Факультет інформаційних технологій
Кафедра цифрової економіки та системного аналізу

СИЛАБУС (SYLLABUS)
Дисципліна «Алгоритмізація та програмування» / Algorithmization and programming

ІНФОРМАЦІЯ ПРО ВИКЛАДАЧА

Викладач	Міценко Сергій Анатолійович
Науковий ступінь	Кандидат технічних наук
Вчене звання	Доцент
Посада	Доцент кафедри цифрової економіки та системного аналізу
Адреса кафедри	м.Київ, вул. Кіото 19, каб. Б-517, Б-519
E-mail	desa@knute.edu.ua
Консультації	Відповідно до графіку індивідуальних консультацій на сайті кафедри

ПОЛІТИКА АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ

https://knute.edu.ua/file/MzEyMQ==/c12a9f74e87d9154696ca0f761da2e5c.pdf

Дотримання академічної доброчесності передбачає:

- самостійне виконання навчальних завдань, завдань поточного та підсумкового контролю результатів навчання (для осіб з особливими освітніми потребами ця вимога застосовується з урахуванням їхніх індивідуальних потреб і можливостей);
- посилання на джерела інформації у разі використання не авторських ідей, розробок, тверджень, відомостей і т.п.;
- дотримання норм законодавства про авторське право і суміжні права;
- надання достовірної інформації про результати власної наукової діяльності, використані методики досліджень і джерела інформації.

Порушенням академічної доброчесності вважається:

- академічний плагіат – оприлюднення (частково або повністю) наукових (творчих) результатів, отриманих іншими особами, як результатів власного дослідження (творчості) та/або відтворення опублікованих текстів (оприлюднених творів мистецтва) інших авторів без зазначення авторства;
- самоплагіат – оприлюднення (частково або повністю) власних раніше опублікованих наукових результатів як нових наукових результатів;
- фабрикація – вигадування даних чи фактів, що використовуються в наукових дослідженнях;
- фальсифікація – свідома зміна чи модифікація вже наявних даних, що стосуються наукових досліджень.

За порушення академічної доброчесності здобувачі освіти можуть бути притягнені до академічної відповідальності:

- повторне проходження оцінювання (модульний контроль, іспит, залік тощо);
- повторне проходження відповідного освітнього компонента освітньо-професійної програми;
- відрахування з Університету;
- позбавлення наданих університетом пільг;

- відмова у присудженні відповідного ступеня вищої освіти;

ПОЛІТИКА ЩОДО ВІДВІДУВАННЯ ЗАНЯТЬ

- відвідування занять є обов'язковим;
- Студент, який пропустив практичне заняття, самостійно вивчає матеріал (при виникненні питань може звертатися за консультацією згідно розкладу консультацій викладачів оприлюдненого на сайті кафедри) за наведеними джерелами, виконує завдання і здає його викладачу.
- за об'єктивних причин (наприклад, хвороба, міжнародне стажування та ін.) навчання може відбуватись в он-лайн формі за погодженням із викладачем дисципліни.

ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Назва дисципліни / тип дисципліни	Алгоритмізація та програмування / обов'язкова
Навчальний рік	2021-2022, 2022-2023
Факультет	Факультет інформаційних технологій
Курс	1,2
Семестр	2, 3
Освітній ступінь	Молодший бакалавр
Галузь знань	12 «Інформаційні технології»
Спеціельність	122 «Комп'ютерні науки»
Загальна характеристика	Кількість годин –360 Кількість кредитів – 12 **Види занять:** лекції, лабораторні, самостійна робота. **Співвідношення аудиторних годин і годин самостійної роботи** - 186/174 **Мова викладання** – українська **Форма викладання** – очна
Підсумковий контроль	Екзамен
Програмне забезпечення	C, C++
Обладнання	Проектор, комп'ютерна техніка із встановленим програмним забезпеченням та доступом до мережі Інтернет.
Необхідні попередні дисципліни	Шкільний курс з дисципліни «Інформатика»; «Дискретна математика».
Методика вивчення	Методика вивчення дисципліни полягає у набутті студентами знань теоретичного і практично-прикладного характеру під час лекцій, лабораторних занять, самостійної роботи та вивчення першоджерел і навчально-методичної літератури.
Мета і завлання	**Метою** вивчення дисципліни "Алгоритмізація та програмування" є формування у студентів навичок оволодіння технологіями обробки простих та структурованих даних, опанування технологій структурного, модульного та об'єктно-орієнтованого програмування на базі мови програмування C++. Завданням вивчення дисципліни є теоретична та практична підготов-ка студентів з таких питань: • парадигми програмування (структурного, модульного, об'єктно-орієнтованого) та засоби сучасних мов програмування для реалізації різних концепцій; • засоби мов програмування для реалізація розгалужених та цикліч-

