Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра систем штучного інтелекту

Звіт

про виконання лабораторних та практичних робіт блоку № 4

На тему: «Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.

з дисципліни: «Основи програмування»

до:

Практичних Робіт до блоку № 4

Виконав:
Студент групи ШІ-13
Недосіка Назарій Вадимович

Львів 2024

Тема роботи:
Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.

Мета роботи:
Навчитися працювати з одновимірними, двовимірними та динамічними масивами, вказівниками та посиланнями, структурами та вкладеними структурами.

Теоретичні відомості:

1. **Теми, необхідні для виконання роботи:**
    - Одновимірні та двовимірні масиви
    - Динамічні масиви
    - Структури та вкладені структури
    - Вказівники та посилання
2. **Джерела використані для ознайомлення з вищезазначеними темами:**

    Лекції та практичні
    W3school
    ChatGPT
    YouTube

Виконання роботи:

Завдання №1 - Class Practice Work

Перевірка чи слово або число є паліндромом

Реалізувати програму, яка перевіряє, чи дане слово чи число є паліндромом за допомогою рекурсії.

Мета Задачі

Навчитися користуватися механізмами перевантаження функції та використовувати рекурсію для вирішення задач обчислення.

Вимоги:

1. Визначення функції:

Реалізуйте рекурсивну функцію *isPalindrome*, яка перевіряє, чи заданий рядок є паліндромом.

2. Приклад визначення функції:

bool isPalindrome(const string& str, int start, int end);

3. Перевантаження функцій:

Перевантажте функцію *isPalindrome* для роботи з цілими значеннями.
bool isPalindrome(ціле число);

4. Рекурсія:

Рекурсивна функція для рядків перевірить символи в поточній початковій і кінцевій позиціях. Якщо вони збігаються, він буде рекурсивно перевіряти наступні позиції, поки початок не перевищить кінець, після чого рядок буде визначено як паліндром.

Кроки реалізації

- Визначте та реалізуйте рекурсивну функцію isPalindrome для рядків.
- Визначте та реалізуйте перевантажену функцію isPalindrome для цілих чисел. Використати математичний підхід щоб перевірити чи число є паліндромом.

Розв'язок задачі:

