

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



**Звіт**  
**про виконання лабораторних та практичних робіт блоку**  
**№ 4**  
**з дисципліни:** «Основи програмування»

**Виконав:**

Студент групи ШІ-11

Лопатін Володимир Дмитрович

Львів 2024

## **Тема:**

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

## **Мета роботи:**

Ознайомитися з принципами роботи з одновимірними та двовимірними масивами, вказівниками та посиланнями, а також динамічними масивами у мові програмування C++. Навчитися створювати й використовувати структури даних, включаючи вкладені структури. Освоїти алгоритми обробки масивів і структур для розв'язання задач із зберігання, організації та аналізу даних. Отримати практичні навички створення ефективних програм з використанням масивів, вказівників і структур даних.

## **Теоретичні відомості:**

- Одновимірні масиви
- Двовимірні масиви
- Вказівники та посилання
- Динамічні масиви
- Структури даних
- Вкладені структури
- Алгоритми обробки масивів

### Одновимірні масиви:

Був знайомий до того доознайомився під час роботи.  
Витрачено 30 хв.

### Двовимірні масиви:

Знайомий, проте на практиці важче.  
Витратив 50 хвилин.

### Вказівники та посилання:

Розумів концепцію до того, але ще краще [ознайомився тут](#)  
Витратив 45 хвилин.

### Динамічні масиви:

Користувався до того, ознайомлювався з іншими випадками.  
Витратив 40 хвилин.

### Структури даних:

Пояснили ChatGPT та викладач на парі.  
На повне ознайомлення загалом витратив 1 годину.

### Вкладені структури:

Розібрався завдяки ChatGPT та [сервісу Acode](#).  
Витратив 30 хвилин.

### Алгоритми обробки масивів:

Пояснювали на парі, потім ще ChatGPT.  
Витрачено 1 годину.

## Виконання роботи:

- 1) Опрацювання завдання та вимог до програм та серидовища:

### Завдання №1

«Перевірка чи слово або число є паліндромом»

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

#### Вимоги:

- Визначення функції:
  - Реалізуйте рекурсивну функцію *isPalindrome*, яка перевіряє, чи заданий рядок є паліндромом.
- Приклад визначення функції:
  - *bool isPalindrome(const string& str, int start, int end);*
- Перевантаження функцій:
  - Перевантажте функцію *isPalindrome* для роботи з цілими значеннями.
  - *bool isPalindrome(ціле число);*
- 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.

### Завдання №2

«Лабораторна №4 з ВНС»

Потрібно побудувати двонаправлене кільце та вивести його починаючи від К-го елемента, потім потрібно було

додати ще один елемент з кінця та спочатку і знову вивести кільце без парних елементів.

### **Завдання №3**

«Лабораторна 5 з ВНС»

Потрібно було перевірити, чи з введеного рядка чисел довжиною  $N^2$  можна побудувати матрицю розміру  $N \times N$  зі зростаючим першим стовпцем.

### **Завдання №4**

«Лабораторна 2 з Algotester»

Потрібно ввести масив цілих чисел, видалити з нього 3 елементи, а потім вивести масив із сум елементів нового масиву, якщо це можливо.

### **Завдання №5**

«Лабораторна 3 з Algotester»

Завдання ввести два масиви, а потім вивести спочатку кількість однакових елементів, а потім – унікальних.