	них алгоритмів; • використання функцій та окремих модулів користувача; • реалізація класичних інформаційних структур (списків, дерев) з використанням статичного та динамічного розподілу пам'яті; • базові технології проектування процесів пошуку та побудови впорядкованих даних у лінійних списках та деревовидних структурах; • використання класів для реалізації парадигми об'єктно-орієнтованого програмування; • засоби створення ієрархічної об'єктної структури з використанням базового об'єкта, подальше розширення його властивостей з використанням статичних та віртуальних методів; • сучасні інструментальні засоби для створення прикладного програмного забезпечення. Автоматизація програмування: технологія RAD (Rappid Application Development); • особливості програмування в операційних системах родини Windows. Система управління повідомленнями. Структура програми для Windows.
Місце дисципліни в освітньо-професійній програмі	
Загальні компетентності	ЗК 2 Здатність застосовувати знання у практичних ситуаціях. ЗК 3 Здатність до розуміння предметної області та професійної діяльності. ЗК 4 Здатність спілкуватися державною мовою як усно, так і письмово в термінології предметної області.
Фахові компетентності (результати навчання)	СК 3 Здатність до логічного мислення, побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв'язності та нерозв'язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем. СК 4 Здатність використовувати сучасні методи математичного моделювання об'єктів, процесів і явищ, розробляти моделі й алгоритми чисельного розв'язування задач математичного моделювання, враховувати похибки наближеного чисельного розв'язування професійних задач. СК 6 Здатність застосовувати методології, технології та інструментальні засоби для розробки програмних систем, продуктів і сервісів інформаційних технологій. СК 8 Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.
Програмні результати навчання	ПР 1 Застосовувати знання основних форм і законів абстрактно-логічного мислення, форм і мето-дів вилучення, аналізу, обробки та синтезу ін-формації в предметній області комп'ютерних наук. ПР 4 Використовувати високорівневі мови програмування, обчислювальні методи і алгоритми для розробки програмних засобів обробки даних. ПР 5 Проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій. ПР 8 Використовувати інструментальні засоби проектування

	концептуальних, логічних та фізичних моделей баз даних, створювати бази даних, розробляти та оптимізувати запити до них, у тому числі із застосуванням мов програмування.

ТЕМАТИКА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
РОЗДІЛ 1. ОСНОВИ ПРОГРАМУВАННЯ МОВОЮ С

Тема 1.1. Введення у дисципліну «Алгоритмізація та програмування»

Вступ. Мета та завдання дисципліни, її місце у освітньому процесі. Парадигма та основні ідеї, покладені у сучасні алгоритмічні мови програмування. Основні ресурси, спільноти користувачів і розробників. Мови програмування: призначення, основні особливості. Підготовка персонального домашнього комп'ютера до роботи з IDE (PyCharm, VisualStudio). Основні її можливості та вбудовані робочі інструменти.

Тема 1.2. Огляд можливостей мови програмування С++. Знайомство з середовищем розробки програм

Класифікація мов програмування та напрямки їх розвитку. Введення до мови програмування С++, її місце серед інших мов програмування. Види програмного забезпечення для розробки програмних продуктів мовою С++. Знайомство з інструментальними засобами Microsoft Visual Studio. Автоматизація програмування: технологія RAD (Rappid Application Development). Поняття проекту. Огляд майстрів для створення проектів різних видів у Microsoft Visual C++. Приклад створення першого проекту для консольної прикладної програми. Основні структурні компоненти програми С++, функція **main()**. Поняття програмного модуля. Приклади програм. Огляд основних етапів створення програмних продуктів. Компіляція та запуск програм на виконання. Етап налагодження програми. Основні види помилок. Виправлення помилок. Інтерактивні засоби Microsoft Visual C++ для налагодження програм: покрокове виконання програми, перегляд значень змінних, встановлення точок зупинок.