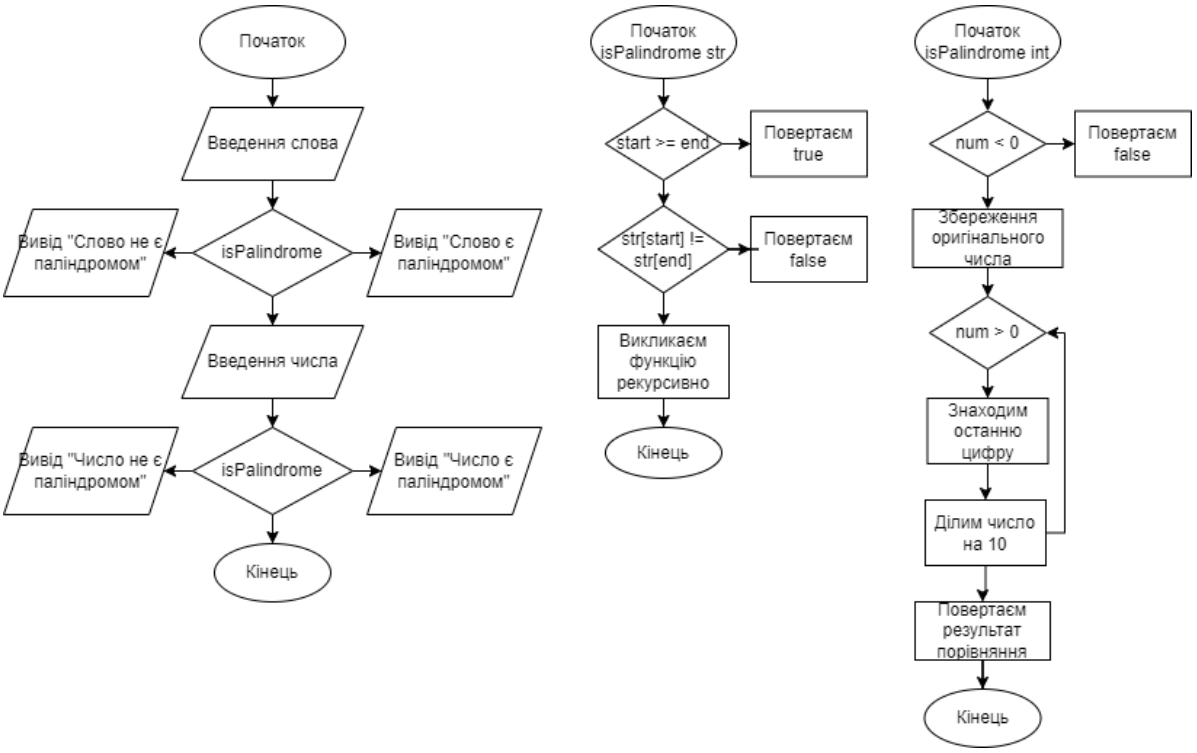
```cpp
#include <iostream>
#include <string>

using namespace std;

bool isPalindrome(const string& str, int start, int end)
{
    if (start >= end)
        return true;

    if (str[start] != str[end])
        return false;

    return isPalindrome(str, start + 1, end - 1);
}

bool isPalindrome(int num)
{
    if (num < 0)
        return false;

    int original = num;
    int reversed = 0;

    while (num > 0)
    {
        reversed = reversed * 10 + num % 10;
        num /= 10;
    }

    return original == reversed;
}

int main()
{
    string input1;
    int input2;

    cout << "Введіть слово: ";
    cin >> input1;

    if (isPalindrome(input1, 0, input1.length() - 1))
    {
        cout << "Це слово є паліндромом" << endl;
    }
    else
    {
        cout << "Це слово не є паліндромом" << endl;
    }

    cout << "Введіть число: ";
    cin >> input2;

    if (isPalindrome(input2))
    {
        cout << "Це число паліндром" << endl;
    }
    else
    {
        cout << "Це число не є паліндромом" << endl;
    }

    return 0;
}
```

Вивід:

```
Введіть слово: radar
Це слово є паліндромом
Введіть число: 330010033
Це число паліндром
```

```
Введіть слово: Palindrome
Це слово не є паліндромом
Введіть число: 12345
Це число не є паліндромом
```

Блок схема до задачі:



Час виконання завдання: ~ 2 години

Завдання №2– VNS Lab 4 - Task 1-14

1) Сформувати одновимірний масив цілих чисел, використовуючи генератор випадкових чисел.
2) Роздрукувати отриманий масив.
3) Перевернути масив і, якщо кількість елементів масиву непарна, знищити його середній елемент.
4) Додати на початок масиву 3 елементи зі значенням M[I+10]-2.
5) Роздрукувати отриманий масив.

Вивід:

```
Введіть кількість елементів масиву(5 - 20): 7
Оригінальний масив: 39 49 98 61 73 87 50
Модифікований масив: 81 95 58 50 87 73 98 49 39
```

Розв’язок задачі:

```cpp
using namespace std;

int main() {
    int size;
    cout << "Введіть кількість елементів масиву(5 - 20): ";
    cin >> size;

    srand(time(0));
    vector<int> array(size);

    for (int &num : array) {
        num = rand() % 100 + 1;
    }

    cout << "Оригінальний масив: ";
    for (int num : array) {
        cout << num << " ";
    }
    cout << endl;

    reverse(array.begin(), array.end());

    if (array.size() % 2 != 0) {
        array.erase(array.begin() + array.size() / 2);
    }

    vector<int> M;
    for (int i = 0; i < array.size(); ++i) {
        M.push_back(array[i] + 10);
    }

    for (int i = 0; i < 3; ++i) {
        array.insert(array.begin(), M[i] - 2);
    }

    cout << "Модифікований масив: ";
    for (int num : array) {
        cout << num << " ";
    }
    cout << endl;

    return 0;
}
```

Час виконання завдання: ~ 1.5 години

Завдання №3 - VNS Lab 5 - Task 1-14

Обчислити добуток всіх стовпців масиву, у яких перший елемент більший від елементів розташованих на головній і бічній діагоналі.

