Чернівецький національний університет імені Юрія Федьковича
(повне найменування закладу вищої освіти)

Факультет математики та інформатики
(назва факультету/навчально-наукового інституту)

Кафедра математичного моделювання
(назва кафедри, що забезпечує викладання)

“ЗАТВЕРДЖУЮ”
Декан факультету
математики та інформатики
_____ Ольга МАРТИНЮК
“25” 06 2025 року

РОБОЧА ПРОГРАМА
навчальної дисципліни
Програмування
(назва навчальної дисципліни)

обов’язкова
(вказати: обов’язкова)

Освітньо-професійна програма «Системний аналіз»
(назва програми)

Спеціальність F4 Системний аналіз та наука про дані
(вказати: код, назва)

Галузь знань F Інформаційні технології
(вказати: шифр, назва)

Рівень вищої освіти перший (бакалаврський)
(вказати: перший (бакалаврський) / другий (магістерський) / третій (освітньо-науковий)

Факультет математики та інформатики
(назва факультету/ навчально-наукового інституту, на якому здійснюється підготовка фахівців за вказаною освітньою програмою)

Мова навчання українська
(вказати: на якій мові читається дисципліна)

Чернівці 2025 рік

Робоча програма навчальної дисципліни «*Програмування*» складена відповідно до освітньо-професійної програми «Системний аналіз»

Розробники:
Караванова Тетяна Петрівна, асистент кафедри математичного моделювання;
Івасюк Галина Петрівна, доцент кафедри математичного моделювання, кандидат фіз.-мат. наук, доцент

Викладачі, що забезпечують читання даної навчальної дисципліни:
Івасюк Галина Петрівна, доцент кафедри математичного моделювання, кандидат фіз.-мат. наук, доцент;
Матвій Олександр Васильович, доцент кафедри математичного моделювання, кандидат фіз.-мат. наук, доцент

Погоджено з гарантом ОП _____ **Андрій ПЕРЦОВ**

Затверджено на засіданні кафедри математичного моделювання
Протокол № 15 від «24» червня 2025 року

Завідувач кафедри _____ **Ігор ЧЕРЕВКО**

Схвалено методичною радою факультету математики та інформатики
Протокол № 12 від «25» червня 2025 року

Голова методичної ради _____ **Віра СІКОРА**

Мета навчальної дисципліни:

– ознайомлення з базовими алгоритмічними структурами, типами алгоритмів, найпростішими структурами даних, методами побудови алгоритмів, представлення розроблених алгоритмів мовою програмування С;

– розвиток логічного та аналітичного мислення (індукції, дедукції, аналізу, синтезу, уміння робити висновки, узагальнення);

– розвиток умінь щодо розв'язування алгоритмічних задач з використанням базових теоретичних основ, математичного апарату, фахової літератури та програмного забезпечення.

Пререквізити: Навчальна дисципліна «Програмування» відповідно до структурно-логічної схеми освітньо-професійної програми слухається здобувачами освіти в першому та другому семестрі першого року навчання паралельно із такими дисциплінами як «Алгебра і геометрія», «Дискретна математика», «Архітектура обчислювальних систем». Вивчення дисципліни «Програмування» є основою для засвоєння інших обов'язкових дисциплін, зокрема «Об'єктно-орієнтоване програмування», «Теорія алгоритмів». Для успішного оволодіння курсу здобувач має вільно володіти знаннями з курсу шкільної інформатики.

Результати навчання: Відповідно до освітньо-професійної програми підготовки бакалаврів галузі знань F Інформаційні технології за спеціальністю F4 Системний аналіз та наука про дані (освітньо-професійна програма «Системний аналіз») вивчення дисципліни сприяє формуванню:

загальних та фахових компетентностей:

ЗК01. Здатність до абстрактного мислення, аналізу та синтезу;

ЗК02. Здатність застосовувати знання у практичних ситуаціях;

ЗК07. Здатність до пошуку, оброблення та аналізу інформації з різних джерел;

ЗК10. Здатність працювати автономно;

ЗК11. Здатність генерувати нові ідеї (креативність);

ФК2. Здатність формалізувати проблеми, описані природною мовою, у тому числі за допомогою математичних методів, застосовувати загальні підходи до математичного моделювання конкретних процесів;

ФК6. Здатність до комп'ютерної реалізації математичних моделей реальних систем і процесів; проєктувати, застосовувати і супроводжувати програмні засоби моделювання, прийняття рішень, оптимізації, обробки інформації, інтелектуального аналізу даних;

програмних результатів навчання:

ПР2. Вміти використовувати стандартні схеми для розв'язання комбінаторних та логічних задач, що сформульовані природною мовою, застосовувати класичні алгоритми для перевірки властивостей та класифікації об'єктів, множин, відношень, графів, груп, кілець, решіток, булевих функцій тощо.

Опис навчальної дисципліни

Загальна інформація

Форма навчання	Рік підготовки	Семестр	Кількість		Кількість годин						Вид підсумкового контролю
			кредитів	всього годин	лекції	практичні	семінарські	лабораторні	самостійна робота	індивідуальні завдання	
Денна	1	1	7	210	30	30	–	30	120	–	залік
	1	2	7	210	30	30	–	30	120	–	екзамен

Структура навчальної дисципліни

Назви змістових модулів і тем навчальних занять	Кількість годин											
	денна форма						*заочна форма*					
	усього	у тому числі					усього	у тому числі				
		л	п	лаб	інд	с.р.		л	п	лаб	інд	с.р.
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 1. Основні поняття алгоритмізації та програмування. Базові алгоритмічні структури												
Тема 1. Основні поняття алгоритмізації та програмування. Базові алгоритмічні структури. Мови програмування	8	2	2	2		2						
Тема 2.Мова програмування С. Структура програми мовою С. Правила оформлення коду програми мовою С. Змінні. Іменовані константи. Стандартні типи даних мови С	9	2	2	2		3						
Тема 3. Лінійні програми. Арифметичні вирази, арифметичні операції і стандартні математичні функції. Пріоритет математичних операцій. Інкремент. Декремент. Оператор присвоєння. Порожній оператор. Введення та виведення даних	16	2	2	2		10						

