Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Звіт**

про виконання

**Лабораторних та практичних робіт № 4**

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

***з розділу***: «Прості структури даних. Одновимірні масиви. Двовимірні Масиви. Алгоритми обробки.»

***Виконала:***

студентка групи ШІ-13

Поліщук Марина Олександрівна

# **Тема роботи:**

# Прості структури даних. Одновимірні масиви. Двовимірні Масиви. Алгоритми обробки.

# **Мета роботи:**

# Навчитися працювати з простими структурами даних, одновимірними\двовимірними масивами масивами.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

-масиви

-генерація рандомних чисел

-рекурсія

-вказівники

1. Індивідуальний план опрацювання теорії:

* Тема №1:масиви
  + Джерела Інформації
    - лекції, практичні заняття
    - <https://drive.google.com/drive/u/0/folders/1RteNrVnAsO5jXanknOc2T7LesLT7Vspx>
    - Що опрацьовано:

Визначення масиву містить тип елементів, ім'я масиву й кількість

елементів у масиві. Ініціалізація масивів можлива при їхньому визначенні:

double d[] = {1, 2, 3, 4, 5};

Довжина масиву обчислюється компілятором за кількістю значень

перерахованих у фігурних дужках. Відповідно до синтаксису в С існують тільки одновимірні масиви, але їхніми елементами, у свою чергу, теж можуть бути масиви.

* + - Статус: Ознайомлена
    - Початок опрацювання теми: 10.10.23
    - Звершення опрацювання теми: 08.11.23
* Тема №3: рекурсія
  + Джерела Інформації:
    - практичні заняття, лекції

<https://www.youtube.com/watch?v=V7q9w_s0nns>

* + Що опрацьовано:
    - Рекурсія-це коли функція викликає сама себе. Головний недолік: витрачається багато пам’яті, оскільки нам потрібно зберігати в пам’яті всі проміжні результати (тому використання циклу в простих прикладах часто є ефективнішим)
    - складна рекусія- це коли функція, яка викликає другу функцію, яка в своєму коді викликає десь першу функцію.

