Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

A blue and white logo

Description automatically generated

**Звіт**

**про виконання лабораторних та практичних робіт блоку № 6**

На тему:  «Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур.»

***з дисципліни:*** «Основи програмування»

до:

ВНС Лабораторної Роботи № 10

Алготестер Лабораторної Роботи № 5

Алготестер Лабораторної Роботи № 7-8

Практичних Робіт до блоку № 6

**Виконав:**

Студент групи ШІ-11

Станько Олег Ігорович

Львів 2024

Тема: «Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур.»

Мета: зрозуміти основи побудови та використання таких структур даних, як черга, стек, списки та дерева, освоїти алгоритми, які дозволяють ефективно маніпулювати та обробляти ці структури.

**Теоретичні відомості**

1. Основи Динамічних Структур Даних
2. Стек
3. Черга
4. Зв'язні Списки
5. Дерева
6. Алгоритми Обробки Динамічних Структур

**індвидуальний план опрацювання теорії**

1. Основи Динамічних Структур Даних  
   [Data Structures Tutorial - GeeksforGeeks](https://www.geeksforgeeks.org/data-structures/)

Ознайомився з виділенням пам'яті для структур даних (stack і heap)

Витрачено 1 годину

1. **Стек**

[Stack Data Structure - GeeksforGeeks](https://www.geeksforgeeks.org/stack-data-structure/)

Вивчив принцип роботи стеку, його застосування та основні операції (push, pop, top).

Витрачено 1 годину.

1. **Черга**

[Queue Data Structure - GeeksforGeeks](https://www.geeksforgeeks.org/queue-data-structure/)

Ознайомився з принципами роботи черги, її застосуванням та основними операціями (enqueue, dequeue, front).

Витрачено 1 годину.

1. **Зв'язні Списки**

[Linked List Data Structure - GeeksforGeeks](https://www.geeksforgeeks.org/linked-list-data-structure/)  
Вивчив основи зв'язних списків, різновиди (однозв'язний, двозв'язний) та основні операції (вставка, видалення, пошук).

Витрачено 2 години.

1. **Дерева**

[Linked List Data Structure - GeeksforGeeks](https://www.geeksforgeeks.org/linked-list-data-structure/)

Ознайомився з деревами, їх видами (бінарні дерева, дерева пошуку) та основними операціями (вставка, видалення, пошук).

Витрачено 2 години.

1. **Алгоритми Обробки Динамічних Структур**

[Algorithms Tutorial - GeeksforGeeks](https://www.geeksforgeeks.org/fundamentals-of-algorithms/)  
Вивчив алгоритми обробки динамічних структур даних, зокрема сортування та пошук.

2 години.

**Виконання роботи:**

**1. Опрацювання завдання та вимог до програм.**

**VNS LAB 10**

Записи в лінійному списку містять ключове поле типу \*char (рядок символів). Сформувати двонаправлений список. Знищити елемент із заданим ключем. Додати К елементів у початок списку.

**Algotester Lab 5**

*Обмеження: 1 сек., 256 МіБ*

В пустелі існує незвичайна печера, яка є двохвимірною. Її висота це N, ширина - M.

Всередині печери є пустота, пісок та каміння. Пустота позначається буквою О , пісок S і каміння X;

Одного дня стався землетрус і весь пісок посипався вниз. Він падає на найнижчу клітинку з пустотою, але він не може пролетіти через каміння.

Ваше завдання сказати як буде виглядати печера після землетрусу.

**Вхідні дані**

У першому рядку 2 цілих числа N та M - висота та ширина печери

У N наступних рядках стрічка rowi яка складається з N цифер - i-й рядок матриці, яка відображає стан печери до землетрусу.

**Вихідні дані**

N рядків, які складаються з стрічки розміром M - стан печери після землетрусу.

**Algotester Lab 7-8**

Ваше завдання - власноруч реалізувати структуру даних "Двійкове дерево пошуку".  
Ви отримаєте Q запитів, кожен запит буде починатися зі слова-ідентифікатора, після якого йдуть його параметри.  
  