Тема 1.3. Основні поняття мови програмування. Базові типи, константи, змінні, операції, вирази

Алфавіт мови програмування. Поняття лексеми. Лексична структура мови програмування. Типи лексем: константи, літерали, ідентифікатори, ключові слова, операції, розділові знаки. Числові константи у різних системах числення, літерні константи, символьні константи, використання спеціальних символів для визначення недрукованих символів. Призначення та використання коментарів у програмі. Поняття синтаксису та семантики мови програмування. Використання метамови для описання синтаксичних конструкцій. Форми Бекуса-Наура. Синтаксичні діаграми. Концепція типу даних. Базові типи. Кваліфікатори типів. Співвідношення між розмірами даних базових типів. Операція **sizeof.** Поняття змінної. Оголошення та ініціалізація змінних. Поняття оператора. Простий та складений оператор. Порожній оператор. Оператор присвоєння. Поняття виразу. Операнди та операції. Унарні та бінарні операції. Арифметичні операції. Операції відношення. Логічні операції. Побітові операції. Операції інкременту та декременту. Приклади використання операцій. Пріоритет виконання операцій у виразах. Перетворення типів у виразах.

Тема 1.4. Керування виконанням програми

Поняття розгалужених та циклічних алгоритмів. Приклади задач, розв'язання яких зводиться до застосування розгалужених та циклічних алгоритмів. Рекурентні співвідношення, циклічні ітераційні процеси, табуляція функцій, наближене обчислення функцій. Реалізація розгалужених алгоритмів за допомогою операторів: **if ... else ; if ... else if ... else; switch.** Порівняння операторів розгалуження.Реалізація циклічних процесів за допомогою операторів: **while ; for ; do while.** Порівняння операторів циклів. Особливості операторів циклів з лічильником та з умовою. Відмінність між синтаксисом та семантикою циклічних конструкцій з пост- та передумовою. Продовження та достроковий вихід із циклів: оператори **breake** та **continue.** Оператор переходу **goto.** Правила побудови вкладених операторів циклів та розгалуження.

Тема 1.5. Функції користувача та класи пам'яті

Парадигма структурного та модульного програмування. Загальні відомості про функції користувача. Синтаксис для оголошення та визначення функцій користувача. Формальні та фактичні параметри

функцій. Виклик функцій. Підстановка фактичних параметрів за значеннями та посиланнями: основна різниця між двома способами передачі аргументів. Приклади створення функцій користувача та їх виклику.Функції зі змінним числом аргументів. Поняття про рекурсію. Труднощі, що виникають під час реалізації рекурсії. Приклади класичних рекурсивних алгоритмів. Функція Акермана. Задача про «Ханойські вежі». Рекурсивні криві. Поняття області видимості та "часу життя" програмних об'єктів. Зовнішні змінні та функції. Локальні імена та локальні змінні, автоматична пам'ять. Оголошення та ініціалізація статичних змінних. Регістрові змінні. Змінні класу **volatile.** Передача аргументів у функцію **main().**

Тема 1.6. Вказівники та адресна арифметика

Поняття та оголошення вказівника. Використання модифікаторів в оголошеннях вказівників. Вказівники типу **void.** Вказівники та адреси. Операції зі вказівниками. Унарні операції одержання адреси та розкриття посилання вказівника. Динамічний розподіл пам'яті. Функції **malloc()** та **free().** Вказівники та аргументи функцій. Використання вказівників для передачі фактичних аргументів за посиланнями. Вказівники на функції.

Тема 1.7. Складені типи даних: масиви

Концепція складених типів даних. Поняття масиву, індексу, елемента. Оголошення та ініціалізація одновимірних масивів. Методи доступу до елементів масивів. Індексний доступ до елементів масиву. Використання вказівників для посилання на елементи масиву. Багатовимірні масиви. Розміщення елементів багатовимірних масивів у пам'яті. Вказівники на багатовимірні масиви. Розрізи багатовимірних масивів. Доступ до рядків та стовпців двовимірних масивів.Масиви символів. Використання вказівників для доступу до символів рядків. Масиви вказівників. Ініціалізація масивів вказівників. Вказівники на вказівники. Правила інтерпретації складних оголошень. Приклади складених оголошень.

