Наступне лабораторне заняття
2	Загальна структура програми С++. Первинні правила будови С++-програми. Змінні величини і змінні типу константи. Константи. Оператор присвоєння, вирази, перетворення типів. Деякі проблеми точності обчислень. Приведення типів. Комбіновані оператори присвоєння. Потоки введення-виведення. Друкування на консолі. Читання числових даних з консолі.	Лекція	2	
	Задачі і програмна реалізація прикладів за змістом теми 2.	Лабораторна робота	2	Наступне лабораторне заняття
3	Умовні оператори. Концепція умовних операторів. Оператор if повний і скорочений. Тернарний оператор. Послідовність операторів if. Оператор перемикання. Оператор перемикання з спільними послідовностями.	Лекція	2	
	Задачі і програмна реалізація прикладів за змістом теми 3.	Лабораторна робота	2	Наступне лабораторне заняття
4	Оператори циклу. Цикл for. Особливості заголовків циклу for. Приклади задач для for. Цикл while. Приклади задач для while. Цикл do-while. Приклади задач для do-while.	Лекція	2	
	1)Задачі і програмна реалізація прикладів за змістом теми 4. 2)Контрольна робота за темами 1-4 на основі скалярних даних.	Лабораторна робота; контрольна	2	Наступне лабораторне заняття
5	Вектори і матриці. Одновимірні масиви. Прийоми програмування векторів. Особливості індексування. Ініціалізація масивів. Двовимірні масиви – матриці. Приклади задач з використаням матриць.	Лекція	2	
	Задачі і програмна реалізація прикладів за змістом теми 5.	Лабораторна робота	2	Наступне лабораторне заняття
6	Задачі пошуку і сортування. Загальна постановка задачі пошуку. Пошук елементів векторів. Мінімальні і максимальні значення. Постановка задачі сортування. Сортування перестановками. Сортування вибором. Пошук у відсортованих масивах.	Лекція	2	
	Задачі і програмна реалізація прикладів за змістом теми 6.	Лабораторна робота	2	Наступне лабораторне заняття
7	Вказівники і динамічні об'єкти. Будова вказівника. Способи роботи з вказівниками. Адресна арифметика. Порівняння вказівників. Зв'язок вказівників і масивів. Динамічні масиви. Подвійне адресування. Приклади задач будови і опрацювання динамічних об'єктів.	Лекція	2	
	Задачі і програмна реалізація прикладів за змістом теми 7.	Лабораторна робота	2	Наступне лабораторне заняття
8	Символьні типи даних. Тип char. Символьні функції для типу char. Приклади задач для типу char. Символьні рядки. Базові операції з рядками. Типові функції для роботи з рядками. Приклади задач на опрацювання рядків.	Лекція	2	
	1)Задачі і програмна реалізація прикладів за змістом теми 8. 2)Контрольна робота за темами 5-8.	Лабораторна робота; контрольна	2	Наступне лабораторне заняття
9	Типи даних програміста. Оголошення типу структури. Ініціалізація структурних величин. Масиви структур. Вказівники на структури. Вкладені структури. Приклади оголошень і опрацювань структур. Перелічення. Оператор typedef. Комплексний приклад задачі для типів даних програміста.	Лекція	2	
	Задачі і програмна реалізація прикладів за змістом теми 9.	Лабораторна робота	2	Наступне лабораторне заняття
10	Файлова система С++. Потоки. Функції форматування виведення. Маніпулятори формата. Робота з файлами. Операції з файлами. Читання і запис текстових файлів.	Лекція	2	

		приклади задач роботи з файлами. Двійковий безформатний ввід-вивід, приклади роботи з двійковими файлами.			