Вам будуть поступати запити такого типу:

* **Вставка**:  
  Ідентифікатор - insert  
  Ви отримуєте ціле число value - число, яке треба вставити в дерево.
* **Пошук**:  
  Ідентифікатор - contains  
  Ви отримуєте ціле число value - число, наявність якого у дереві необхідно перевірити.  
  Якщо valuevalue наявне в дереві - ви виводите Yes, у іншому випадку No.
* **Визначення розміру**:  
  Ідентифікатор - size  
  Ви не отримуєте аргументів.  
  Ви виводите кількість елементів у дереві.
* **Вивід дерева на екран**  
  Ідентифікатор - printprint  
  Ви не отримуєте аргументів.  
  Ви виводите усі елементи дерева через пробіл.  
  Реалізувати використовуючи перегрузку оператора <<

**Вхідні дані**

Ціле число Q - кількість запитів.  
У наступних рядках Q запитів у зазначеному в умові форматі.

**Вихідні дані**

Відповіді на запити у зазначеному в умові форматі.

**Для того щоб отримати**50%**балів за лабораторну достатньо написати свою структуру.  
Для отримання**100%**балів ця структура має бути написана як**[**шаблон класу**](https://cplusplus.com/doc/oldtutorial/templates/)**, у якості параметру використати**int**. Використовувати STL заборонено.**

**Class Practice Work**

***Реалізувати метод реверсу списку:*** Node\* reverse(Node \*head);

*Умови задачі:*

-       використовувати цілочисельні значення в списку;

-       реалізувати метод реверсу;

-       реалізувати допоміжний метод виведення вхідного і обернутого списків;

bool compare(Node \*h1, Node \*h2);

*Умови задачі:*

-       використовувати цілочисельні значення в списку;

-       реалізувати функцію, яка ітеративно проходиться по обох списках і порівнює дані в кожному вузлі;

-       якщо виявлено невідповідність даних або якщо довжина списків різна (один список закінчується раніше іншого), функція повертає ***false***.

Node\* add(Node \*n1, Node \*n2);

*Умови задачі:*

-       використовувати цифри від 0 до 9 для значень у списку;

-       реалізувати функцію, яка обчислює суму двох чисел, які збережено в списку; молодший розряд числа записано в голові списка (напр. 379  ⟹  9→7→3);

-       функція повертає новий список, передані в функцію списки не модифікуються.

TreeNode \*create\_mirror\_flip(TreeNode \*root);

*Умови задачі:*

-       використовувати цілі числа для значень у вузлах дерева

-       реалізувати функцію, що проходить по всіх вузлах дерева і міняє місцями праву і ліву вітки дерева

-       функція повертає нове дерево, передане в функцію дерево не модифікується

Self-practice task

**Одного осіннього дня ти, студенте, сидиш на лекції, яку викладач однотонно бубнить собі під ніс. Вигляд з вікна приковує до себе твій погляд — надворі з дерев опадає листя.**

**Вигляд з вікна описується прямокутником, що складається з n×mn×m клітинок. Кожна клітинка прямокутника порожня (позначається крапкою) або містить листок з дерева (позначається зірочкою). Листя падає вертикально вниз.**