Тема 1.8. Складені типи даних: структури, об'єднання, перелічення

Концепція складеного типу даних. Основні відомості про структури. Тег та елементи структури. Приклади структур. Описання та ініціалізація структури. Використання декларації **typedef** для створення нових типів. Операції зі структурами. Доступ до елементів структури за допомогою операції ".". Використання вказівників для одержання доступу до елементів структури: операція "->". Масиви структур. Поняття запису. Рекурсивні оголошення структур з використанням вказівників. Приклади рекурсивних структур: лінійний список, бінарне дерево.Об'єднання **(union)** як спосіб збереження даних типів даних в одній області пам'яті. Приклади об'єднань. Створення списків-переліків (множин). Декларація **enum.** Зміна нумерації елементів множин.

Тема 1.9. Алгоритми на масивах

Використання циклів для обробки масивів. Алгоритм вибору простих чисел за допомогою масиву. Приклади цікавих алгоритмів на масивах за допомогою масивів чисел та масивів структур.

Тема 1.10. Директиви препроцесора, макропідстановка, умовна компіляція

Призначення C++ пре процесора. Використання головних файлів (*.h). Поняття та використання макропідстановки. Синтаксис та використання директиви **#include.** Синтаксис та використання директиви **#define.** Синтаксис та використання директиви **#undef.** Умовна компіляція. Директива **#if.**

Тема 1.11. Функції стандартної бібліотеки

Головний файл **string.h.** Огляд функцій стандартної бібліотеки для роботи з рядками: функції типу **char *** та **void** *. Копіювання рядків. Порівняння рядків. Конкатенація рядків. Пошук символів та лексем у рядках. Головний файл **stdio.h.** Огляд функцій стандартної бібліотеки для організації введення/виведення. Функції **getchar()** та **putchar()-** Приклади функцій користувача для введення/виведення рядків. Форматне виведення: функція **printf().** Форматне введення: функція **scanf()-**Функції для одержання доступу до файлів: відкриття файлів, закриття файлів, позиціоноване введення/виведення. Головний файл **ctype.h.** Огляд функцій для перевірки класів символів. Головний файл **math.h.** Огляд математичних функцій. Головний файл **stdlib.h.** Огляд функцій загального призначення. Головний файл **time.h.** Огляд функцій дати та часу.

Тема 1.12. Принципи аналізу алгоритмів

Розробка та емпіричний аналіз. Ріст функцій. Q – нотація. Приклади алгоритмічного аналізу. Гарантії передбачення та обмеження.

РОЗДІЛ 2. СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ

Тема 2.1. Огляд класичних алгоритмів для абстрактних типів даних

Алгоритми та структури. Застосування алгоритмів для реалізації практичних завдань в різних сферах діяльності людини. Задача зв'язності. Алгоритми об'єднання-пошуку. Лінійні списки. Алгоритми сортування. Пошук даних та обробка строк. Геометричні алгоритми та графи. . Приклад реалізації алгоритму зв'язності. Оцінка алгоритмів.

Тема 2.2. Організація лінійних списків та їх обробка

Поняття лінійного списку. Методи організації та збереження списків. Використання масивів для збереження елементів списків. Спосіб зв'язаного збереження списків з використанням рекурсивно визначених структур. Огляд операцій, що виникають під час обробки списків. Реалізація операцій для послідовного збереження списків. Реалізація операцій для зв'язаного збереження списків. Двозв'язані списки. Реалізація основних операцій для двозв'язаних списків. Циклічні списки.

Тема 2.3. Стеки

Стек (структура LIFO - Last In First Out) як спеціальний тип лінійного списку. Поняття вершини стека. Організація та збереження елементів стека. Основні операції для роботи зі стеками. Реалізація операції додавання елемента у вершину стека. Реалізація операції видалення елемента з вершини стека. Перегляд елементів без видалення їх зі стека.