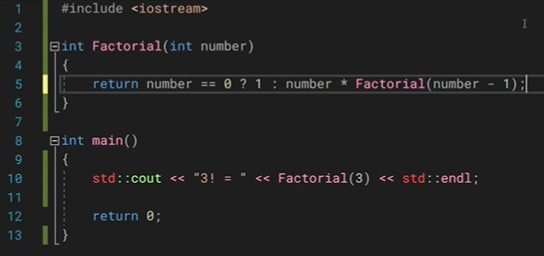


Рисунок 1 приклад застосування рекурсії для обчислення факторіалу

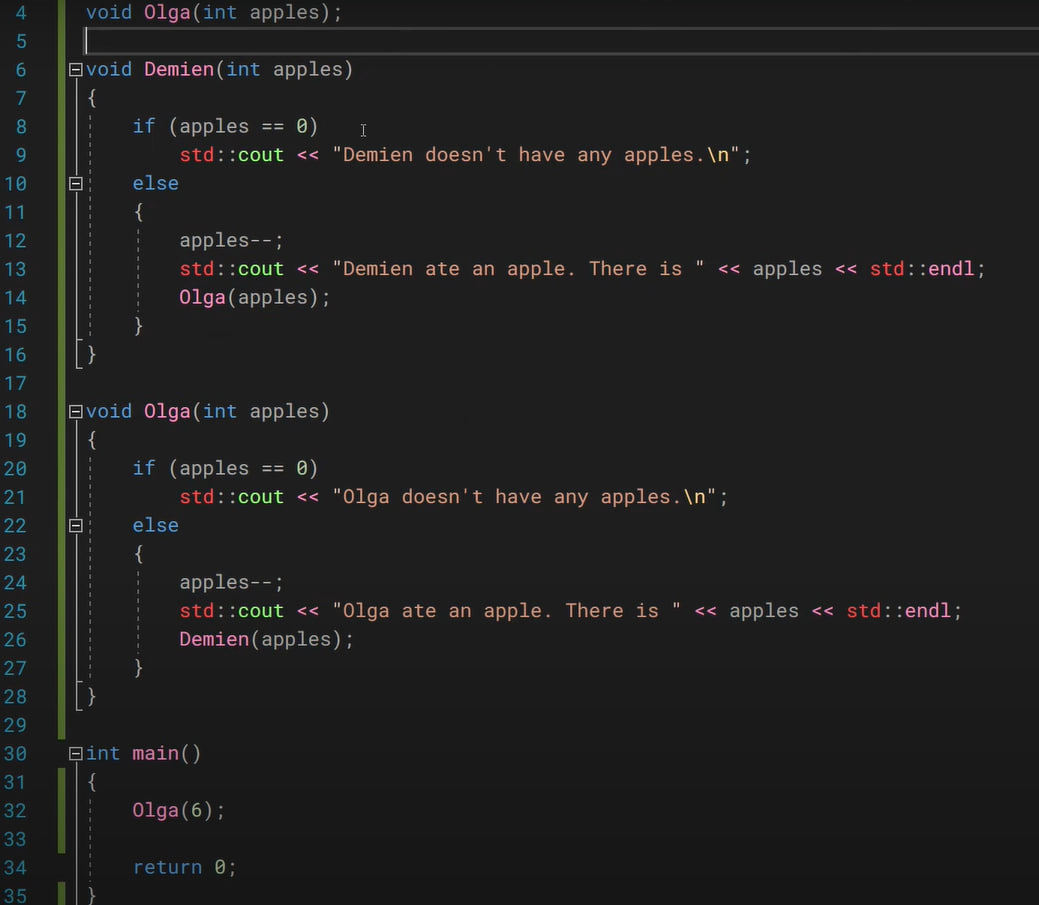


Рисунок 2 приклад складної рекурсії

* + Статус: Ознайомлена
  + Початок опрацювання теми: 22.11.23
  + Звершення опрацювання теми: 07.11.23
* Тема №4: вказівники
  + Джерела Інформації
  + <https://drive.google.com/drive/u/0/folders/1RteNrVnAsO5jXanknOc2T7LesLT7Vspx>
  + <https://www.youtube.com/watch?v=zopWRlYOXWw>
  + Що опрацьовано: Вираз &a дозволяє отримати адресу ділянки пам'яті, виділеного змінній а.

Операція & застосовна тільки до об'єктів, які мають ім'я й розташовані у

пам'яті.

Ми можемо ініціалізовувати вказівник на константу, але не можемо якщо у нас є константа ми не можемо зробити вказівник на змінну і присвоїти йому адресу константи.