Розв’язок задачі:

```cpp
#include <iostream>
using namespace std;

int IsMyMatrix(int arr[100][100], int n) {
    int product = 1;
    bool exist = false;

    for (int j = 0; j < n; j++) {
        int FirstNumber = arr[0][j];
        bool IsTruecolum = true;

        for (int i = 0; i < n; i++) {
            if (FirstNumber <= arr[i][i] || FirstNumber <= arr[i][n - i - 1]) {
                IsTruecolum = false;
                break;
            }
        }

        if (IsTruecolum) {
            exist = true;
            for (int r = 0; r < n; r++) {
                product *= arr[r][j];
            }
        }
    }

    return exist ? product : 0;
}

int main() {
    int n;

    cout << "Введіть розмір квадратної матриці: ";
    cin >> n;
    cout << "Введіть елементи матриці" << endl;
    int arr[100][100];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> arr[i][j];
        }
    }

    int result = IsMyMatrix(arr, n);
    if (result != 0) {
        cout << result << endl;
    } else {
        cout << "Таких стовпців немає" << endl;
    }

    return 0;
}
```

Вивід:

```
Введіть розмір квадратної матриці: 3
Введіть елементи матриці
5 20 3
1 2 0
7 3 9
120
```

Час виконання завдання: ~ 1.5 години

Завдання №4 – Algotester lab 2 variant 3

Вам дано масив цілих чисел розміром N, на першій та останній клітинці розміщено по дрону. Вони одночасно взлітають.

На початку кожного ходу швидкість дрону стає рівною значенню клітинки, у якій він знаходиться.

Тобто лівий дрон у першу секунду з клітинки з індексом перелетить у клітинку з індексом a1, тобто його наступна позиція рахується як поточна позиція + число у поточній позиції (перегляньте пояснення для візуалізації) Правий робить аналогічно в протилежну сторону.
Вони це роблять до моменту, коли трапиться одна з зазначених подій:

Якшо 2 дрони опиняються в одній клітинці - ви виводите **Collision**.

Якщо лівий дрон опиниться справа від правого - це **Miss**

У випадку якщо вони зупиняться один навпроти одного, тобто у клітинках ai та ai+1 - виведіть **Stopped**
Врахуйте, що перевіряти треба також до взльоту.

Input

У першому рядку ціле число N - розмір масиву
У другому рядку N цілих чисел - елементи масиву
Output
У першому рядку фінальна позиція першого та другого дрона.

У другому рядку одне зі слів

Розв’язок задачі:

```cpp
#include <iostream>
#include <vector>

using namespace std;

int main(){
    int size;
    cin >> size;
    vector<int> vec(size);

    for(int i = 0; i < size; i++)cin >> vec[i];

    int leftDrone = 1;
    int rightDrone = size;
    int i = 0;
    int j = size - 1;

    while(i != j && i < j && j != i + 1){
        leftDrone += vec[i];
        i += vec[i];
        rightDrone -= vec[j];
        j -= vec[j];
    }

    if(i == j){
        cout << leftDrone << " " << rightDrone << "\nCollision";
    }

    else if(j == i + 1){
        cout << leftDrone << " " << rightDrone << "\nStopped";
    }

    else{
        cout << leftDrone << " " << rightDrone << "\nMiss";
    }

    return 0;
}
```

Вивід:

```
1 3 1 1 5 1 1 2 1 2
5 6
Stopped
```

Час виконання завдання: ~ 2 години

Завдання №5 – Algotester lab 3 variant 3

Вам дана стрічка s.
Ваше завдання зробити компресію стрічки, тобто якщо якась буква йде більше одного разу підряд у стрічці замінити її на букву + кількість входжень підряд.