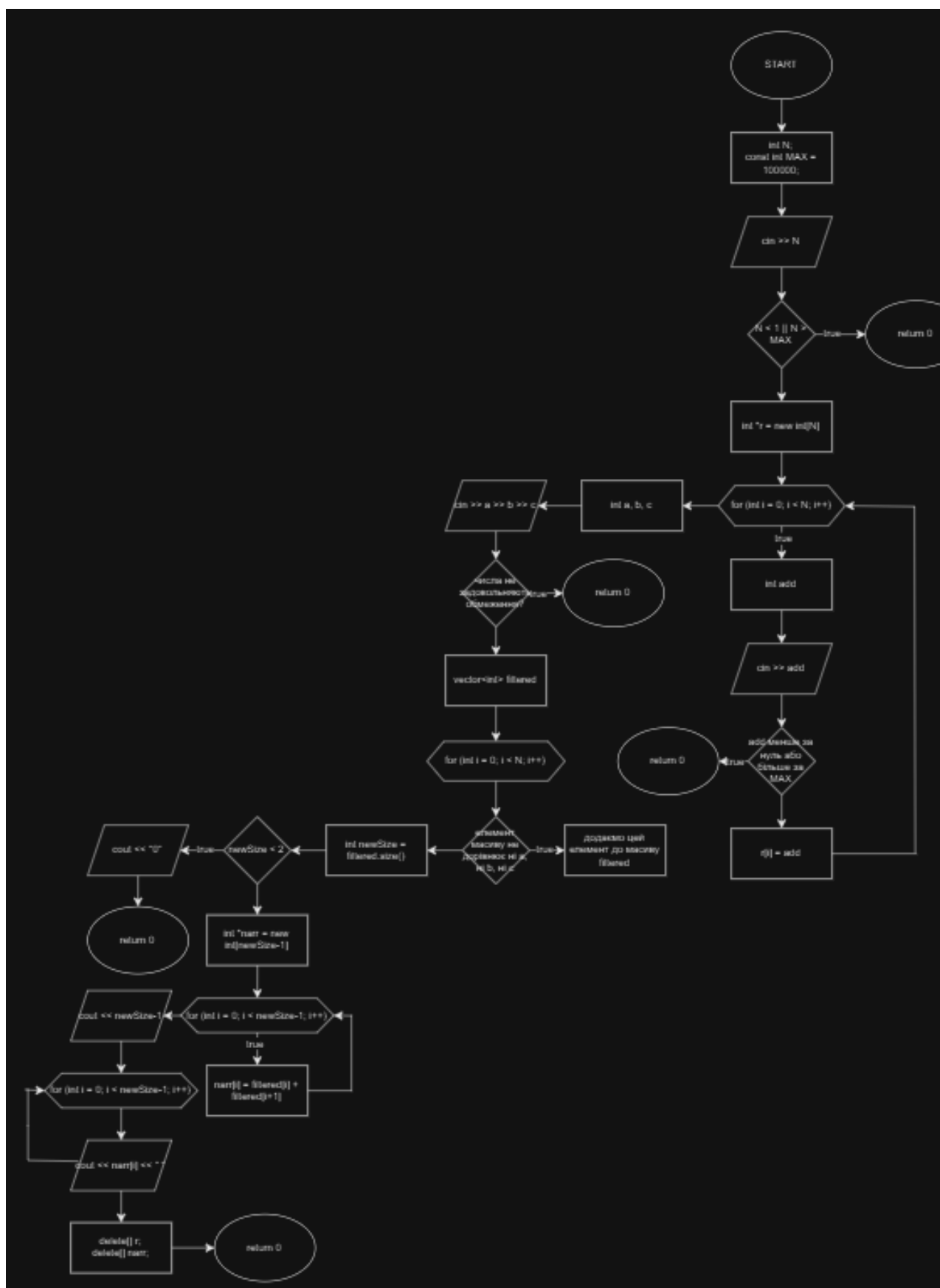
### **Завдання №6**

«Вибіркове завдання з Algotester «Непарний масив»»

Задача полягала в тому, щоб ввести масив цілих чисел і переставити елементи в ньому так, щоб сума двох послідовних елементів була непарним числом(якщо це можливо).

2) Дизайн та планова оцінка часу виконання завдань:

## Завдання №4



1) Код програм з посиланням на зовнішні ресурси:

## Завдання №1

```
#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) {
    string str = to_string(num);
    return isPalindrome(str, 0, str.length()-1);
}

int main() {
    string str;
    int start, end, num;
    cin >> str >> start >> end >> num;
    cout << "For string: " << isPalindrome(str, start, end) << endl;
    cout << "For number: " << isPalindrome(num);
    return 0;
}
```

## Завдання №2

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

void printRing(const vector<int>& ring, int k) {
    for (int i = 0; i < ring.size(); i++) {
        cout << ring[(i + k) % ring.size()] << " ";
    }
    cout << endl;
}

int main() {
    vector<int> ring;
    int k, siz, firstEl, lastEl;

    cin >> siz >> k >> firstEl >> lastEl;
    for (int m = 0; m < siz; m++) {
        int newEl;
        cin >> newEl;
        ring.push_back(newEl);
    }

    printRing(ring, k);
    ring.insert(ring.begin(), firstEl);
    ring.push_back(lastEl);

    for (int n = ring.size() - 1; n >= 0; n--) {
        if (ring[n] % 2 == 0) {
            ring.erase(ring.begin() + n);
        }
    }

    printRing(ring, k);

    return 0;
}
```



## Завдання №3

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

bool canBuild(int n, const string& input) {
    if (input.size() != n*n) return false;

    vector<vector<int>> matrix(n, vector<int>(n));
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            matrix[i][j] = input[(i * n) + j] - '0';
        }
    }

    for (int k = 1; k < n; ++k) {
        if (matrix[k][0] <= matrix[k-1][0]) return false;
    }
    return true;
}

int main() {
    string input;
    int n;
    cin >> input;
    cout << endl;
    cin >> n;
    if (canBuild(n, input) == true) {
        cout << "Можливо побудувати матрицю із зростаючим першим стовпцем.";
    } else {
        cout << "Неможливо побудувати матрицю із зростаючим першим стовпцем.";
    }
    return 0;
}
```

