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

Звіт

про виконання лабораторних та практичних робіт блоку № 4
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:

ВНС Лабораторної Роботи №4
ВНС Лабораторної Роботи №5
Алготестер Лабораторної Роботи №2
Алготестер Лабораторної Роботи №3
Практичних Робіт до блоку №4

Виконав:
Студент групи ШІ-13
Шийка Стефан Андрійович

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:

Дослідження одновимірних і двовимірних масивів для зберігання і впорядкування даних, що забезпечує швидкий доступ і обробку великих обсягів інформації.

Дослідження вказівників та посилань для розуміння адресації пам’яті та оптимізації використання ресурсів, що дозволяє ефективніше працювати з динамічними структурами даних.

Дослідження динамічних масивів для створення програм із змінною кількістю елементів, що підвищує гнучкість і адаптивність коду.

Дослідження структур даних та вкладених структур для організації складних об'єктів, що забезпечує кращу структуру і читабельність програмного коду.

Дослідження алгоритмів обробки масивів і структур для реалізації ефективної обробки даних, що сприяє написанню оптимізованих і масштабованих програм.

Теоретичні відомості:

У даній роботі розглядаються основні принципи роботи з масивами та структурами даних, зокрема одновимірні й двовимірні масиви для організації і зберігання великих обсягів даних. Особливу увагу приділено вказівникам і посиланням як засобам управління пам'яттю та ефективного доступу до даних. Розглянуто динамічні масиви, які забезпечують гнучке управління розміром даних під час виконання програми. Досліджено основи структур даних і вкладених структур для створення складних, логічно організованих об’єктів. Описано алгоритми обробки масивів і структур, що дозволяють ефективно виконувати операції пошуку, сортування і модифікації даних, покращуючи оптимізацію коду.

Джерела:

-aCode

- Harvard CS50 lectures+tasks

- chatGPT / Google

Lab# programming: VNS Lab 4

Time estimated: 1h+

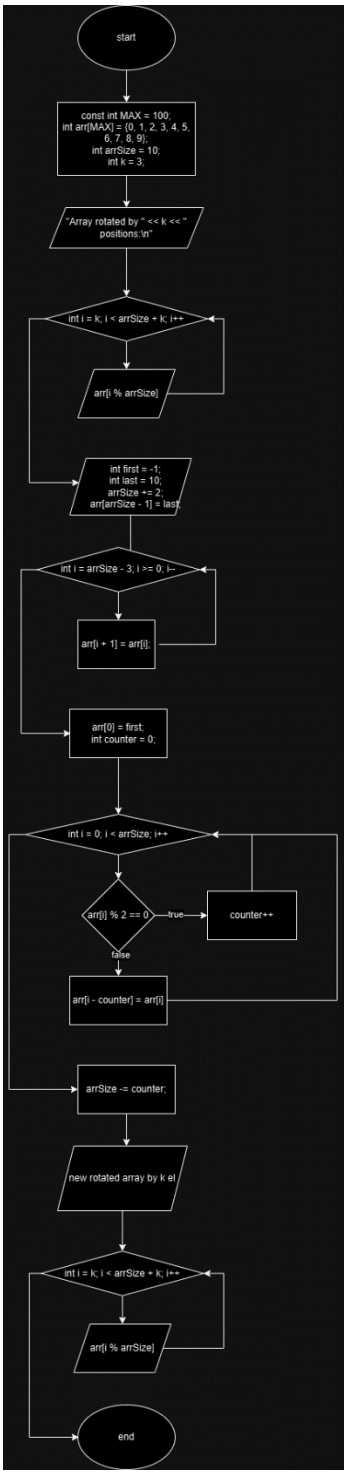
Spent: 30min

18.

1) Реалізувати з використанням масиву однонаправлене кільце (перегляд можливий зліва направо, від останнього елемента можна перейти до першого).
2) Роздрукувати отриманий масив, починаючи з К-ого елемента і до К-1.
3) Додати в кільце перший і останній елементи.
4) Знищити з кільця парні елементи.
5) Роздрукувати отриманий масив, починаючи з К-ого елемента і до К-1.