		Задачі і програмна реалізація прикладів за змістом теми 10.	Лабораторна робота	2	Наступне лабораторне заняття
	11	Функції визначені програмою. Концепція програмованих функцій. Загальний вигляд функції. Передавання параметрів за значенням. Передавання параметрів за посиланням. Приклади задач на будову функцій. Функції типу void. Функції і структури, масиви як параметри функцій, приклади задач. Прототипи функцій. Принцип локалізації.	Лекція	2	
		Задачі і програмна реалізація прикладів за змістом теми 11.	Лабораторна робота	2	Наступне лабораторне заняття
	12	Методи роботи з функціями. Вбудовані функції. Аргументи за замовчуванням. Перевантаження функцій. Задачі і програмування методів роботи з функціями. Неоднозначність перевантажених функцій. Шаблони функцій.	Лекція	2	
		1)Задачі і програмна реалізація прикладів за змістом теми 12. 2)Контрольна робота за темами 9-12.	Лабораторна робота; контрольна	2	Наступне лабораторне заняття
	13	Проєктування функцій. Функціонально-параметрична декомпозиція задачі. Приклади декомпозиції. Рекурсивні функції. Приклади задач на рекурсію. Вказівники на функції, приклади задач. Вказівники на перевантажені функції.	Лекція	2	
		Задачі і програмна реалізація прикладів за змістом теми 13.	Лабораторна робота	2	Наступне лабораторне заняття
	14	Технологія програмування із захистом від помилок. Принципи опрацювання виключних ситуацій. Приклад опрацювання виключної ситуації. Генерування помилки функцією, приклад програмної реалізації. Повторне генерування виключної ситуації.	Лекція	2	
		Задачі і програмна реалізація прикладів за змістом теми 14.	Лабораторна робота	2	Наступне лабораторне заняття
	15	Обчислювальна складність алгоритмів і програм. Критерії аналізу алгоритму і його складності. Підрахунок числа операцій. Приклади оцінок складності алгоритму. Вимірювання часу виконання програми.	Лекція	2	
		1)Задачі і програмна реалізація прикладів за змістом теми 15. 2)Контрольна робота за темами 13-15.	Лабораторна робота; контрольна		Наступне лабораторне заняття
	16	Підсумковий огляд базових засобів мови С++, порівняння із засобами інших мов. Роль С++ в розробці сучасних програмних проєктів. Огляд питань проєктування і розробки програмного забезпечення із застосуванням парадигм узагальненого, модульного, імперативного, процедурного програмування.	Лекція	2	
		Додаткові задачі за темами курсу. Підготовка до іспиту.	Лабораторна робота	2	Початок сесії
Поточний контроль, форми	Поточний контроль знань студентів виконується під час проведення лабораторних занять протягом семестру і має на меті перевірку рівня підготовленості студента до виконання завдань за темою, запланованою на окреме кожне заняття. Крім того, поточний контроль виконують для перевірки виконання студентом індивідуальних завдань. Форми поточного контролю: – усне індивідуальне опитування; – письмове загальне опитування у формі задач; – опитування у формі тестування; – письмова контрольна робота; – перевірка плану виконання з теми лабораторного заняття; – експертна оцінка виконання завдань і отримання достовірних результатів; – тестування результатів програмної реалізації завдань; – перевірка правильності виконання індивідуальних завдань, виданих на самостійне опрацювання; – захист студентом індивідуальних завдань. Максимальна кількість балів за поточний контроль – 50.				

<table>
<tr><td>Екзаменаційний контроль, форма</td><td>Іспит в кінці семестру.
Екзаменаційний контроль проводиться за розкладом, визначеним навчальним графіком для складання іспитів.
На іспит студент отримує 4 задачі з різних розділів курсу "Основи програмування". Всі 4 задачі потрібно реалізувати на комп'ютері мовою С++ і отримати працездатні програми. Для складання програм студент може обрати середовище програмування для мови С++, яке підтримує стандарт мови і бібліотеки функцій. Для всіх задач достатньо консольного режиму виконання.
До кожної реалізованої задачі студент має самостійно скласти тестові вхідні дані (декілька варіантів) і продемонструвати виконання програми на кожному тесті.
Шкала оцінювання на іспиті: алгоритм і програма задач А, Б, Г, Д – по 10 балів кожна, самостійне складання тестів до задач – 4 бали, реалізація технології try-throw-catch для однієї чи більше задач – 4 бали, дотримання системних (дистанційних) правил роботи – 2 бали. Разом – 50 балів.
Оцінка за екзаменаційне завдання може отримати додаткові бали, якщо завдання чи проєкти студента, виконані впродовж семестру, мали закінчений характер і високий рівень фахової реалізації.
Максимальна кількість балів за екзаменаційний контроль – 50.</td></tr>
<tr><td>Підсумковий контроль, шкала оцінок</td><td>Підсумковий контроль включає в себе результати поточного контролю знань студентів, що проводився протягом семестру, та екзаменаційний контроль.
Максимальна кількість балів підсумкового контролю – 100.
Шкала оцінювання підсумкового контролю:
<table>
<tr><th>Бали</th><th>ECTS</th><th>Національна шкала</th><th>Коментар</th></tr>
<tr><td>90 - 100</td><td>A</td><td>5</td><td>відмінно</td></tr>
<tr><td>81 - 89</td><td>B</td><td>4</td><td>дуже добре</td></tr>
<tr><td>71 - 80</td><td>C</td><td>4</td><td>добре</td></tr>
<tr><td>61 - 70</td><td>D</td><td>3</td><td>задовільно</td></tr>
<tr><td>51 - 60</td><td>E</td><td>3</td><td>достатньо</td></tr>
<tr><td>26 - 50</td><td>FX</td><td>2+</td><td>незадовільно</td></tr>
<tr><td>0 - 25</td><td>F</td><td>2-</td><td>повторний курс</td></tr>
</table></td></tr>
<tr><td>Пререквізити</td><td>Для вивчення курсу студенти потребують базових знань з дисципліни "Вступ до програмування". Одночасно з вивченням курсу студенти вивчають дисципліну "Алгоритми обчислювальних процесів" і проходять навчальну практику, впродовж яких виконують завдання з програмування, поглиблюють набуті в курсі знання та удосконалюють навички.</td></tr>
<tr><td>Навчальні методи та техніки, які використовують під час викладання курсу</td><td>Лекції з мультимедійними презентаціями та з демонстрацією прийомів програмування засобами мови С++; лабораторні заняття у вигляді проєктування алгоритмів і програм, виконання практичних завдань та проєктів; індивідуальні домашні завдання на програмування розв'язків задач; самостійне опрацювання навчальних матеріалів: підручників, конспектів лекцій, електронних ресурсів, готових програм мовою С++, додаткових матеріалів, розміщених у хмарному сховищі (Microsoft Teams, Google Диск). Обговорення теоретичного та практичного матеріалу в онлайн сервісах, формулювання творчих завдань для студентів, виконання яких готує до нового теоретичного і практичного матеріалу.</td></tr>
<tr><td>Необхідне обладнання</td><td>Для проведення лекцій: комп’ютер, проєктор, доступ до мережі інтернет.