Тема 2.4. Черги

Черга (структура FIFO - First In First Out) як спеціальний тип лінійного списку. Поняття початку та кінця черги. Основні операції для роботи з чергами. Реалізація додавання елемента в чергу. Реалізація видалення елемента з черги. Циклічні черги.

Тема 2.5. Деревовидні структури

Основні поняття та визначення: дерево, корінь, вузол, листок, твірний вузол, дочірній вузол, рівень вузла, праве та ліве під дерево, глибина дерева. Способи збереження дерев. Алгоритми для проходження дерева: прямий, симетричний, обернений. Основні операції для роботи з деревами. Реалізація проходження дерева та деяких операцій. Поняття бінарного дерева. Особливості збереження бінарних дерев.

РОЗДІЛ 3. C++ ТА ОБ'ЄКТНООРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Тема 3.1. Особливості програмування у C++

Коментарі C++. Нові ключові слова. Використання оголошень в операторах. Операція для розширення області видимості змінних та функцій. Прототипи функцій. Перевантаження функцій. Приклади перевантаження функцій. Присвоєння значень формальних параметрів за замовченням. Вказівники на константи та на змінні типу **void.** Динамічний розподіл пам'яті. Операції **new, delete.**Складені типи даних: **union, enum.**Потокове введення/виведення. Бібліотека **iostream.**

Тема 3.2. Основні поняття та властивості об'єктно-орієнтованого програмування (ООП). Класи C++

Концепція об'єктно орієнтованого програмування, її відмінність від концепції структурного програмування. Компоненти об'єктно орієнтованої парадигми: об'єкт, повідомлення, клас, властивість, метод. Основні властивості ООП: абстракція, інкапсуляція, наслідування, поліморфізм. Етапи розробки об'єктно орієнтованих прикладних програм. Інтерпретація основних понять ООП в термінах C++. Клас як розширення структурного типу. Синтаксис описання класу. Розділи **private, protected, public.** Інкапсуляція даних та функцій. Призначення та створення конструкторів класу. Параметри конструктора. Призначення та створення деструктора класу. Приклади опису класів.Об'єкт як екземпляр деякого класу. Створення об'єкта. Інтерпретація властивостей та методів об'єкта в термінах полів даних та реалізацій функцій-членів деякого класу. Доступ до властивостей та методів об'єкта. Поняття повідомлення. Передача повідомлення об'єкта. Використання вказівника **this** для одержання доступу до поточного об'єкта. Опис класу в двох файлах: інтерфейс класу (файл *.h), визначення функцій класу (файл *.cpp). Додаткові властивості ООП: параметризовані типи, колекції та множини.

Тема 3.3. Структура та ієрархія класів, наслідування, поліморфізм

Поняття простого наслідування, базового класу, похідного класу, ієрархії класів. Синтаксис визначення похідного класу на основі базового. Відкриті та закриті похідні класи. Правила доступу до полів даних та функцій базових класів. Правила доступу до похідних класів та їх об'єктів. Створення ієрархії класів для простого наслідування. Приклади простого наслідування. Поняття поліморфізму та інтерпретація його в термінах віртуальних функцій С++. Зв'язування методів з об'єктами на етапі виконання програми - "пізнє зв'язування". Зв'язування методів з об'єктами на етапі компіляції - "раннє зв'язування". Відмінності між двома способами зв'язування. Поняття поліморфного кластера. Реалізація віртуальних функцій. Приклади віртуальних функцій. Поняття множинного наслідування. Подання множинного наслідування у вигляді графів. Приклади множинного наслідування. Конфлікти, що можуть виникати під час множинного наслідування. Розв'язання конфліктів.

Тема 3.4. Перевантаження операторів у С++. Шаблони та шаблонні функції.

Способи перевантаження операцій у С++. Обмеження, що виникають під час перевантаження операцій. Використання класів для перевантаження операцій. Приклад класу для перевантаження операцій " + "," * ". Перевантаження операцій new та delete. Перевантаження оператора присвоєння. Шаблони та шаблонні функції.

Тема 3.5. Створення власних класів на основі структур для реалізації списків, стеків та черг.