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

const int MAX = 100;
int arr[MAX] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
int arrSize = 10;
int k = 3;

int main() {
    cout << "Array rotated by " << k << " positions:\n";
    for (int i = k; i < arrSize + k; i++) {
        cout << arr[i % arrSize] << " ";
    }
    cout << endl;

    int first = -1;
    int last = 10;
    arrSize += 2;
    arr[arrSize - 1] = last;

    for (int i = arrSize - 3; i >= 0; i--) {
        arr[i + 1] = arr[i];
    }
    arr[0] = first;

    int counter = 0;
    for (int i = 0; i < arrSize; i++) {
        if (arr[i] % 2 == 0) {
            counter++;
        } else {
            arr[i - counter] = arr[i];
        }
    }
    arrSize -= counter;

    cout << "Array rotated by " << k << " positions after adding new first and last elements and deleting the even ones:\n";
    for (int i = k; i < arrSize + k; i++) {
        cout << arr[i % arrSize] << " ";
    }
    cout << endl;
}
```

```
Array rotated by 3 positions:
3 4 5 6 7 8 9 0 1 2
3 4 5 6 7 8 9 0 1 2
Array rotated by 3 positions after adding new first and last elements and deleting the even ones:
5 7 9 -1 1 3
PS C:\Users\user\Desktop\c++\epic4>
```



Lab# programming: VNS Lab 5

Time estimated: 2h

Spent: 3h+

18. Задано двовимірний масив N x N. Послідовно розглядаються квадратні підмасиви, правий верхній елемент яких лежить на бічній діагоналі. У кожному такому підмасиві перебуває максимальний елемент. Шляхом перестановок рядків і стовпців (повністю) елемент треба перемістити в правий верхній кут підмасиву. Перевірити чи вийшла на бічній діагоналі спадаюча послідовність елементів.

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

const int N = 4;

void rearrangeMatrix(int matrix[N][N]);
void moveMaxToTopRight(int matrix[N][N], int startRow, int startCol);
void swapRows(int matrix[N][N], int row1, int row2);
void swapColumns(int matrix[N][N], int col1, int col2);
bool checkDiagonal(int matrix[N][N]);

int main() {
    int matrix[N][N] = {
        {12, 8, 3, 5},
        {7, 15, 9, 10},
        {2, 5, 18, 4},
        {6, 11, 14, 20}
    };

    rearrangeMatrix(matrix);
    bool result = checkDiagonal(matrix);

    cout << "Matrix after rearrangement:\n";
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            cout << matrix[i][j] << " ";
        }
        cout << endl;
    }

    cout << "\nIs the anti-diagonal descending: " << (result ? "Yes" : "No") << endl;
}

void rearrangeMatrix(int matrix[N][N]) {
    for (int i = 0; i < N; i++) {
        moveMaxToTopRight(matrix, N - 1 - i, N - 1 - i);
    }
}

void moveMaxToTopRight(int matrix[N][N], int startRow, int startCol) {
    int maxVal = matrix[startRow][startCol];
    int maxRow = startRow;
    int maxCol = startCol;

    for (int i = 0; i <= startRow; i++) {
        for (int j = 0; j <= startCol; j++) {
            if (matrix[i][j] > maxVal) {
                maxVal = matrix[i][j];
                maxRow = i;
                maxCol = j;
            }
        }
    }

    if (maxRow != startRow) {
        swapRows(matrix, maxRow, startRow);
    }
    if (maxCol != startCol) {
        swapColumns(matrix, maxCol, startCol);
    }
}

void swapRows(int matrix[N][N], int row1, int row2) {
    for (int col = 0; col < N; col++) {
        int temp = matrix[row1][col];
        matrix[row1][col] = matrix[row2][col];
        matrix[row2][col] = temp;
    }
}

void swapColumns(int matrix[N][N], int col1, int col2) {
    for (int row = 0; row < N; row++) {
        int temp = matrix[row][col1];
        matrix[row][col1] = matrix[row][col2];
        matrix[row][col2] = temp;
    }
}

bool checkDiagonal(int matrix[N][N]) {
    for (int i = 0; i < N - 1; i++) {
        if (matrix[i][N - 1 - i] <= matrix[i + 1][N - 2 - i]) {
            return false;
        }
    }
    return true;
}

```

```
Matrix after rearrangement:
6 2 11 20
14 7 5 4
18 12 15 10
9 0 8 5

Is the anti-diagonal descending: No
```

Lab# programming: Algotester Lab 2

Time estimated: 15min

Spent: 15 min

Lab 2v1

Limits: 1 sec., 256 MiB

У вас є дорога, яка виглядає як N чисел.

Після того як ви по ній пройдете - вашу втому можна визначити як різницю максимального та мінімального елементу.