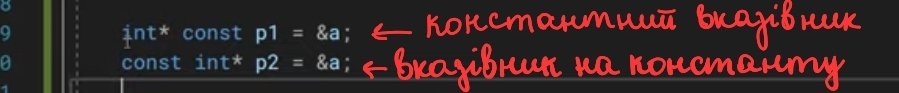


Рисунок 3 константний вказівник/вказівник на константу

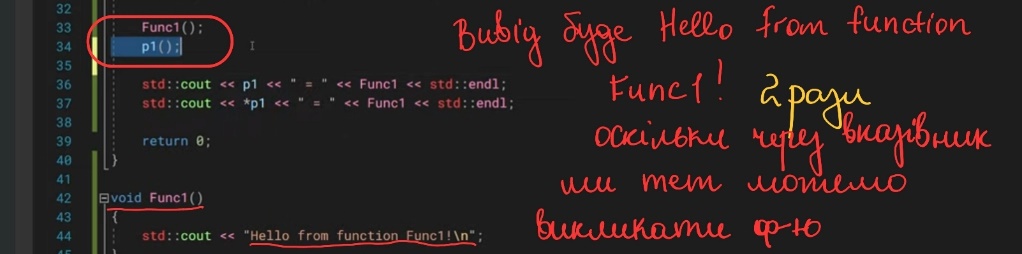


Рисунок 4 виклик функції через вказівник

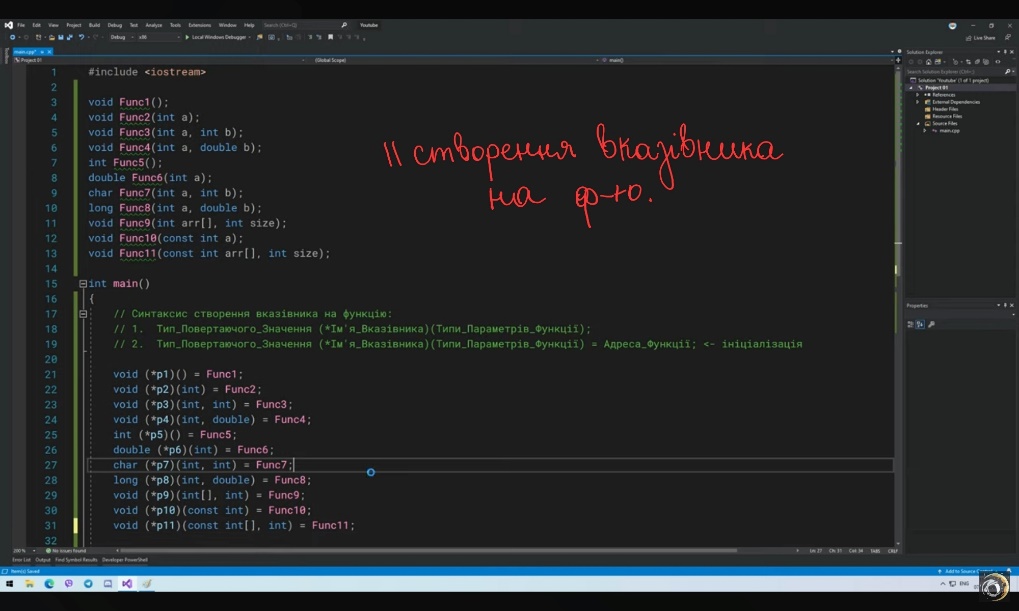


Рисунок 5 створення вказівника на різні види функцій

Нуль-вказівник: int\* a= 0\NULL\nullptr; (тут може бути тільки нуль, якщо не нуль, буде видавати помилку)

Ми можемо робити вказівник на вказівник.

Адреса (ім’я) масиву-адреса його першого елементу.

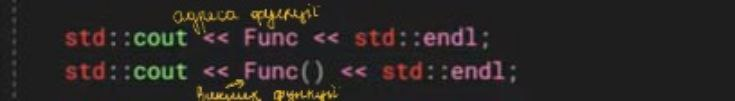


Рисунок 6 адреса/виклик ф-ї

\* - це операція розіменування. Операндом цієї операції завжди є вказівник.

* + Статус: Ознайомлена
  + Початок опрацювання теми: 19.11.23
  + Звершення опрацювання теми: 07.12.23
  + Статус: Ознайомлена
  + Початок опрацювання теми: 19.11.23
  + Звершення опрацювання теми: 07.12.23

# **Виконання роботи:**

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

Завдання №1 Опрацювання теорії

* Деталі завдання: опрацювання різноманітних матеріалів, аналіз та дослідження відео та статей на задані теми. Вивчення найважливіших моментів.

Завдання № 2 VNS Lab 4 - Task 1

* Варіант завдання: 5

- Деталі завдання:

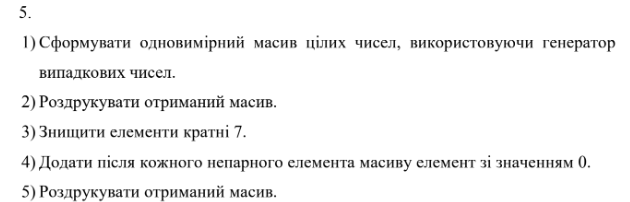


Рисунок 7 деталі завдання

* Важливі деталі для врахування в імплементації програми: використання функцій, також використано генерацію випадково числа за поточним часом, що робить результат ще більш рандомним.

Завдання № 3 VNS Lab 5 - Task 1

* Варіант завдання: 21
* Деталі завдання:



Рисунок 8 деталі завдання

Важливі деталі для врахування в імплементації програми: знання двовимірних масивів, вміння з ними працювати.