**Знайди вигляд з вікна після того, як усе листя опаде. Уважаємо, що нове листя у вікні не з’явиться.**

**Вхідні дані**

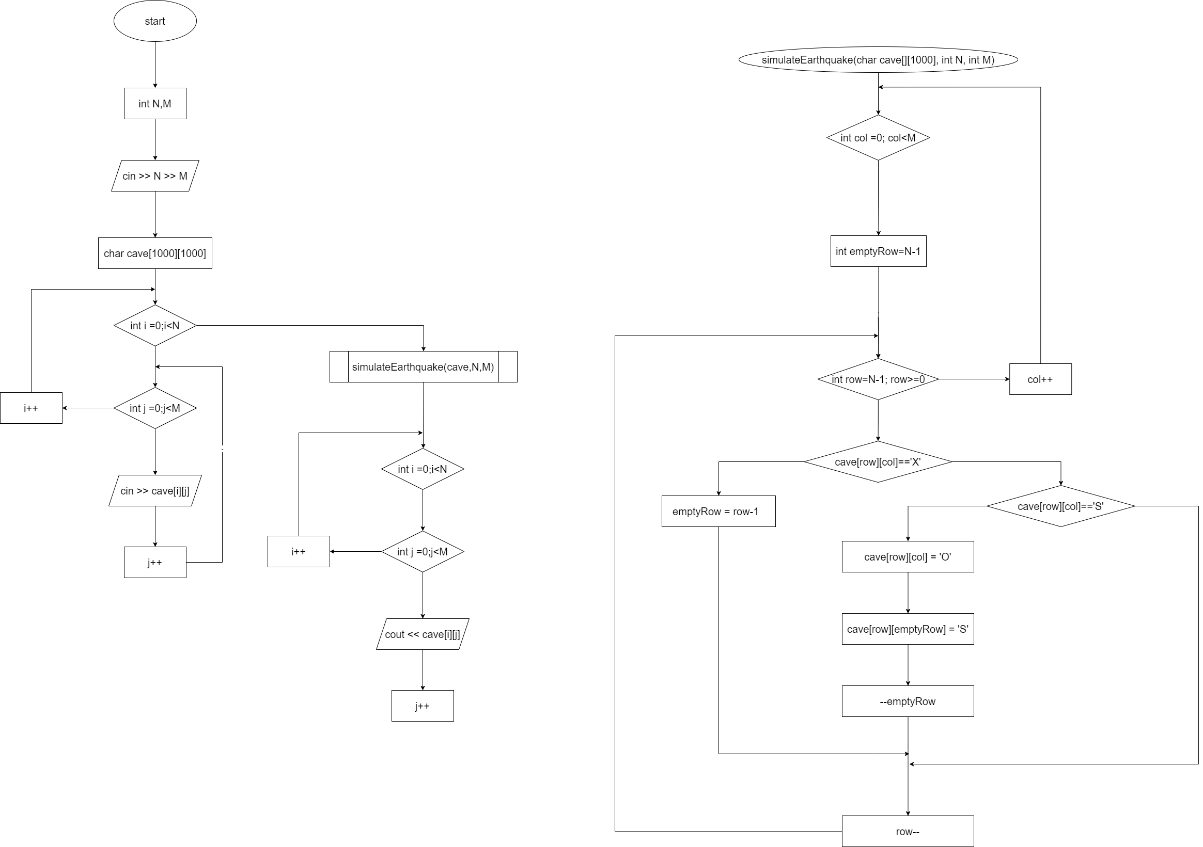
**У першому рядку задано два цілі числа n і m — розміри прямокутника.**

**У наступних n рядках задано прямокутник розміру n×m — опис вигляду з вікна.**

**Вихідні дані**

**В nn рядках виведи вигляд з вікна після опадання листя.**

Блок-схема



Коди:

**VNS LAB 10**

**`** **A screen shot of a computer program

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

**Algo lab 5**

**A screen shot of a computer program

Description automatically generated**

**Algo lab 7\_8**

**A screen shot of a computer program

Description automatically generated**

**Practice**

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

Self practice work

A screen shot of a computer program

Description automatically generated

Робота з командою



Зустрічалися в зумі 19 листопада

Висновок:

Протягом виконання цього епіку я ознайомився з основами динамічних структур даних та їх основними операціями, такими як вставка, видалення та пошук. Перш за все, я вивчив принципи виділення пам'яті для різних структур даних, зокрема для стеку та черги. Оволодів технікою роботи з цими структурами, зокрема операціями push, pop, top для стека і enqueue, dequeue, front для черги.

Далі я вивчив зв'язні списки, зокрема однозв'язні та двозв'язні, а також їх основні операції. Це дозволило мені краще зрозуміти їх структуру і застосування. Окремо я ознайомився з деревами, зокрема з бінарними деревами і деревами пошуку, що є важливою частиною роботи з динамічними структурами.

Завершенням навчання стало вивчення алгоритмів обробки динамічних структур, таких як сортування та пошук, що допомогло мені краще зрозуміти, як ефективно працювати з цими структурами в реальних задачах.

У результаті я значно поглибив свої знання в області динамічних структур даних і алгоритмів їх обробки, що є важливим кроком для подальшого вивчення програмування.