Тема 4. Логічні оператори і вирази. Оператор умовного переходу **if-else**. Оператор безумовного переходу **goto**. Оператор множинного вибору **switch**	12	2	2	2		6						
Тема 5. Вкладені розгалуження. Сходинковий оператор **if-else-if**. Оператор «**? :**»	13	2	2	2		7						
Тема 6. Алгоритми поєднання розгалуження та повторення	13	2	2	2		7						
Тема 7. Циклічні конструкції. Цикл з лічильником **for**. Деякі особливості циклу for. Оператор послідовного виконання «**,**». Цикли **while** та **do-while**	18	2	2	2		12						
Тема 8. Вкладені цикли. Покрокове введення та виведення даних. Рекурентні послідовності	19	2	2	2		13						
Разом за ЗМ 1	**108**	**16**	**16**	**16**		**60**						
Змістовий модуль 2.Масиви. Алгоритми обробки елементів масивів												
Тема 1. Одновимірні масиви	12	2	2	2		6						
Тема 2. Найпростіші алгоритми роботи з одновимірними масивами. Пошук заданого елемента, пошук мінімального/максимального елемента	13	2	2	2		7						
Тема 3. Додаткові способи введення даних. Основи роботи з файлами і потоками. Генератор випадкових чисел. Константні вхідні дані, ініціалізація масивів	13	2	2	2		7						
Тема 4. Двовимірні масиви	16	2	2	2		10						
Тема 5. Базові алгоритми для обробки елементів двовимірних масивів	16	2	2	2		10						
Тема 6. Рядки і символьні масиви. Стандартні функції для роботи з рядками	16	2	2	2		10						
Тема 7. Найпростіші алгоритми роботи із	16	2	2	2		10						

символьними та рядковими величинами												
Разом за ЗМ 2	**102**	**14**	**14**	**14**		**60**						
Змістовий модуль 3. Динамічні змінні. Функції користувача. Структурування даних												
Тема 1. Вказівники. Операції з вказівниками. Практичне застосування вказівників	14	2	2	2		8						
Тема 2. Використання вказівників для роботи з масивами. Динамічні масиви	15	2	2	2		9						
Тема 3. Функції користувача	13	2	2	2		7						
Тема 4. Масиви і функції. Передача масиву у функцію	14	2	2	2		8						
Тема 5. Рекурсивні алгоритми	14	2	2	2		8						
Тема 6. Структури. Масиви структур. Структури в структурах	17	2	2	2		11						
Тема 7. Використання структур для роботи з функціями. Використання вказівників для роботи зі структурами	17	2	2	2		11						
Разом за ЗМ3	**104**	**14**	**14**	**14**		**62**						
Змістовий модуль 4. Динамічні структури даних. Класичні алгоритми обробки даних												
Тема 1. Додаткові можливості структурування даних. Об’єднання (**union**). Злічені типи даних	15	2	2	2		9						
Тема 2. Використання оператора **typedef**. Алгоритми обробки структурованих даних	14	2	2	2		8						
Тема 3. Динамічні структури даних. Стек. Черга	17	2	2	2		11						
Тема 4. Зв’язний список. Алгоритми обробки елементів зв’язного списку	17	2	2	2		11						
Тема 5. Пошукові алгоритми. Рекурсивний пошук. КМП-пошук	11	2	2	2		5						
Тема 6. Алгоритми сортування. Класифікація методів сортування. Прямі методи сортування	11	2	2	2		5						
Тема 7. Лінійні методи сортування. Сортування підрахунком. Цифрове сортування	11	2	2	2		5						

Тема 8. Оператори препроцесора # та ##. Створення власних заголовних файлів	10	2	2	2		4						
Разом за ЗМ4	**106**	**16**	**16**	**16**		**58**						
Усього годин	**420**	**60**	**60**	**60**		**240**						

Тематика лекційних занять з переліком питань

№	Назва теми з основними питаннями
Змістовий модуль 1. Основні поняття алгоритмізації та програмування. Базові алгоритмічні структури	
1	Тема 1. Основні поняття алгоритмізації та програмування. Базові алгоритмічні структури. Мови програмування
2	Тема 2.Мова програмування С. Структура програми мовою С. Правила оформлення коду програми мовою С. Змінні. Іменовані константи. Стандартні типи даних мови С
3	Тема 3. Лінійні програми. Арифметичні вирази, арифметичні операції і стандартні математичні функції. Пріоритет математичних операцій. Інкремент. Декремент. Оператор присвоєння. Порожній оператор. Введення та виведення даних
4	Тема 4. Логічні оператори і вирази. Оператор умовного переходу **if-else**. Оператор безумовного переходу **goto**. Оператор множинного вибору **switch**
5	Тема 5. Вкладені розгалуження. Сходинковий оператор **if-else-if**. Оператор «**? :**»
6	Тема 6. Алгоритми поєднання розгалуження та повторення Приклади задач, де використовується одночасно перевірка умови й циклічне повторення.
7	Тема 7. Циклічні конструкції. Цикл з лічильником **for**. Деякі особливості циклу for. Оператор послідовного виконання «**,**». Цикли **while** та **do-while**
8	Тема 8. Вкладені цикли. Покрокове введення та виведення даних. Рекурентні послідовності
Змістовий модуль 2. Масиви. Алгоритми обробки елементів масивів	
9	Тема 1. Одновимірні масиви: ● Ініціалізація масивів. ● Типові помилки при роботі з масивами (вихід за межі, неправильна ініціалізація). ● Використання циклів для обробки масивів (введення, виведення, обчислення).
10	Тема 2. Найпростіші алгоритми роботи з одновимірними масивами. Пошук заданого елемента, пошук мінімального/максимального елемента
11	Тема 3. Додаткові способи введення даних. Основи роботи з файлами і потоками. Генератор випадкових чисел. Константні вхідні дані, ініціалізація масивів
12	Тема 4. Двовимірні масиви: ● Оголошення двовимірних масивів. ● Використання вкладених циклів для обробки. ● Приклади застосування двовимірних масивів (таблиці, матриці).
13	Тема 5. Базові алгоритми для обробки елементів двовимірних масивів: ● Методи ітеративного обходу елементів двовимірного масиву. ● Алгоритми визначення мінімального та максимального значень у двовимірному масиві. ● Обчислення сумарних та середніх характеристик елементів масиву за рядами, стовпцями та в цілому. ● Приклади застосування двовимірних масивів у практичних задачах програмування.
14	Тема 6. Рядки і символьні масиви. Стандартні функції для роботи з рядками: ● Поняття рядка як символьного масиву в мові С.