* Завдання №4 Algotester Lab 2
* Варіант завдання: 1
* Деталі завдання:

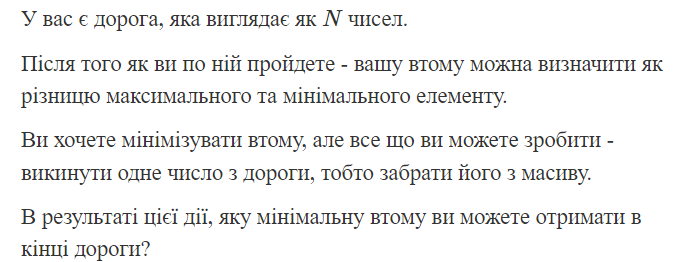


Рисунок 9 деталі завдання

* Важливі деталі для врахування в імплементації програми: врахування обмежень на алготестері
* Завдання №4 Algotester Lab 3
* Варіант завдання: 2
* Деталі завдання:

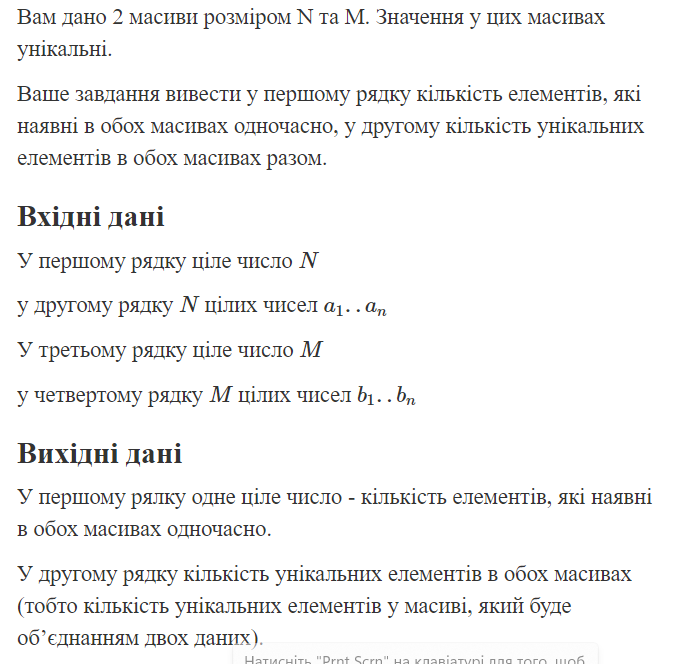


Рисунок 10 деталі завдання

* Важливі деталі для врахування в імплементації програми: врахування обмежень на алготестері

Завдання №5 Class Practice Task

* Деталі завдання: Реалізувати програму, яка перевіряє, чи дане слово чи число є паліндромом за допомогою рекурсії.
* Важливі деталі для врахування в імплементації програми: Паліндром — це послідовність символів, яка однаково читається вперед і назад (наприклад, «radar», «level», «12321»).
* Завдання №5 Self Practice Task
* Деталі завдання:

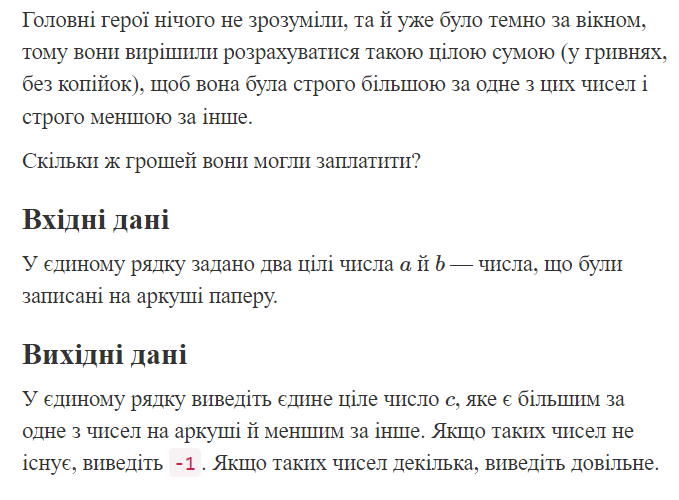


Рисунок 11 деталі завдання

* Важливі деталі для врахування в імплементації програми: врахування обмежень на алготестері
* Завдання №6 Self Practice Task 2
* Деталі завдання:

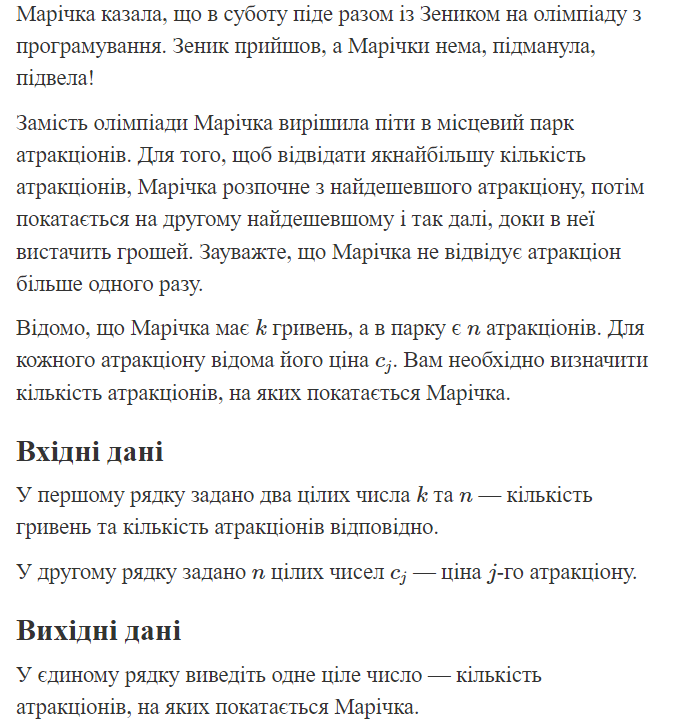


Рисунок 12 деталі завдання

* Важливі деталі для врахування в імплементації програми: врахування обмежень на алготестері

## **2. Дизайн та планована оцінка часу виконання завдань:**

Програма №5 Self Practice Task

* Блок-схема

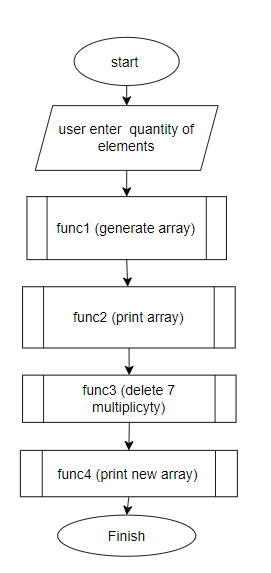


Рисунок 13 draw io

* Планований час на реалізацію: 4 години
* Важливі деталі для врахування в імплементації: використання циклу while (true)

## **3. Конфігурація середовища до виконання завдань:**

## **4. Код програм з посиланням на зовнішні ресурси:**

Завдання № 2 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

// #include <iostream>

#include <cstdlib>

#include <ctime>

#include <vector>

using namespace std;

// Ініціалізація масиву з випадкових чисел, довжина задана користувачем

void GetArray(int a[], int arrSize) {

    // Ініціалізація генератора випадкових чисел за поточним часом

    srand(time(0));

    // Генерація випадкових чисел для масиву

    for (int i = 0; i < arrSize; ++i) {

        a[i] = rand() % 977; // Генерація чисел від 0 до 976

    }

}

void PrintArray (const int a[], int arrSize){

    for (int i; i< arrSize; i++){

        cout << a[i] << " ";

    }

}

    //створюємо новий вектор на основі старого, ігноруючі числа кратні 7-ми

vector<int> Delete7Multiplicity(vector<int>& myVector)

{

    vector<int> newVector;

    for(int i = 0; i < myVector.size(); i++)

    {

        if(myVector[i] % 7 != 0)

            newVector.push\_back(myVector[i]);

    }

    return newVector;

}

void Print7Multiplicity(vector<int>& myVector){

cout << "Vector after deleting: ";

//змінній нам присвоюється при кожній ітерації елемент вектора, з яким потім виконується умова

    for (int num : myVector) {

        cout << num << " ";

    }

    cout << endl;

}

int main() {

    unsigned int arrSize;

    int a[100];

    cout << "Enter number of elements in array (from 1 to 100): ";

    cin >> arrSize;

    GetArray (a, arrSize);

    cout << "Your array: ";

    PrintArray(a, arrSize);

    cout << endl;

    vector<int> myVector(a, a + arrSize);