## Завдання №4

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

int main() {
    int N;
    const int MAX = 100000;
    cin >> N;
    if (N < 1 || N > MAX) return 0;
    int *r = new int[N];
    for (int i = 0; i < N; i++) {
        int add;
        cin >> add;
        if (add < 0 || add > MAX) return 0;
        r[i] = add;
    }
    int a, b, c;
    cin >> a >> b >> c;
    if (a < 0 || a > MAX || b < 0 || b > MAX || c < 0 || c > MAX) return 0;

    vector<int> filtered;
    for (int i = 0; i < N; i++) {
        if (r[i] != a && r[i] != b && r[i] != c) {
            filtered.push_back(r[i]);
        }
    }
    int newSize = filtered.size();
    if (newSize < 2) {
        cout << '0';
        return 0;
    }
    int *narr = new int[newSize - 1];
    for (int i = 0; i < newSize - 1; i++) {
        narr[i] = filtered[i] + filtered[i + 1];
    }
    cout << newSize - 1 << endl;
    for (int i = 0; i < newSize - 1; i++) {
        cout << narr[i] << " ";
    }
    cout << endl;
    delete[] r;
    delete[] narr;
    return 0;
}
```

## Завдання №5

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

int main() {
    int N, M;
    cin >> N;
    if (N < 0 || N > 100) return 0;
    vector<int> firstArr;
    for (int i = 0; i < N; ++i) {
        int a;
        cin >> a;
        if (a < 0 || a > 100) return 0;
        firstArr.push_back(a);
    }
    cin >> M;
    if (M < 0 || M > 100) return 0;
    vector<int> secondArr;
    for (int i = 0; i < M; ++i) {
        int b;
        cin >> b;
        if (b < 0 || b > 100) return 0;
        secondArr.push_back(b);
    }
    int equivEl;
    for (int elF : firstArr) {
        for (int elS : secondArr) {
            if (elF == elS) {
                equivEl++;
                break;
            }
        }
    }
    int uniqEl = (M + N) - equivEl;
    cout << equivEl << endl << uniqEl;
    return 0;
}
```

## Завдання №6

```
#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

int main() {
    int n;
    cin >> n;
    if (n < 1 || n > 100000) return 0;
    vector<long long> arr;
    vector<long long> even, odd;
    for (int i = 0; i < n; ++i) {
        long long a;
        cin >> a;
        if (a < 1 || a > 1000000000) return 0;
        arr.push_back(a);
        if (arr[i] % 2 == 0) {
            even.push_back(arr[i]);
        } else {
            odd.push_back(arr[i]);
        }
    }
    if (abs((int)even.size() - (int)odd.size()) > 1) {
        cout << "-1";
        return 0;
    }
    vector<long long> result;
    bool startWithEven = even.size() >= odd.size();
    while(!odd.empty() || !even.empty()) {
        if (!even.empty() && startWithEven) {
            result.push_back(even.back());
            even.pop_back();
        } else if (!odd.empty() && !startWithEven) {
            result.push_back(odd.back());
            odd.pop_back();
        }
        startWithEven = !startWithEven;
    }
    for (long long num : result) {
        cout << num << " ";
    }
    return 0;
}
```

2) Результати виконання завдань, тестування та фактично  
затрачений час

### **Завдання №1**

```
crirc  
0 4  
3445443  
For string: 1  
For number: 1
```

Тут я ввів рядок, потім початковий та кінцевий індекси та  
число .

Витратив 30 хвилин.

### **Завдання №2**

```
5 2 16 7  
1 58 5 6 2  
5 6 2 1 58  
7 1 5
```

Я ввів перші два рядки.

Витратив на завдання близько 1 години.

### **Завдання №3**

```
143635953
```

```
3
```

```
Можливо побудувати матрицю із зростаючим першим стовпцем.
```

```
PS D:\> █
```

На це завдання пішло 25 хвилин.

## Завдання №4

```
5
```

```
1 2 7 5 9
```

```
2 8 5
```

```
2
```

```
8 16
```

```
PS D:\> █
```

Витратив на завдання приблизно 40 хвилин.

## Завдання №5

```
6
```

```
2 3 4 9 8 0
```

```
3
```

```
3 5 7
```

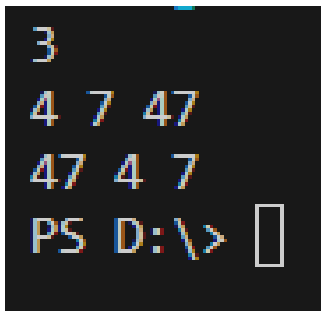
```
1
```

```
8
```

```
PS D:\> █
```

Опрацьовував завдання 30 хвилин.

## Завдання №6



```
3
4 7 47
47 4 7
PS D:\> 
```

На завдання пішло 40 хвилин.

### Висновки:

Під час виконання лабораторної роботи я ознайомився з принципами роботи з одновимірними та двовимірними масивами, вказівниками, посиланнями, динамічними масивами та структурами даних у мові програмування C++. Було реалізовано алгоритми обробки масивів і структур для вирішення завдань зі збереження, організації та аналізу даних.

Також я навчився створювати вкладені структури й ефективно використовувати їх для моделювання складних об'єктів.

Завдяки виконаним завданням вдалося закріпити навички написання ефективного та оптимізованого коду з використанням структурованих даних, динамічної пам'яті та вказівників.

[Pull request](#)