	● Методи оголошення та ініціалізації символьних масивів. ● Доступ до окремих символів рядка та операції над ними. ● Стандартні бібліотечні функції для обробки рядків (strlen, strcpy, strcat, strcmp та ін.). ● Практичні приклади використання рядків у програмуванні.
15	Тема 7. Найпростіші алгоритми роботи із символьними та рядковими величинами: ● Пошук символів або підрядків у рядку. ● Модифікація символів та перетворення рядків.
Змістовий модуль 3. Динамічні змінні. Функції користувача. Структурування даних	
16	Тема 1. Вказівники. Операції з вказівниками. Практичне застосування вказівників ● Оголошення та ініціалізація вказівників (створення вказівників на змінні різних типів, виведення адреси та значення) ● Розіменування вказівників (доступ і зміна значення змінної через вказівник) ● Арифметика вказівників (операції ++, --, додавання/віднімання цілих чисел до вказівника) ● Практичне застосування (прості приклади: обмін значеннями двох змінних через вказівники, робота з подвійним вказівником)
17	Тема 2. Використання вказівників для роботи з масивами. Динамічні масиви ● Доступ до елементів масиву через вказівник. ● Перегляд і обробка масиву за допомогою вказівників. ● Використання malloc(), calloc(), free() для створення динамічного масиву. ● Приклади: обчислення суми/середнього значення елементів динамічного масиву
18	Тема 3. Функції користувача ● Використання вказівників для передачі параметрів у функцію. ● Написання функцій із різними типами повернених значень (int, double, char). ● Використання вказівників для повернення кількох результатів з функції (наприклад, одночасне знаходження суми та добутку чисел).
19	Тема 4. Масиви і функції. Передача масиву у функцію ● Функція для ініціалізації масиву (заповнення випадковими числами або введення з клавіатури). ● Функції для обробки масиву: підрахунок кількості від'ємних/додатних елементів. ● Передача рядків у функцію як символьних масивів: реалізація власних аналогів strlen, strcpy. ● Приклад: функція, яка шукає у масиві перший елемент, що задовольняє певну умову (наприклад, парний або більший за задане число).
20	Тема 5. Рекурсивні алгоритми ● Прості рекурсивні функції (обчислення факторіалу, степеня числа). ● Рекурсивні алгоритми для роботи з числами (НСД, сума цифр числа). ● Приклади: рекурсивне обчислення чисел Фібоначчі, рекурсивний вивід елементів масиву.
21	Тема 6. Структури. Масиви структур. Структури в структурах ● Оголошення та ініціалізація структур, доступ до полів. ● Масиви структур: збереження та вивід даних (наприклад, відомості про студентів). ● Структури в структурах: створення складніших моделей (адреса у структурі «Студент»). ● Приклад: підрахунок кількості структур, які задовольняють певну умову (наприклад, студенти з балом вище 90).
22	Тема 7. Використання структур для роботи з функціями. Використання вказівників для роботи зі структурами ● Передача структур у функції за значенням і за адресою. ● Функції для обробки структур (заповнення, вивід, пошук за критерієм).

	● Використання вказівників на структури, доступ до полів через оператор `->`. ● Приклад: функція, яка отримує масив структур і обчислює середнє значення певного поля.
Змістовий модуль 4. Динамічні структури даних. Класичні алгоритми обробки даних	
23	Тема 1. Додаткові можливості структурування даних. Об'єднання (**union**). Злічені типи даних ● Оголошення та ініціалізація `union`, доступ до полів. ● Використання `enum` для створення злічених типів даних. ● Практичні приклади: збереження даних різних типів у `union` (наприклад, числа та символи). ● Приклад: створення перерахування для днів тижня та використання його у програмі.
24	Тема 2. Використання оператора **typedef**. Алгоритми обробки структурованих даних ● Створення псевдонімів типів за допомогою `typedef` для структур, масивів, вказівників. ● Практичні приклади: спрощення оголошень складних типів (наприклад, масив структур). ● Використання `typedef` для підвищення читабельності алгоритмів обробки даних. ● Приклад: оголошення типу `Student` через `typedef` та написання функції для виводу даних.
25	Тема 3. Динамічні структури даних. Стек. Черга ● Створення динамічного стека та черги через масиви та вказівники. ● Операції над стеком: додавання (push), видалення (pop), перегляд верхнього елементу. ● Операції над чергою: додавання (enqueue), видалення (dequeue), перегляд першого елементу.
26	Тема 4. Зв'язний список. Алгоритми обробки елементів зв'язного списку ● Оголошення структури вузла та створення простого однозв'язного списку. ● Додавання і видалення елементів списку на початку, в кінці та за ключем. ● Перегляд і обробка елементів списку (пошук, підрахунок елементів). ● Приклади: обчислення суми елементів списку або пошук максимального значення.
27	Тема 5. Пошукові алгоритми. Рекурсивний пошук. КМП-пошук
28	Тема 6. Алгоритми сортування. Класифікація методів сортування. Прямі методи сортування
29	Тема 7. Лінійні методи сортування. Сортування підрахунком. Цифрове сортування
30	Тема 8. Оператори препроцесора # та ##. Створення власних заголовних файлів

Тематика практичних занять з переліком питань

№	Назва теми (питання/завдання)
1	Основні поняття алгоритмізації та програмування. Базові алгоритмічні структури. * Поняття алгоритму та його властивості. * Способи подання алгоритмів (словесний, графічний, псевдокод). * Базові алгоритмічні структури: слідування, розгалуження, повторення.
2	Мова програмування С. Структура програми мовою С. Правила оформлення коду програми мовою С. Змінні. Іменовані константи. Стандартні типи даних мови С * Загальна структура програми мовою С. * Підключення бібліотек (#include).