    // Викликаємо функцію для видалення елементів, які діляться на 7

    myVector = Delete7Multiplicity(myVector);

    // Викликаємо функцію для виведення елементів після видалення

    Print7Multiplicity(myVector);

return 0;

}

## Завдання № 3 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <vector>

using namespace std;

bool CheckIfNumberOccurs(vector<int> arr, int num)

{

    /\*тут треба написати код який будет як раз шукати\*/

    for (int i = 0; i < arr.size(); i++)

    {

        if(arr[i] == num)

            return true;

    }

    return false;

}

int main()

{

    int arr[3][3] = {

        {1,3,3},

        {5,6,7},

        {7,8,9}

        };

    cout << "Our array: " << endl ;

    //вивід вихідного масиву (для того щоб на простому показати як працювати з кожним ел-ом 2-го масиву)

    for(int i = 0; i < 3; i++)

    {

        for(int j = 0; j < 3; j++)

        {

            cout << arr[i][j] << " ";

        }

        cout << endl;

    }

    vector<int> temp, finalArray;

    for(int i = 0; i < 3; i++)

    {

        for(int j = 0; j < 3; j++)

        {

            if(CheckIfNumberOccurs(temp, arr[i][j]))

                finalArray.push\_back(arr[i][j]);

            temp.push\_back(arr[i][j]);

        }

    }

    cout << "Numbers that occure few times: ";

    for(int i = 0; i < finalArray.size(); i++){

        cout << finalArray[i] << " " ;}

        cout << endl;

        int max=finalArray[1];

    for (int i = 0; i < finalArray.size(); i++)

    {

        if(finalArray[i]>max){

            max=finalArray[i];

        }

        cout << "Max of numbers that occure few times:" << max;

    return 0;

}}

Завдання № 4 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

using namespace std;

int main (){

    int m;

    long long N;

    cin >> N;

    long long r [N];

    for(int i = 0; i < N; i++)

     {

        cin >> r[i];

     }

    for (int i= 0; i< N; i++)

    {

        for (int  j=0; j< N-1-i; j++)

        {

            if (r[j]>r[j+1])

            {

                int temp=r[j];

                r[j]=r[j+1];

                r[j+1]= temp;

            }

        }

    }

    if (N==1)

    {

        cout << 0;

    }

    else if (N==2){

        cout << r[N-1]-r[0];

    }

    else if (r[N-2]-r[0]<r[N-1]-r[1])

    {

    m= r[N-2]-r[0];

    cout << m;

    }

    else if (r[N-2]-r[0]>=r[N-1]-r[1])

    {

    m=(r[N-1]-r[1]);

    cout << m;

    }

    return 0;

}

## Завдання № 5 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

//

# include <iostream>

using namespace std;

int main (){

    unsigned int N, M;

    cin >> N;

    long long nums1[N];

    for (int i=0; i<N; i++){

        cin >> nums1[i];

    }

    cin >> M;

    long long nums2[M];

    for (int i=0; i<M; i++){

        cin >> nums2[i];

    }

    long long count=0;

    for (int i=0; i<N; i++){

        for (int j=0; j<M; j++){

        if (nums1[i]==nums2[j]){

            count++;

        }}}

    cout << count << endl;

    cout <<N+M-count;

return 0;

}

}

Завдання № 6 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

#include <iostream>

#include <string>

using namespace std;

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

bool isPalindrome(const int& number);

int main() {

    string input;

    cout << "Enter a word or number to check if it's a palindrome: ";

    cin >> input;

    // Перевірка, чи введено слово або число

    //isdigit-перевірка чи введення є числом

    if (isdigit(input[0])) {

        //stoi-перетворення стрінги в число

        int number = stoi(input);

        if (isPalindrome(number)) {

            cout << "Word or number is a palindrome";

        } else {

            cout << "Word or number is not a palindrome";

        }

    } else {

        if (isPalindrome(input, 0, input.length() - 1)) {

            cout << "Word or number is a palindrome";

        } else {

            cout << "Word or number is not a palindrome";

        }

    }

    return 0;

}

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(const int& number) {

    string strNumber = to\_string(number);

    int start = 0;

    int end = strNumber.length() - 1;

    while (start <= end) {

        if (strNumber[start] != strNumber[end]) {

            return false;