Створення конструкторів та деструкторів класів. Застосування шаблонів. Створення функцій – членів класу, функцій друзів та помічників класу. Функції обробки виключень. Тестування створених класів.

Тема 3.6. Створення власних класів та рекурсивних функцій у віконних додатках основаних на проектах API

Створення простого проекту віконного додатку Application Programming Interface (API). Створення власного класу з рекурсивною функцією малювання. Створення вікна діалогу передачі параметрів функції. Ініціалізація елементів управління та функція отримання параметрів. Створення елементу меню вікна та функції обробки повідомлення.

РОЗДІЛ 4. АЛГОРИТМИ ВПОРЯДКУВАННЯ ТА ПОШУКУ

Тема 4.1. Бінарні дерева та вирази

Подання арифметичних виразів у вигляді бінарного дерева. Приклади бінарних дерев. Різні записи виразів, що одержуються в результаті різних способів проходження дерева: польський запис (прямий та обернений), повний запис (з використанням дужок). Побудова рекурсивного алгоритму та його реалізація для введення виразу у повному записі. Побудова рекурсивного алгоритму та його реалізація для виконання аналітичного диференціювання виразу. Побудова рекурсивного алгоритму та його реалізація для введення виразу у звичайному математичному записі з урахуванням пріоритетів операцій.

Тема 4.2. Прості алгоритми впорядкування масивів та списків

Постановка задачі впорядкування. Прості алгоритми впорядкування. Метод попарного порівняння (алгоритм "бульбашки"). Впорядкування методом вставок. Впорядкування методом вибору максимального чи мінімального елемента. Оцінка складності та швидкості алгоритмів. Порівняння простих методів упорядкування. Сортування пов’язаних списків.

Тема 4.3. Алгоритми швидкого впорядкування

Модифікація алгоритмів впорядкування. Алгоритми швидкого впорядкування. Впорядкування методом розподілу. Бітове впорядкування. Бінарне впорядкування. Метод Шелла. Характеристики методів впорядкування. Злиття двох списків. Впорядкування методом злиття.

Тема 4.4. Черги по пріоритетах та пірамідальне сортування

Пірамідальна структура даних сортування. Абстрактний тип даних черги по пріоритетах. Алгоритми дерев сортування. Черга по пріоритетах для індексних елементів.

Тема 4.5. Задачі пошуку

Постановка задачі пошуку. Послідовний пошук. Формулювання алгоритму для послідовного пошуку. Реалізація алгоритму послідовного пошуку. Оцінка складності та швидкості алгоритму послідовного пошуку. Бінарний пошук. Формулювання та реалізація алгоритму для бінарного пошуку. М-блочний пошук.

Тема 4.6. Пошук методом хешування

Методи обчислення адреси. Поняття функції хешування. Масив хешування. Формулювання та реалізація алгоритму пошуку з використанням хешування. Виникнення колізії. Алгоритми для обробки колізій. Порівняння різних методів пошуку. Основні переваги методів з використанням адрес.

Тема 4.7. Пошук та впорядкування у деревовидних структурах

Поняття впорядкованого дерева. Побудова впорядкованого дерева. Пошук інформації у впорядкованому дереві. Алгоритми для додавання вузла у впорядковане дерево. Алгоритми для видалення вузла із упорядкованого дерева. Поняття збалансованого дерева. Балансування дерева. Перебалансування після видалення. Задачі про розміщення ферзів та пошук підмножини. Перебір з поверненням та дерева пошуку. Рекурсивне та не рекурсивне розв'язання задачі про рюкзак.

Тема 4.8. Зовнішній пошук у деревовидних структурах

Постановка задачі пошуку для великих масивів даних. Індексований послідовний доступ. Структура та побудова B- дерева. Алгоритми для додавання вузла у B-дерево, розділення вузлів. Алгоритми для пошуку даних у B-деревах.