	* Правила запису інструкцій (синтаксис, відступи, коментарі). * Оголошення змінних і констант. * Типи даних: `int`, `float`, `double`, `char`.
3	Лінійні програми. Арифметичні вирази, арифметичні операції і стандартні математичні функції. Пріоритет математичних операцій. Інкремент. Декремент. Оператор присвоєння. Порожній оператор. Введення та виведення даних * Складання простих лінійних програм. * Арифметичні оператори (`+`, `-`, `*`, `/`, `%`). * Пріоритет операцій. * Використання стандартних математичних функцій (`math.h`). * Інкремент і декремент (`++`, `--`). * Оператор присвоєння `=` та комбіновані оператори (`+=`, `-=`). * Оператор `;` як порожня інструкція. * Введення/виведення даних: `scanf()`, `printf()`.
4	Логічні оператори і вирази. Оператор умовного переходу **if-else**. Оператор безумовного переходу goto. Оператор множинного вибору **switch** * Логічні оператори: `&&`, `||`, `!`. * Логічні вирази та їх значення (істина/хиба). * Оператор `if`, `if-else`. * Особливості вкладеного `if`. * Оператор `goto`: застосування та обмеження. * Оператор `switch-case`: синтаксис, приклади використання.
5	Вкладені розгалуження. Сходинковий оператор **if-else-if**. Оператор «? :» * Конструкції з кількома умовами. * Побудова вкладених `if`. * Сходинковий `if-else-if`. * Тернарний оператор `? :` – скорочений запис умовного виразу.
6	Алгоритми поєднання розгалуження та повторення * Розробка алгоритмів для обробки послідовностей чисел. * Використання умов всередині циклів (`if` у циклі).
7	Циклічні конструкції. Цикл з лічильником **for**. Деякі особливості циклу for. Оператор послідовного виконання «,». Цикли **while** та **do-while** * Організація циклу з лічильником `for` (пропуск виразів у заголовку циклу). * Використання оператора `,` у виразах. * Цикл `while` та `do-while`. * Відмінності між циклами `for`, `while`, `do-while`.
8	Вкладені цикли. Покрокове введення та виведення даних. Рекурентні послідовності * Використання вкладених циклів. Задачі з обробки двовимірних структур даних (таблиць, матриць). * Покрокове введення/виведення даних. * Рекурентні послідовності: поняття, приклади (факторіал, числа Фібоначчі).
9	Одновимірні масиви * Оголошення та ініціалізація одновимірних масивів, доступ до елементів масиву. * Введення та виведення масиву. * Пошук заданого елемента. * Пошук мінімального/максимального елемента.
11	Додаткові способи введення даних. Основи роботи з файлами і потоками. Генератор випадкових чисел. Константні вхідні дані, ініціалізація масивів
12	Двовимірні масиви: * Оголошення та ініціалізація.

	* Введення та виведення елементів двовимірного масиву. * Перегляд елементів по рядках/ стовпцях.
13	Базові алгоритми для обробки елементів двовимірних масивів
14	Рядки і символьні масиви. Стандартні функції для роботи з рядками * Оголошення та ініціалізація символьних масивів. * Введення та виведення рядків. * Використання стандартних функцій (`strlen`, `strcpy`, `strcmp`, `strcat`).
15	Найпростіші алгоритми роботи із символьними та рядковими величинами
16	Вказівники. Операції з вказівниками. Практичне застосування вказівників * Оголошення та використання вказівників. * Арифметика вказівників. * Зв'язок вказівників і масивів.
17	Використання вказівників для роботи з масивами. Динамічні масиви * Вивести елементи масиву, використовуючи лише вказівники. * Обчислити суму елементів масиву через вказівник. * Продемонструвати різницю між передачею змінної "за значенням" і "за адресою". * Динамічне виділення пам'яті (`malloc`, `calloc`). * Звільнення пам'яті (`free`).
18	Функції користувача * Оголошення та визначення функцій. * Параметри та тип, що повертається.
19	Масиви і функції. Передача масиву у функцію * Передача масиву у функцію "за адресою". * Робота з масивом у функції.
20	Рекурсивні алгоритми * Реалізація базових рекурсивних функцій (факторіал, степінь числа). * Рекурсивні алгоритми для роботи з числами (НСД, підрахунок цифр числа). * Рекурсивна обробка масиву (пошук суми, добутку). * Приклад: обчислення чисел Фібоначчі або рекурсивний вивід елементів.
21	Структури. Масиви структур. Структури в структурах * Оголошення, ініціалізація та доступ до полів структур. * Створення масиву структур для збереження однорідних даних. * Використання вкладених структур (структура в структурі). * Приклад: підрахунок кількості елементів, що задовольняють певну умову.
22	Використання структур для роботи з функціями. Використання вказівників для роботи зі структурами * Передача структур у функції за значенням і за адресою. * Написання функцій для введення, виведення та пошуку даних у масиві структур.
23	Додаткові можливості структурування даних. Об'єднання (**union**). Злічені типи даних * Оголошення та ініціалізація об'єднань (union), доступ до полів. * Створення та використання зліченних типів даних (enum). * Порівняння роботи struct і union у спільних прикладах. * Приклади: опис елементів меню або днів тижня через enum і збереження даних різних типів у union.
24	Використання оператора **typedef.** Алгоритми обробки структурованих даних *Створення псевдонімів типів за допомогою typedef. *Застосування typedef до структур, масивів, вказівників.

	*Використання псевдонімів для підвищення зручності обробки структурованих даних. *Приклад: опис типу Student і написання функцій обробки його масиву.
25	Динамічні структури даних. Стек. Черга * Реалізація стека з використанням масиву або списку. * Операції над стеком: push, pop, перегляд верхнього елемента. * Реалізація черги та основні операції enqueue, dequeue.
26	Зв'язний список. Алгоритми обробки елементів зв'язного списку * Оголошення структури вузла та створення однозв'язного списку. * Додавання і видалення елементів списку. * Перегляд і пошук елементів у списку. * Приклад: обчислення суми або середнього значення елементів списку.
27	Пошукові алгоритми. Рекурсивний пошук. КМП-пошук * Реалізація лінійного та бінарного пошуку. * Рекурсивна реалізація бінарного пошуку. * Алгоритм Кнута–Морріса–Пратта (КМП) для пошуку підрядка в рядку. * Приклад: пошук заданого елемента або підрядка в тексті.
28	Алгоритми сортування. Класифікація методів сортування. Прямі методи сортування *Сортування обміном (bubble sort). *Сортування вибором (selection sort). *Сортування вставками (insertion sort). *Порівняння ефективності прямих методів на масиві однакових даних.
29	Лінійні методи сортування. Сортування підрахунком. Цифрове сортування * Принцип роботи алгоритму сортування підрахунком (counting sort). * Реалізація цифрового сортування (radix sort). * Порівняння результатів різних методів на вибірках різного розміру. * Приклад: сортування набору чисел за кількістю розрядів.
30	Оператори препроцесора # та ##. Створення власних заголовних файлів * Використання директив #define, #include, #if, #ifdef. * Застосування оператора з'єднання ## та перетворення в рядок #. * Створення власного заголовного файлу з константами, макросами, прототипами функцій.