Input
У першому рядку стрічка SS
Output
Стрічка Scompressed

Розв'язок задачі:

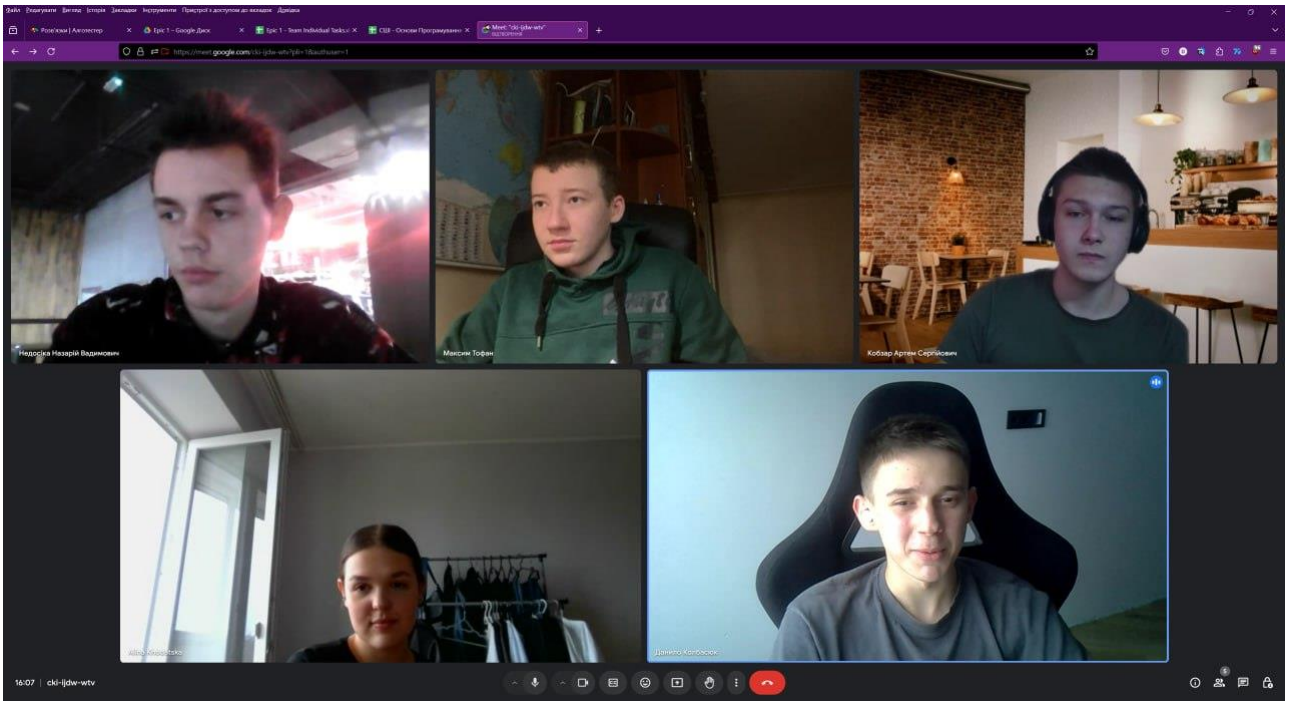
```cpp
#include <iostream>
#include <string>
using namespace std;

string compress_string(const string& s) {
    string result;
    int n = s.length();

    for (int i = 0; i < n; ) {
        char current_char = s[i];
        int count = 1;

        while (i + 1 < n && s[i + 1] == current_char) {
            count++;
            i++;
        }

        result += current_char;
        if (count > 1) {
            result += to_string(count);
        }

        i++;
    }

    return result;
}

int main() {
    string s;
    cin >> s;

    cout << compress_string(s) << endl;

    return 0;
}
```

Вивід:

```
AAAABBBCDD
A4B3CD2
```

Час виконання завдання: ~ 45 хв

Meet



Висновки: Виконуючи цей епік я поглибив свої знання у зберіганні та обробці даних. Застосування вказівників і динамічних масивів дозволяє зекономити пам'ять і гнучко керувати даними, а розуміння структур даних та алгоритмів обробки допомагає створювати оптимізовані рішення для різноманітних задач.