Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

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

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

***з дисципліни:*** «Мови та парадигми програмування»

до:

ВНС Лабораторної Роботи № 10

Алготестер Лабораторної Роботи № 5

Алготестер Лабораторної Роботи № 7-8

Практичних Робіт № 6

***Виконав:***

студентгрупи ШІ-13

Літовінський Роман

### Тема роботи:

Динамічні структури. Бінарне дерево. Динамічний масив. Лінкед ліст

### Мета роботи:

Ознайомлення з динамічними структурами,бінарним деревом, динамічним масивов, лінкед лістом

### Теоретичні відомості:

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Динамічні структури
* Тема №2: Бінарне дерево
* Тема №3 Динамічний масив
* Тема №4 Лінкед і дабл лінкед ліст

1. Індивідуальний план опрацювання теорії:

**Тема №1 Динамічні структури**

* + Джерела Інформації

https://purecodecpp.com/uk/archives/1538

* + Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 5.12.2023
  + Звершення опрацювання теми: 5.12.2023

**Тема №2 Бінарне дерево**

* + Джерела Інформації:

https://www.bestprog.net/ru/2023/08/01/c-binary-search-tree-binarytree-class-integers-ru/#google\_vignette

* + Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 5.12.2023
  + Звершення опрацювання теми: 5.12.2023

**Тема №3 Динамічний масив**

* + Джерела Інформації

https://acode.com.ua/urok-90-dynamichni-masyvy/

Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 5.12.2023
  + Звершення опрацювання теми: 5.12.2023

**Тема №4 Лінкед і дабл лінкед ліст**

* + Джерела Інформації

https://www.bestprog.net/uk/2022/02/13/c-an-example-of-the-implementation-of-a-linear-singly-linked-list-ua/

Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 5.12.2023
  + Звершення опрацювання теми: 5.12.2023

### Виконання роботи:

**Завдання №1**

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

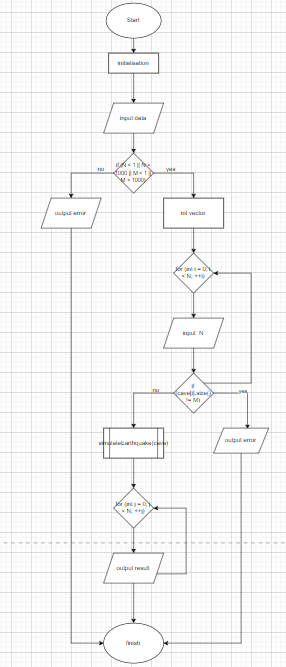
**Завдання №1**

Тут вказуєте

-Варіант 2,Algolab 5

У даній задачі, потрібно симулювати землетрус у двовимірній печері. В печері є пустота (означена символом "O"), пісок ("S") та каміння ("X"). Землетрус призводить до того, що весь пісок падає вниз до найнижчої пустої клітинки, але не може пройти через каміння. Потрібно розмістити пісок під камінням, якщо під камінням є пустота. Тобто, якщо нижня клітина каміння є пустою, пісок опиниться на цій клітинці. Процес продовжується до тих пір, поки пісок не опиниться на нижніх клітинах каміння або досягне дна печери.

1. *Дизайн та планована оцінка часу виконання завдань:*

**

1. *Код програми*

#include <iostream>

#include <vector>

using namespace std;

void simulateEarthquake(int N, int M, vector<vector<char>>& cave) {

    for (int col = 0; col < M; ++col) {

        for (int row = N - 1; row >= 0; --row) {

            if (cave[row][col] == 'S') {

                for (int k = row + 1; k < N; ++k) {

                    if (cave[k][col] == 'X' || cave[k][col] == 'S') {

                        break;

                    } else if (cave[k][col] == 'O') {

                        swap(cave[k][col], cave[k - 1][col]);

                    }

                }

            }

        }

    }

    for (int i = 0; i < N; ++i) {

        for (int j = 0; j < M; ++j) {

            cout << cave[i][j];

        }

        cout << endl;

    }

}

int main() {

    int N, M;

    cin >> N >> M;

    vector<vector<char>> caveMap(N, vector<char>(M));

    for (int i = 0; i < N; ++i) {

        for (int j = 0; j < M; ++j) {

            cin >> caveMap[i][j];

        }

    }

    simulateEarthquake(N, M, caveMap);

    return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*



Потрачено часу: 1 день

**Завдання №2**

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

-Варіант 1,Algolab 78

1. *Код програми*

#include <iostream>

using namespace std;

struct TreeNode {

    int data;

    TreeNode\* left;

    TreeNode\* right;

    TreeNode(int number) : data(number), left(nullptr), right(nullptr) {}

};

TreeNode\* createNewTree() {

    return nullptr;

}

TreeNode\* insert(TreeNode\* root, int number) {

    if (root == nullptr) {

        return new TreeNode(number);

    }

    if (number < root->data) {

        root->left = insert(root->left, number);

    } else if (number > root->data) {

        root->right = insert(root->right, number);

    }

    return root;