Тематика лабораторних занять з переліком питань

№	Назва теми (завдання)
1	Лінійні та розгалужені програми (ознайомитися з лінійними та розгалуженими алгоритмами, навчитися застосовувати оператори розгалуження для реалізації різних сценаріїв виконання програм і перевірки умов)
2	Вкладені розгалуження та множинний вибір (ознайомитися з вкладеними розгалуженнями та конструкціями множинного вибору, навчитися реалізовувати складніші умови та альтернативні варіанти виконання програми)
3	Циклічні програми (ознайомитися з циклічними структурами програм, навчитися реалізовувати повторювані обчислення та обробку даних за допомогою циклів)
4	Одновимірні масиви (ознайомитися з одновимірними масивами, навчитися організовувати зберігання даних та виконувати базові операції обробки елементів масиву)
5	Двовимірні масиви (ознайомитися з двовимірними масивами, навчитися здійснювати зберігання та обробку даних у вигляді таблиць за допомогою вкладених циклів)
6	Алгоритми обробки символьних масивів (ознайомитися з алгоритмами обробки символьних масивів, навчитися виконувати базові операції над рядками та окремими символами.)
7	Вказівники (ознайомитися з поняттям вказівників у мові С, навчитися використовувати їх для доступу до змінних та організації ефективної роботи з пам’яттю)
8	Функції користувача (ознайомитися з поняттям функцій користувача, навчитися їх оголошувати, визначати та використовувати для структуризації програмного коду)
9	Структуровані типи користувача (ознайомитися зі структурованими типами користувача, навчитися оголошувати структури та виконувати доступ і обробку їхніх елементів у програмі)
10	Необчислювальні алгоритми для обробки даних (ознайомитися з алгоритмами сортування та пошуку даних, навчитися реалізовувати базові методи обробки масивів для впорядкування та пошуку елементів)

Завдання для самостійної роботи студентів

№	Назва теми	Завдання для самостійної роботи	К-сть год.
1.	Базові алгоритмічні структури. Мови програмування	Розробити блок-схеми алгоритмів знаходження розв’язку наступних задач: • Обчислити суму перших k непарних чисел; • Обчислити добуток усіх чисел від 1 до k, кратних 3; • Визначити кількість цифр у числі k^3; • Знайти найменший дільник числа k , що більший за 1; k – номер залікової книжки студента.	5
2.	Лінійні програми	• Реалізувати програму для обчислення площі прямокутника або трикутника за введеними сторонами. • Написати програму для конвертації температури (°C ↔ °F).	10

		• Розробити програму для обчислення вартості товару з ПДВ або знижкою.	
3.	Розгалужені програми	• Написати програму, яка визначає, чи введене число додатне, від'ємне чи нуль. • Реалізувати програму, що перевіряє, чи можна утворити трикутник зі сторін a, b, c. • Створити програму, яка обчислює податок залежно від доходу: до 5000 – 10%, 5001–10000 – 15%, більше 10000 – 20%.	20
4.	Циклічні програми	• Реалізувати програму для знаходження добутку непарних чисел від 1 до N. • Створити програму, яка обчислює суму цифр цілого числа.	25
5.	Одновимірні масиви	• Створити масив дійсних чисел і підрахувати кількість елементів більших за середнє. • Реалізувати пошук першого від'ємного елемента в масиві. • Написати програму для обчислення суми елементів із парними індексами. • Розробити програму для циклічного зсуву елементів масиву праворуч.	20
6.	Двовимірні масиви	• Знайти суму елементів заданого стовпця матриці. • Реалізувати програму для транспонування квадратної матриці. • Написати програму, яка обчислює суму всіх елементів, розташованих вище головної діагоналі. • Обчислити добуток двох матриць.	10
7.	Алгоритми обробки масивів	• Реалізувати програму для підрахунку кількості елементів, кратних заданому числу. • Написати функцію для підрахунку кількості унікальних елементів масиву. • Написати функцію для видалення елементів масиву, які повторюються.	10
8.	Рядки і символьні масиви.	• Написати програму, яка підраховує кількість слів у рядку. • Створити функцію для перетворення всіх малих літер у великі. • Реалізувати перевірку, чи містить рядок лише цифри.	20
9.	Вказівники	• Реалізувати функцію, яка підраховує кількість парних елементів у масиві через вказівники. • Створити рядок довжиною k, заповнити випадковими літерами латинського алфавіту та вивести лише голосні. • Написати функцію, яка за допомогою вказівників знаходить у рядку кількість голосних і приголосних.	17
10.	Функції користувача	• Реалізувати функцію, яка повертає новий динамічний масив — копію переданого масиву. • Написати функцію void printLine(char c, int n), яка виводить символ c задану кількість разів. • Створити функцію void circle(float r, float length, float area), яка за радіусом обчислює довжину кола та площу круга (результати повертає через параметри).	15