        }

        start++;

        end--;

    }

    return true;

}

Завдання № 7 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

//

# include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main (){

    int a,b;

    cin >> a;

    cin >> b;

        if (a==b){

        return 0;

    }

    if (a > b) {

         srand(time(nullptr));

    int random\_number = b + 1 + rand() % (a - b - 1);

        a-b > 1 ? cout << random\_number : cout << "-1";

    }

    if (b > a){

         srand(time(nullptr));

    int random\_number = a + 1 + rand() % (b - a - 1);

        b-a > 1 ? cout << random\_number : cout << "-1";

    }

return 0;

}

Завдання № 8 Деталі по програмі + Вставка з кодом з підписами до вставки. Посилання на файл програми у пул-запиті GitHub

//веселі каруселі

# include <iostream>

using namespace std;

int main (){

       int k, n, quan=0;

       cin >> k;

       cin >> n;

    int c[n];

    for (int i=0; i<n; i++)

    {

        cin >> c [i];

    }

      for (int i= 0; i< n; i++)

  {

      for (int  j=0; j< n-1-i; j++)

      {

          if (c[j]>c[j+1])

          {

             int temp=c[j];

             c[j]=c[j+1];

             c[j+1]= temp;

          }

      }

  }

    for (int j=0; j<n; j++){

        if (k < c[j]){

           break;

        }

        k=k-c[j];

        quan ++;

    }

    cout << quan;

return 0;

}

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

Завдання № 1 VNS lab 4 task 1

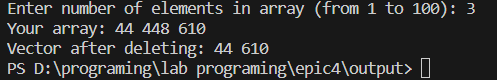
Деталі по виконанню і тестуванню програми 

Рисунок 12 тест 1

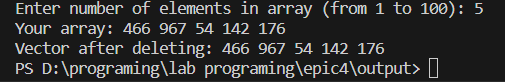


Рисунок 13 тест 2

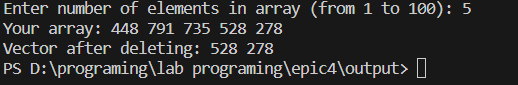


Рисунок 14 тест 3

Час затрачений на виконання завдання: 6 годин

Завдання № 2 VNS lab 5 task 1

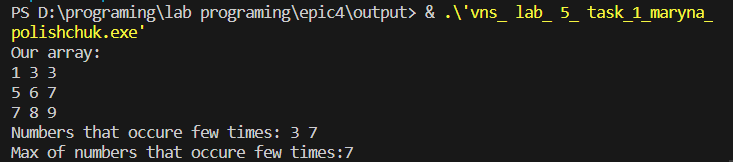


Рисунок 15 тест 1

Час затрачений на виконання завдання: 7 годин

Завдання № 3 Algotester Lab 2v1

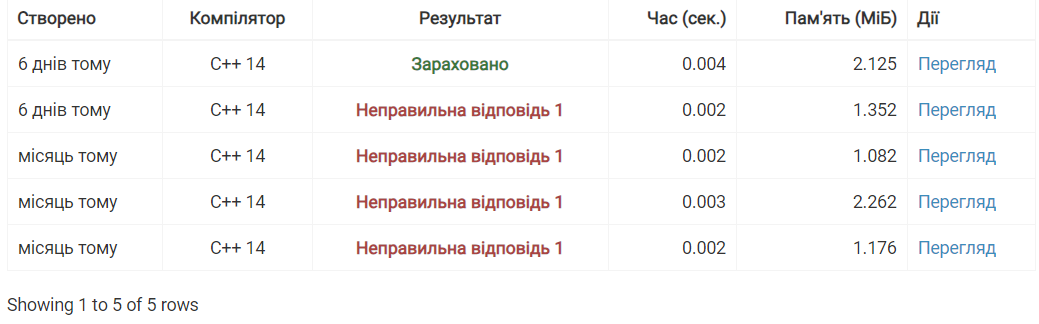
Деталі по виконанню і тестуванню програми 