Перелік навчальних робіт студентів та оцінки їх у балах з дисципліни «Алгоритмізація та програмування»

Види робіт	К-сть балів
Лабораторне заняття №1. Ознайомлення з середовищем розробки програм	2
Лабораторне заняття №2. Програма з найпростішою структурою	2
Лабораторне заняття №3. Використання керуючих конструкцій	2
Лабораторне заняття №4. Використання функцій, визначених користувачем	2
Лабораторне заняття №5. Вказівники та адресна арифметика	2
Лабораторне заняття №6. Складені типи даних масиви	4
Лабораторне заняття №7. Складені типи даних: структури, об'єднання, перерахування	4
Лабораторне заняття №8. Алгоритми на масивах.	4
Лабораторне заняття №9. Директиви препроцесора, макропідстановка, умовна компіляція	4
Лабораторне заняття №10. Функції стандартної бібліотеки, обробка символьної інформації, функції введення/виведення	4
Лабораторне заняття №11. Принципи аналізу алгоритмів	4
Лабораторне заняття №12. Організація списків та їх обробка	4
Лабораторне заняття №13. Стеки	4
Лабораторне заняття №14. Черги	4
Лабораторне заняття №15. Деревовидні структури	4
Лабораторне заняття №1. Основні поняття та властивості об`єктно-орієнтованого програмування (ООП). Класи C++	3
Лабораторне заняття №2. Наслідування та поліморфізм	3
Лабораторне заняття №3. Перевантаження операторів C++. Шаблони та шаблонні функції	4
Лабораторне заняття №4. Створення власних класів на основі структур для реалізації списків, стеків та черг.	4
Лабораторне заняття №5. Створення власних класів у віконних додатках основаних на проектах API	4

Лабораторне заняття №6. Бінарні дерева та вирази	4
Лабораторне заняття №7. Прості алгоритми впорядкування масивів та списків	4
Лабораторне заняття №8. Алгоритми швидкого впорядкування	4
Лабораторне заняття №9. Черги по пріоритетах та пірамідальне сортування	4
Лабораторне заняття №10. Задачі пошуку	4
Лабораторне заняття №11. Пошук методом хешування.	4
Лабораторне заняття №12. Пошук та впорядкування у деревовидних структурах	4
Лабораторне заняття №13. Зовнішній пошук у деревовидних структурах	4
Модульний контроль	40
Виконання індивідуального завдання (СР)	60
Разом: Аудиторна робота	**140**
Самостійна робота (СР)	**60**
Всього:	**200**

КОНТРОЛЬ ТА КРИТЕРІЇ ОЦІНЮВАННЯ ЗНАНЬ СТУДЕНТІВ

При вивченні дисципліни використовуються наступні форми контролю знань студентів: поточний; модульний; підсумковий.

Поточний контроль передбачає перевірку теоретичних питань, самостійної роботи, практичних робіт та усне опитування по кожній практичній роботі. По даному виду контролю оцінювання знань здійснюється у відповідності до бального розподілу наведеного в попередній таблиці.

Підсумкова модульна оцінка за семестр є сумою оцінок, отриманих студентом за виконання лабораторних завдань та двох оцінок модульного контролю. Максимальна модульна оцінка становить 100 балів.

Формою підсумкового контролю є екзамен. Максимальна екзаменаційна оцінка становить 100 балів.

Підсумкова оцінка з дисципліни обчислюється як середнє арифметичне підсумкової модульної та екзаменаційної оцінки.

СПИСОК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

Основний:

1. Бєлов Ю. Вступ до програмування мовою С++. Організація обчислень: навч. посіб. / Ю.Бєлов, Т.Карнаух, Ю.Коваль, А.Ставровський. – К. : Видавничо-поліграфічний центр "Київський університет", 2012. – 175 с. с.: іл. (укр.)
2. Л. М. Клакович, С. М. Левицька. Теорія алгоритмів: Навчальний посібник. — Друге видання, доповнене. — Львів :Видавничий центр ЛНУ ім. Івана Франка, 2015. — 161 с.
3. Зацерковний В. І. Алгоритмізація та програмування: навчальний посібник / В. І. Зацерковний, В. І. Гур'єв, І. В. Фірсова. – Ніжин: НДУ ім. М. Гоголя, 2013. – 302 с.
4. Матвієнко М.П. Теорія алгоритмів: підручник / Матвієнко М.П. –Київ: Ліра-К, 2017. – 344 с.
5. Sedgewick R., Algorithms in C++ :Fundamentals, data structurs, sorting, serching/ Princeton University 2011.– 1056 с.