}

bool contains(TreeNode\* node, int number) {

    while (node != nullptr) {

        if (number == node->data) {

            return true;

        } else if (number < node->data) {

            node = node->left;

        } else {

            node = node->right;

        }

    }

    return false;

}

int size(TreeNode\* node) {

    if (node == nullptr) {

        return 0;

    }

    int leftSize = size(node->left);

    int rightSize = size(node->right);

    int size = 1 + leftSize + rightSize;

    return size;

}

void print(TreeNode\* node) {

    if (node != nullptr) {

        print(node->left);

        cout << node->data << " ";

        print(node->right);

    }

}

int main() {

    int Q;

    cin >> Q;

    TreeNode\* root = createNewTree();

    for (int i = 0; i < Q; i++){

        string option;

        cin >> option;

        if (option == "insert") {

            int number;

            cin >> number;

            root = insert(root, number);

        } else if (option == "contains") {

            int number;

            cin >> number;

            if (contains(root, number)==true) {

                cout << "Yes"<< endl;

            } else {

                cout << "No" << endl;

            }

        } else if (option == "size") {

            cout << size(root) << endl;

        } else if (option == "print") {

            print(root);

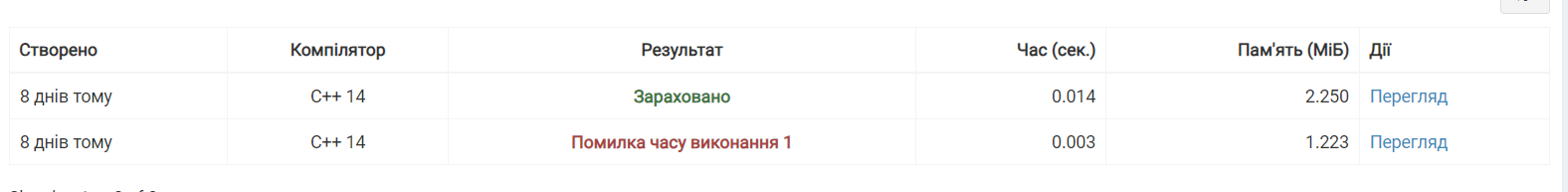
            cout << endl;

        }

    }

    return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*

Потрачено часу: 1 день

**Завдання №3**

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

-Варіант 5,VNS Lab 10

Записи в лінійному списку містять ключове поле типу int. Сформувати

однонаправлений список. Знищити з нього К елементів, починаючи із

заданого номера, додати К елементів, починаючи із заданого номера;

1. *Код програми*

#include <iostream>

#include <fstream>

#include <vector>

using namespace std;

// Функція для запису списку у файл

void zapis\_do\_failu(const vector<int>& lista, const string& nazva\_failu) {

    ofstream file(nazva\_failu);

    if (file.is\_open()) {

        for (const auto& elem : lista) {

            file << elem << " ";

        }

        file.close();

        cout << "list successfully saved to file " << nazva\_failu << endl;

    } else {

        cout << "Error opening file for writing!" << endl;

    }

}

// Функція для виведення списку

void druk\_spysku(const vector<int>& lista) {

    if (lista.empty()) {

        cout << "the list is empty" << endl;

    } else {

        cout << "List: ";

        for (const auto& elem : lista) {

            cout << elem << " ";

        }

        cout << endl;

    }

}

int main() {

    vector<int> lista = {1, 2, 3, 4, 5}; // Приклад списку

    // Записуємо список у файл

    zapis\_do\_failu(lista, "spysok.txt");

    // Очищуємо список

    lista.clear();

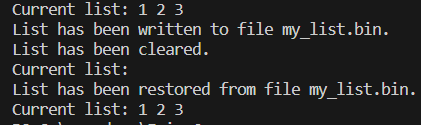
    // Виводимо список

    druk\_spysku(lista);

    return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*



Потрачено часу: 1 день

**Завдання №4**

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

- Class Practice Work

1. *Код програми*

#include <iostream>

struct Node {

    int data;

    Node\* next;

    Node(int value) : data(value), next(nullptr) {}

};

Node\* reverse(Node\* head) {

    Node\* prev = nullptr;

    while (head) {

        Node\* next = head->next;

        head->next = prev;

        prev = head;

        head = next;

    }

    return prev;

}

void printList(Node\* head) {

    while (head) {

        std::cout << head->data << " ";

        head = head->next;

    }

    std::cout << std::endl;

}

int main() {

    Node\* head = new Node(1);

    head->next = new Node(2);

    head->next->next = new Node(3);

    head->next->next->next = new Node(4);

    std::cout << "Input list: ";

    printList(head);

    head = reverse(head);

    std::cout << "Reversed list: ";

    printList(head);

    while (head) {

        Node\* temp = head;

        head = head->next;

        delete temp;

    }

    return 0;

}

**

1. *Результати виконання завдань, тестування та фактично затрачений час:*

### Висновки:

Ознайомлений з динамічними структурами,бінарним деревом, динамічним масивов, лінкед лістом.