Ви хочете мінімізувати втому, але все що ви можете зробити - викинути одне число з дороги, тобто забрати його з масиву.

В результаті цієї дії, яку мінімальну втому ви можете отримати в кінці дороги?

Input

У першому рядку ціле число N - кількість чисел

У другому рядку масив r, який складається з N цілих чисел

Output

Єдине ціле число m - мінімальна втома, яку можна отримати

```cpp
#include <iostream>
#include <climits>

using namespace std;

int main(){
    int N, m;

    cin >> N;
    if(N <= 2){
        cout << 0;
        return 0;
    }
    int r[N];

    for(int i = 0; i < N; i++){
        cin >> r[i];
    }


    int min1 = INT_MAX, min2 = INT_MAX;
    int max1 = INT_MIN, max2 = INT_MIN;

    for(int i = 0; i < N; i++){
        if (r[i] < min1) {
            min2 = min1;
            min1 = r[i];
        } else if (r[i] < min2) {
            min2 = r[i];
        }

        if (r[i] > max1) {
            max2 = max1;
            max1 = r[i];
        } else if (r[i] > max2) {
            max2 = r[i];
        }
    }

    if(max1 - min2 <= max2 - min1){
        m = max1 - min2;
    }else{
        m = max2 - min1;
    }

    cout << m;
}
```

a few seconds ago	Lab 2v1 - Lab 2v1	C++ 23	Accepted	0.003	1.234	1857424

Lab# programming: Algotester Lab 3

Time estimated: 15min

Spent: 30 min

Lab 3v3

Limits: 1 sec., 256 MiB

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 S

Output

Стрічка $S_{compressed}$

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

using namespace std;

int main() {
    string S;
    getline(cin, S);
    int len = S.length();
    string result;

    for (int i = 0; i < len; i++) {
        int counter = 1;

        while (i + 1 < len && S[i] == S[i + 1]) {
            counter++;
            i++;
        }

        result.push_back(S[i]);

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

    cout << result;
}
```

a few seconds ago	Lab 3v3 - Lab 3v3	C++ 23	Accepted	0.003	1.254	1857430

Practice# programming: Class Practice Task

Time estimated: 45min

Spent: 25min

```cpp
#include <iostream>
#include<cmath>

using namespace std;

bool isPalindrome(const string& s, int start, int end){
    if(start >= end) return true;
    if(s[start] != s[end]) return false;
    return isPalindrome(s, start+1, end-1);
}

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

    int original = n;
    int reversed = 0;

    while (n > 0) {
        int digit = n % 10;
        reversed = reversed * 10 + digit;
        n /= 10;
    }

    return original == reversed;
}

int main(){
    string s;
    int n;
    cout << "Input a string: ";
    getline(cin, s);
    cout << endl;
    cout << "Input an integer: ";
    cin >> n;
    cout << endl;

    cout << "String is palindrome: " << (isPalindrome(s, 0, s.length()-1) ? "Yes" : "No") << endl;
    cout << "Integer is palindrome: " << (isPalindrome(n) ? "Yes" : "No") << endl;
}
```

```
Input a string: radar

Input an integer: 112211

String is palindrome: Yes
Integer is palindrome: Yes
```

```
Input a string: monkey

Input an integer: 12343

String is palindrome: No
Integer is palindrome: No
```

Practice# programming: Self Practice Task

Time estimated: 20min

Spent: 45min

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

using namespace std;

int main(){
    int k;
    string substr = "TOILET";
    string s;
    cin >> k;
    cin.ignore();
    getline(cin, s);

    int len = s.length();
    int counter = 0;
    int position = 0;

    while((position = s.find(substr, position)) != string::npos){
        counter++;
        position += substr.length();
    }

    cout << ((k <= counter) ? "YES" : "NO");
}
```

```
2
HELPTOILETMENPLEASETOILET
YES
PS C:\Users\user\Desktop\c++\epic4>
```

2 minutes ago	0081 - Допоможе чи заб'є?	C++ 23	Accepted	0.005	1.168	1857548

PULL

Висновок: В процесі виконання лабораторної роботи я навчився використовувати одновимірні та двовимірні масиви для зберігання і впорядкування даних, що покращує доступ до великого обсягу інформації. Також я ознайомився з поняттями вказівників і посилань, що дозволяє ефективно управляти пам’яттю і використовувати динамічні масиви. Окрім того, я досліджував структури даних та алгоритми обробки масивів, що сприяє написанню оптимізованих і масштабованих програм.