Рисунок 16 зараховано на алготестері



Рисунок 17 тест 1



Рисунок 18 тест 2



Рисунок 19 тест 3

Час затрачений на виконання завдання: 12 годин

Завдання № 4 Algotester Lab 3v2

Деталі по виконанню і тестуванню програми



Рисунок 20 тест 1



Рисунок 21 тест 2

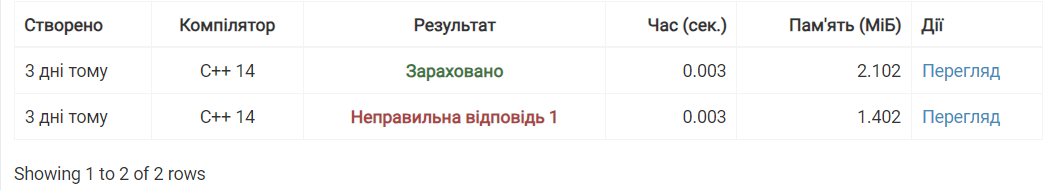


Рисунок 22 зараховано на алготестері

Час затрачений на виконання завдання: 4 години

Завдання № 5 Class Practice Task

Деталі по виконанню і тестуванню програми

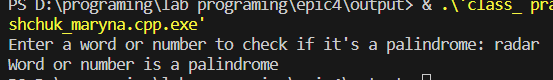


Рисунок 23 ntcn 1



Рисунок 24 тест 2



Рисунок 25 тест 3

Час затрачений на виконання завдання: 9 годин

Завдання №6 Self Practice Task 1

Деталі по виконанню і тестуванню програми



Рисунок 26 тест 1

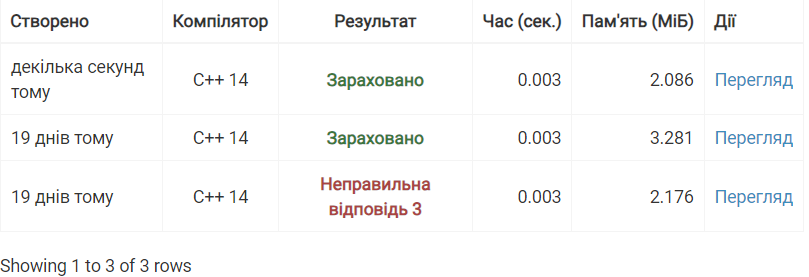


Рисунок 27 алготестер прийняв

Час затрачений на виконання завдання: 5 годин

Завдання №7 Self Practice Task 2

Деталі по виконанню і тестуванню програми

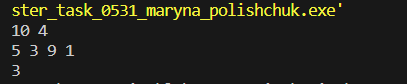


Рисунок 28 тест 1

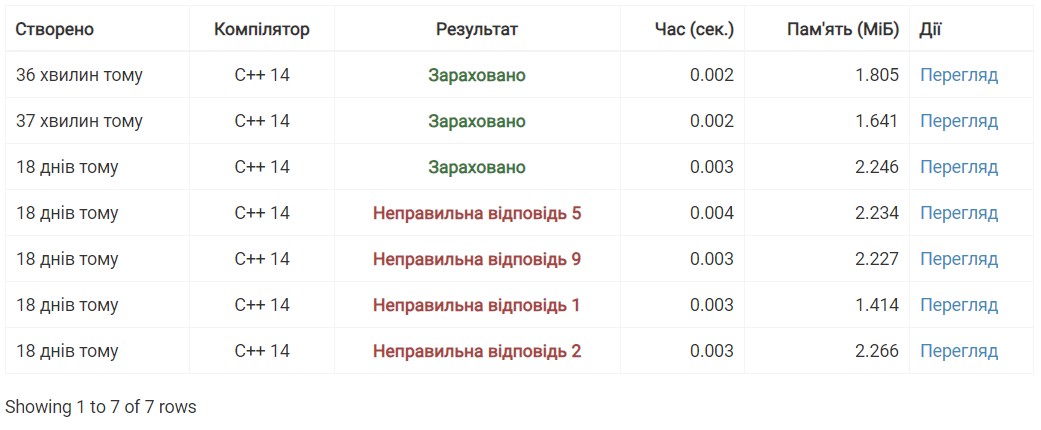


Рисунок 29 алготестер прийняв

Час затрачений на виконання завдання: 3 години

# **Висновки:**

# Ми навчилися працювати з простими структурами даних, одновимірними\двовимірними масивами.