		• Написати функцію, яка перевіряє, чи є задане число простим. • Реалізувати програму, що містить функції для роботи з рядками: – перетворення літер на великі; – порівняння двох рядків без використання strcmp().	
11.	Рекурсивні алгоритми	• Реалізувати рекурсивну функцію int sumDigits(int n) — обчислення суми цифр числа. • Створити рекурсивну функцію для підрахунку кількості цифр у числі. • Написати функцію, яка рекурсивно виводить усі натуральні числа від 1 до N.	8
12.	Структури	• Створити структуру «Точка» та обчислити відстань між двома точками. • Реалізувати структуру «Товар» і програму для підрахунку загальної вартості товарів. • Написати програму, що сортує масив структур «Студент» за середнім балом. • Розробити функцію, яка знаходить студента за номером залікової книжки.	22
13.	Додаткові можливості структурування даних	• Створити union для збереження різних типів даних у полі структури «Дані». • Реалізувати програму, що використовує enum для задання режиму роботи (AUTO, MANUAL, TEST). • Використати typedef для оголошення скорочених імен структурних типів.	17
14.	Динамічні структури даних	• Створити програму, яка використовує чергу для моделювання черги пацієнтів у реєстратурі. • Реалізувати функцію int countElements(struct Node head) — підрахунок кількості елементів. • Написати функцію для пошуку елемента у списку за значенням. • Створити програму, що реалізує однозв'язний список для зберігання відомостей про товари (назва, ціна, кількість). • Реалізувати: – додавання нового товару; – пошук за назвою; – обчислення загальної вартості всіх товарів. • Нехай номер варіанта студента — k. – Створити список із k елементів, обчислити суму всіх заданих елементів. – Додати до списку елемент зі значенням $\boldsymbol{k^2}$ після третього вузла. – Зі стеку видалити всі елементи, менші за k.	22
15.	Пошукові алгоритми.	• Пошук у масивах структур. • Реалізувати функцію для пошуку студента за номером id (лінійним пошуком). • Реалізувати функцію для пошуку студента з найвищим середнім балом. • Створити функцію для пошуку студента за прізвищем у масиві структур.	5

		• Створити функцію int findAll(int arr, int n, int key, int positions), яка знаходить усі позиції заданого елемента.	
16.	Алгоритми сортування	• Реалізувати програму, що дозволяє користувачу вибрати метод сортування (через меню). • Написати програму, яка зчитує з файлу список студентів (ПІБ і бал), сортує їх за балами та зберігає відсортований список у новий файл. • Реалізувати функцію, що сортує динамічний масив дійсних чисел за допомогою вказівників. • Нехай номер варіанта студента — k. – Створити масив із k+10 випадкових чисел і відсортувати його методом вибору. – Упорядкувати перші k елементів масиву методом вставок.	10
17.	Створення власних заголовних файлів	Створити заголовний файл fileutils.h, що містить функції для роботи з файлами: • відкриття файлу для читання/запису; • підрахунок кількості рядків; • копіювання вмісту з одного файлу в інший.	4

Самостійна робота також складається з повторення матеріалу, засвоєного на лекціях, самостійного опанування частини теоретичного матеріалу, роботи з контрольними запитаннями та завданнями лабораторних робіт.

Методи навчання

Лекції, лабораторні роботи, тестування, аудиторне та онлайн-навчання з використанням систем Moodle та Google Meet.

Методи навчання:

- вербальні методи (лекція, бесіда, диспут, пояснення, розповідь тощо);
- практичні методи (лабораторні роботи);
- наочні методи (демонстрація, ілюстрація);
- робота з інформаційними ресурсами: з навчально-методичною, науковою, нормативною літературою та інтернет-ресурсами;
- самостійна робота над індивідуальним завданням або за програмою навчальної дисципліни;
- навчання з використанням відповідних онлайн-платформ.

Система контролю та оцінювання

Засобами оцінювання та демонстрування результатів навчання є:

- усне та письмове опитування;
- виконання практичних та лабораторних робіт;
- стандартизовані тести.

Поточний контроль знань відбувається на практичних та лабораторних заняттях шляхом усного та письмового опитування теоретичних основ теми,

виконання практичних та лабораторних завдань, а також у вигляді контрольних робіт та тестування.

Форми підсумкового контролю: 1 семестр – залік, 2 семестр – екзамен.

Критерії оцінювання поточного та підсумкового контролю

Система оцінювання рівня навчальних досягнень ґрунтується на принципах ECTS та є накопичувальною. Впродовж першого та другого семестрів студенти виконують 12 лабораторних робіт (по шість у кожному). Лабораторні роботи оцінюються по 6 або 7 балів за кожну (див. таблицю нижче).

Виконуючи завдання лабораторної роботи (ЛР), студент повинен оформити і завантажити для подальшої перевірки на сайт електронного навчання звіт разом із працездатними файлами програмної реалізації завдань ЛР (правила оформлення, варіанти завдань розміщені на платформі https://moodle.chnu.edu.ua/ на сторінці навчальної дисципліни).

50% балів, відведених на оцінювання ЛР, студент отримує за повністю правильно виконану ЛР та оформлений звіт. Решта 50% балів виставляється після захисту студентом виконаного звіту. У випадку неістотної помилки знімається 10-20% балів, а у випадку істотної 20-40% балів, якщо ж студент не опанував теоретичний матеріал, плутається в означеннях, то знімається до 50% балів від усієї суми балів за ЛР.

По завершенню кожного модуля студенти проходять тестове онлайн опитування на сайті електронного навчання ЧНУ. Протягом одного модуля студенти можуть отримати 20 балів за захист лабораторних завдань та 10 балів за успішне проходження тестів. Весь річний курс розбито на чотири модулі.

Залік проводиться у формі тестового опитування з використанням системи електронного навчання Moodle. Кожному студенту випадковим чином генерується 20 запитань. Сумарна максимальна кількість балів за виконання тестів становить 40 балів. Час виконання тесту – 40 хвилин. Загальна оцінка за залік визначається з урахуванням всіх отриманих балів у першому семестрі (модулі 1, 2 та підсумковий тест).

Екзамен проводиться у формі тестового опитування з використанням системи електронного навчання Moodle. Кожному студенту випадковим чином генерується 22 запитання чотирьох рівнів складності. 1 рівень – 4 запитання по 0,25 бала, 2 рівень – 8 запитань по 0,5 бала, 3 рівень – 5 запитань по 1 балу, 4 рівень – 5 запитань по 2 бали. Сумарна максимальна кількість балів за виконання тестів становить 20 балів. Час виконання тесту – 30 хвилин.