Для проведення лабораторних та виконання завдань: комп’ютер; ОС Windows/Linux; доступ до інтернету; середовища програмування мовою С++ (Microsoft Visual Studio, Code Blocks тощо).
Кожен студент повинен мати свій доступ до корпоративного середовища Microsoft 365 через надану персональну поштову адресу *@lnu.edu.ua, а також свій акаунт (поштову адресу) в Google *@gmail.com.</td></tr>
</table>

	Уся література і робочі матеріали, які студенти не зможуть знайти самостійно, буде надана викладачем виключно в освітніх цілях без права її передачі третім особам. Студенти заохочуються до використання також й іншої літератури та джерел, яких немає серед рекомендованих.
Критерії оцінювання (окремо для кожного виду навчальної діяльності)	**Оцінювання** проводиться за 100-бальною шкалою. 50 балів нараховують за виконання лабораторних завдань, домашніх завдань, контрольних робіт впродовж семестру, ще 50 балів – за виконання екзаменаційного завдання. Лабораторні завдання індивідуальні. Домашні завдання можуть бути індивідуальні та командні. Упродовж семестру студент виконує не менше 10 лабораторних і домашніх, які оцінюють різною шкалою залежно від складності (критерії надаються студентам до кожного лабораторного чи домашнього завдання). Для кожного завдання визначено термін виконання – зазвичай до наступного лабораторного заняття. Вчасно виконані завдання оцінюють так (у відсотках від максимальної оцінки): • 100% – умови завдання виконано повністю, алгоритми складено правильно, програма містить належні коментарі, роботу програми перевірено на достатньому наборі тестових даних, автор відповідає на всі запитання щодо використаних підходів, чітко інтерпретує отримані результати, немає ознак недоброчесності; • 80% – наведено логічно правильну послідовність розв'язування, алгоритми складено правильно, бракує окремих коментарів чи тестів, автор не досить повно пояснює використані підходи, немає ознак недоброчесності; • 60% – у правильній послідовності розв'язування допущено окремі помилки, які автор уміє виправити після зауваження викладача, бракує коментарів чи тестів, на запитання щодо використаних підходів автор відповідає з помилками, немає ознак недоброчесності; • 40% – у правильній послідовності розв'язування пропущено окремі етапи, завдання виконано частково, автор не розуміє недоліків поданої роботи, не вміє їх виправити, немає ознак недоброчесності; • 20% – завдання виконано частково, немає тестів, програма працює правильно для окремих наборів вхідних даних, автор не може самостійно інтерпретувати отримані результати, виправити помилки, немає ознак недоброчесності; • 0% – завдання не виконано, написана програма не відповідає умові, або ж виявлено ознаки недоброчесності: запозичення, фраґменти коду, дію яких автор пояснити не може, автор не володіє відповідним теоретичним матеріалом тощо; • можуть бути нараховані додаткові бали за повністю виконане завдання, яке містить кілька способів розв'язування, використовує особливо ефективний спосіб, демонструє креативність автора тощо. Запізнення терміну виконання завдання зменшує максимальну оцінку за завдання, ступінь зменшення визначає викладач до кожної роботи. Загальні критерії і шкала оцінювання **екзаменаційних завдань** викладені в розділі "Екзаменаційний контроль, форма". Перелік окремих вимог і зразки екзаменаційних завдань будуть надані студентам до закінчення лекційних і лабораторних занять. **Відвідання занять** є важливою складовою навчання. Очікується, що всі студенти відвідають усі лекції і лабораторні заняття курсу. Активність під час проведення лекцій і лабораторних заохочується балами при оцінюванні відповідного лабораторного завдання чи проєкта. У будь-якому випадку студенти зобов'язані дотримуватися усіх строків визначених для виконання усіх видів робіт, передбачених курсом. Виконані роботи завантажують у відповідне хмарне сховище. Альтернативою відвідування лабораторних занять в університеті може бути дистанційна онлайнова робота за розкладом проведення занять. **Академічна доброчесність**: очікується, що роботи студентів будуть їхнім оригінальними дослідженнями чи міркуваннями. Відсутність посилань на використані джерела, фабрикування джерел, списування, втручання в роботу

	інших студентів, здавання чужих комп'ютерних програм як своїх становлять, але не обмежують, приклади можливої академічної недоброчесності. Виявлення ознак академічної недоброчесності в письмовій роботі студента є підставою для її незарахування викладачем, незалежно від масштабів плагіату чи обману.
Опитування	Анкету-оцінку з метою оцінювання якості курсу буде надано після завершення курсу.