Практичною частиною екзамену є розв’язання студентами двох задач, що містяться у базі даних даного курсу і також генерується випадковим чином. Виконання цього завдання оцінюється по 10 балів за кожне. Час виконання становить 30 хвилин.

Сумарна максимальна оцінка за екзамен становить 40 балів. Загальна оцінка за навчальну дисципліну визначається з урахуванням всіх отриманих балів у другому семестрі (модулі 3, 4 та підсумковий модуль).

Розподіл балів, які отримують студенти у першому семестрі

Поточне тестування та самостійна робота								Підсумковий тест (залік)	Сума
Змістовий модуль 1				Змістовий модуль 2				40	100
Т1-Т3	Т4-Т5	Т6-Т8	тест	Т1-Т3	Т4-Т5	Т6-Т7	тест		
6	7	7	10	7	7	6	10		

Розподіл балів, які отримують студенти у другому семестрі

Поточне тестування та самостійна робота								Підсумковий модуль (екзамен)	Сума
Змістовий модуль 3				Змістовий модуль 4				40	100
Т1-Т2	Т3-Т5	Т6-Т7	тест	Т1-Т2	Т3-Т4	Т5-Т8	тест		
7	7	6	10	6	7	7	10		

Шкала оцінювання: національна та ECTS

100-бальна шкала	**Оцінка за національною шкалою**		**Оцінка за шкалою ЄКТС**	
			Оцінка	**Пояснення за розширеною шкалою**
90-100	**Зараховано**	**Відмінно**	**A**	**відмінно**
80-89		**Добре**	**B**	**дуже добре**
70-79			**C**	**добре**
60-69		**Задовільно**	**D**	**задовільно**
50-59			**E**	**достатньо**
35-49	**Незараховано**	**Незадовільно**	**FX**	**(незадовільно) з можливістю повторного складання**
1-34			**F**	**(незадовільно) з обов'язковим самостійним повторним опрацюванням освітнього компонента до перескладання**

Перелік питань для самоконтролю та підсумкового контролю навчальних досягнень студентів

1.Основні поняття алгоритмізації та програмування. Базові алгоритмічні структури
2.Мови програмування. Класифікація мов програмування. Мова програмування С. Структура програми мовою С.
3.Елементи мови програмування С (алфавіт, ідентифікатори, команди, директиви препроцесора). Типи даних
4.Лінійні програми. Структура лінійних програм. Приклади

5.Арифметичні вирази. Інструкція присвоєння. Приклади
6.Введення та виведення даних . Приклади
7.Оператор умовного переходу. Приклади
8.Оператор вибору (перемикач). Приклади
9.Оператор безумовного переходу. Поняття мітки. Оператор “?”. Приклади
10.Оператор циклу з повторенням for. Приклади
11.Оператори циклу з передумовою while та з післяумовою do/while. Приклади
12.Використання у циклах операторів break, exit(), continue. Приклади
13.Вкладені цикли. Приклади
14.Покрокове введення та виведення даних. Приклади
15.Рекурентні послідовності. Приклади
16.Одновимірні масиви. Приклади
17.Пошук заданого елемента в послідовності. Алгоритм бінарного пошуку в упорядкованій послідовності. Приклади
18.Пошук мінімального/максимального елемента в послідовності. Приклади
19.Константне задання елементів масиву. Ініціалізація масивів. Генератор випадкових чисел. Приклади
20.Використання файлів для введення/виведення даних. Приклади
21.Двовимірні масиви. Багатовимірні масиви. Приклади
22.Створення тестових файлів для роботи з масивами. Приклади
23.Пошук мінімального та максимального елементів у двовимірному масиві. Приклад
24.Обробка елементів двовимірних масивів по рядках та по стовпцях. Приклади
25.Рядки та символьні масиви. Введення та виведення символьних та рядкових даних. Приклади
26.Спеціальні функції для введення та виведення значень символьних і рядкових величин. Приклади
27.Стандартні функції для роботи з рядками. Приклади
28.Лінійні пошукові алгоритми для рядкових величин. Приклади тестів
29.Алгоритми зміни довжин рядкових величин. Приклади
30.Порівняння рядкових величин. Посимвольне введення рядкових величин. Перетворення символьних значень у числові. Приклади
31.Масиви рядків. Приклади
32.Поняття динамічних змінних. Опис динамічних змінних або змінних-вказівників. Оператори для роботи з вказівниками. Приклади
33.Арифметичні дії з вказівниками. Порівняння вказівників. Приклади
34.Динамічне виділення пам’яті та вказівники. Приклади
35.Обробка елементів масиву за допомогою адресної арифметики. Приклади
36.Індексація елементів одновимірного масиву за допомогою вказівників. Приклади
37.Обробка елементів двовимірного масиву за допомогою вказівників. Приклади
38. Вказівники на символьні масиви. Ініціалізація вказівників. Масиви вказівників. Приклади

39.Поняття функції. Загальний вигляд опису функції. Локальні та глобальні параметри або змінні. Фактичні та формальні аргументи. Прототип функції. Приклади
40.Способи передачі і повернення параметрів у функцію: параметри-значення. Приклади
41.Способи передачі і повернення параметрів у функцію: параметри-вказівники. Приклади
42.Масиви як параметри функцій. Приклади
43.Додаткові можливості передавання параметрів у функції. Функція main(). Приклади
44.Поняття рекурсії. Принцип виконання рекурсивних функцій. Приклади
45.Вказівники на функції. Приклади
46.Сучасні технології програмування. Переваги та недоліки використання функцій
47.Поняття структури. Оголошення змінних описаної структури. Особливості оголошення структур. Звернення до полів структури. Приклади
48.Масиви структур. Структури в структурах. Структури з масивів. Приклади
49.Використання структур для роботи з функціями: передавання полів структур в функції. Приклади
50.Використання структур для роботи з функціями: передавання структури у функцію. Приклади
51.Використання вказівників на структури. Приклади
52.Бітові поля. Операції над бітовими полями. Приклади
53.Об’єднання (union). Приклади
54.Злічені типи даних. Операції над елементами злічених типів. Приклади
55.Використання операторів typedef та sizeof. Алгоритми обробки структурованих даних.
56.Поняття про системи числення. Зв’язок між системами числення з основою 2k.
57.Однорозрядний суматор. Алгоритм заміни операції віднімання на операцію додавання в однорозрядному суматорі.
58.Бітові оператори. Приклади.
59.Робота з файлами. Тип змінних FILE.
60. Використання можливостей середовища програмування для роботи з файлами.
61.Рекурсивний пошук у впорядкованому одновимірному масиві. Оцінка алгоритму. Приклади тестів
62.Прямий пошук підрядка у рядку. Оцінка алгоритму. Приклади тестів
63.КМП-пошук. Оцінка алгоритму. Приклади тестів
64.Основні поняття алгоритмів сортування. Класифікація методів сортування. Прямі методи сортування. Оцінка ефективності
65.Сортування вибором. Оцінка алгоритму. Приклади тестів
66.Сортування обміном. Оцінка алгоритму. Приклади тестів
67.Сортування вставленням. Оцінка алгоритму. Приклади тестів
68.Лінійні методи сортування: сортування підрахунком. Оцінка алгоритму. Приклади тестів

69. Лінійні методи сортування: цифрове сортування. Оцінка алгоритму. Приклади тестів
70. Директиви препроцесора. Створення власних заголовочних файлів. Приклади.

Зарахування результатів неформальної/інформальної освіти

Здобувачі вищої освіти має право на участь у неформальній/інформальній освіті.

У межах поточного контролю можуть визнаватися результати неформальної/інформальної освіти за умови наявності сертифікату або освітньої декларації про результати неформальної/інформальної освіти з питань, що відповідає тематиці курсу («Порядок визнання у Чернівецькому національному університеті імені Юрія Федьковича результатів навчання, здобутих шляхом неформальної та/або інформальної освіти», https://www.chnu.edu.ua/media/4g5fzssb/poriadok-vyznannia-rezultativ-navchannia-zdobutykh-shliakhom-neformalnoi-ta-abo-informalnoi-osvity.pdf).

Студентам можуть бути зараховані додаткові бали, отримані через неформальну освіту, до загальної суми балів, набраної з освітньої компоненти, за умови, що результати з проблеми, за якою відбувалося навчання, відповідають тематиці курсу.

Рекомендована література
Основна

1. Караванова Т.П. Методика розв'язування алгоритмічних задач. Основи алгоритмізації та програмування: Навчально-методичний посібник для вчителів / Т.П.Караванова. – Кам'янець-Подільський: Аксіома, 2013, 460 с.
2. Караванова Т.П. Методика розв'язування алгоритмічних задач. Побудова алгоритмів: Навчально-методичний посібник для вчителів / Т.П.Караванова. – Кам'янець-Подільський: Аксіома, 2014, 344 с.
3. Вінник В.Ю. Алгоритмічні мови та основи програмування: мова С. Житомир: ЖДТУ. 2007, 328 с. URL: http://programming.in.ua/programming/c-language/327-vinnyk-algorithmic-languages-and-the-basics-of-programming-c-language.html
4. Браян. В., Керніган, Деніс М. Річі Мова програмування С, друге видання. URL: http://programming.in.ua/programming/c-language/227-book-programming-c-kernighan.html
5. Brian W. Kernighan, Dennis Ritchie The C Programming Language, 2nd Edition, Publisher(s): Pearson, 1988, 272 p.
6. Мова програмування С/С++. Практикум: навчальний посібник / О. В. Прокопенко, М. О. Попов, Г. Л. Чумак. – К.: Київський національний університет імені Тараса Шевченка, 2024. – 375 с. URL:

https://iht.knu.ua/wp-content/uploads/2024/06/C_C-%D0%9F%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%BA%D1%83%D0%BC_2024.pdf

7. Програмування. Частина 2. Навчальний посібник / Укл.: Івасюк Г.П., Фратавчан Т.М., Івасюк Р.В. Чернівці: Чернівецький нац. ун-т, 2025. 164 с.

Допоміжна

1. Herbert Schildt C++: A Beginner's Guide, Second Edition, Publisher: McGraw-Hill Prof Med/Tech, 2003, 567 p.
2. Herbert Schildt Herb Schildt's C++ Programming Cookbook, Publisher: McGraw Hill Professional, 2008, 509 p.
3. Robert Sedgewick Algorithms in C++, Parts 1-4: Fundamentals, Data Structure, Sorting, Searching, Third Edition 3rd Edition. – Publisher: Addison-Wesley Professional; 3rd edition (July 13, 1998). — 738 pages.
4. Stephen Prata C Primer Plus — Publisher: Addison-Wesley Professional, 2013. — 1037 pages.
5. Караванова Т.П. Інформатика: основи алгоритмізації та програмув.: 777 задач з рек. та прикл.: Навч. посіб. для 8-9 кл. із поглибленим вивч. інф-ки / За заг. ред. М.З.Згуровського — К.: Генеза, 2012. — 286 с.: іл. — Бібліограф.: 286 с.
6. Караванова Т.П. Інформатика: методи побудови алгоритмів та їх аналіз: необчислювальні алгоритми: Навч. посіб. для 9-10 кл. із поглибл. вивч. інф-ки – К.: Генеза. – 2007.– 216 с.:іл. — Бібліограф.: 212 с.

Інформаційні ресурси

1. https://moodle.chnu.edu.ua/enrol/index.php?id=2535 – розміщення курсу на платформі https://moodle.chnu.edu.ua/.
2. https://moodle.chnu.edu.ua/course/view.php?id=2536 – розміщення курсу на платформі https://moodle.chnu.edu.ua/.

Політика академічної доброчесності

Дотримання політики щодо академічної доброчесності учасниками освітнього процесу при вивченні навчальної дисципліни регламентовано такими документами:

– «Етичний кодекс Чернівецького національного університету імені Юрія Федьковича» https://www.chnu.edu.ua/universytet/normatyvni-dokumenty/etychnyi-kodeks-chernivetskoho-natsionalnoho-universytetu-imeni-yuriia-fedkovycha/

– «Положенням про виявлення та запобігання академічного плагіату у Чернівецькому національному університету імені Юрія Федьковича» https://www.chnu.edu.ua/universytet/normatyvni-dokumenty/polozhennia-pro-vyiavlennia-ta-zapobihannia-akademichnomu-plahiatu/