Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**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

Мостова Віта Любомирівна

# **Тема роботи:** Прості структури даних. Одновимірні масиви. Двовимірні масиви. Алгоритми обробки

# **Мета роботи:** Навчитись використовувати прості структури даних. Розібратись в роботі з одно- та двовимірними масивами. Використовувати алгоритми обробки.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Прості структури даних
* Тема №2: Одновимірні масиви
* Тема №3: Двовимірні масиви
* Тема №4: Алгоритми обробки масивів

1. Індивідуальний план опрацювання теорії:

* Тема №1: Прості структури даних
  + Джерела Інформації
    - <https://ua5.org/struktury-danyh/1623-klasyfikacziya-struktur-danyh.html>
    - [http://dspace.wunu.edu.ua/bitstream/316497/24160/1/fkit\_kn\_pzs\_asd\_LEK.pdf](http://dspace.wunu.edu.ua/bitstream/316497/24160/1/fkit_kn_pzs_asd_LEK.pdf%20)
  + Що опрацьовано: Опрацьовано матеріали надані вище. Розібрано, що таке прості структури даних
  + Статус: Ознайомлена
  + Початок опрацювання теми: 1.12.2023
  + Звершення опрацювання теми: 6.12.2023
* Тема №2: Одновимірні масиви.
  + Джерела Інформації:
    - [http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/65\_C++/index.html](http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/65_C++/index.html%20)
    - [https://acode.com.ua/urok-77-masyvy/](https://acode.com.ua/urok-77-masyvy/%20)
    - <https://www.youtube.com/watch?v=ULdbOaMBPYc>
  + Що опрацьовано: Опрацьовано матеріали надані вище. Розібрано як створювати одновимірні масиви.
  + Статус: Ознайомлена
  + Початок опрацювання теми: 29.11.2023
  + Звершення опрацювання теми: 6.12.2023
* Тема №3: Двовимірні масиви
  + Джерела Інформації:
    - <https://acode.com.ua/urok-81-bagatovymirni-masyvy/>
    - <https://cherto4ka.xyz/2020/02/08/%D0%B1%D0%B0%D0%B7%D0%BE%D0%B2%D1%96-%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D1%97-%D0%BE%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B8-%D0%B4%D0%B2%D0%BE%D0%B2%D0%B8%D0%BC%D1%96%D1%80%D0%BD%D0%B8%D1%85/>
  + Що опрацьовано: : Опрацьовано матеріали надані вище. Розібрано як створювати двовимірні масиви.
  + Статус: Ознайомлена
  + Початок опрацювання теми: 30.11.2023
  + Звершення опрацювання теми: 6.12.2023
* Тема №4 Алгоритми обробки масивів
  + Джерела Інформації:
    - [https://studfile.net/preview/9287302/page:31/](https://studfile.net/preview/9287302/page:31/%20)
  + Що опрацьовано: Опрацьовано матеріали надані вище. Розібрано типові алгоритми обробки масивів
  + Статус: Ознайомлена
  + Початок опрацювання теми: 30.11.2023
  + Звершення опрацювання теми: 6.12.2023

# **Виконання роботи:**

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

Завдання №1 ***«Однонапрямлене кільце»***

* Варіант 18
* Реалізувати однонаправлене кільце. Роздрукувати починаючи з К-ого елемента і до К-1. Додати в кільце перший і останній елементи. Знищити з кільця парні елементи. Роздрукувати отримане кільце.

Завдання №2 ***«Послідовність спадна?»***

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

Завдання №3 ***«Дрони»***

* Дано масив цiлих чисел розмiром N, на першiй та останнiй клiтинцi розмiщено по дрону.

Вони одночасно взлiтають. На початку кожного ходу швидкiсть дрону стає рiвною значенню клiтинки, у якiй вiн знаходиться. Дрони рухаються до моменту, коли трапиться одна з зазначених подiй:

* дрони опиняються в однiй клiтинцi – програма виводить «Collision»
* лiвий дрон опиниться справа вiд правого - «Miss»
* дрони зупиняться один навпроти одного – «Stopped»

Завдання №4 ***«Компресія стрічки»***

* Дана стрічка s. Потрібно зробити компресію стрічки, тобто якщо якась буква йде більше одного разу підряд у стрічці замінити її на букву + кількість входжень підряд.
* Великі та маленькі букви вважаються різними

Завдання №5 ***«Перевірка чи слово або число є паліндромом»***

* Реалізувати програму, яка перевіряє, чи дане слово чи число є паліндромом за допомогою рекурсії. Для цього визначити та реалізувати рекурсивну функцію isPalindrome для рядків та перевантажену функцію isPalindrome для цілих чисел.
* Перетворити ціле число на рядок і використати рядкову версію функції, щоб перевірити, чи це паліндром.

Завдання №6 ***«Найбільша зростаюча послідовність»***

* Задано послідовність із n цілих чисел ai. Потрібно знайти довжину найбільшої зростаючої підпослідовності заданої послідовності.

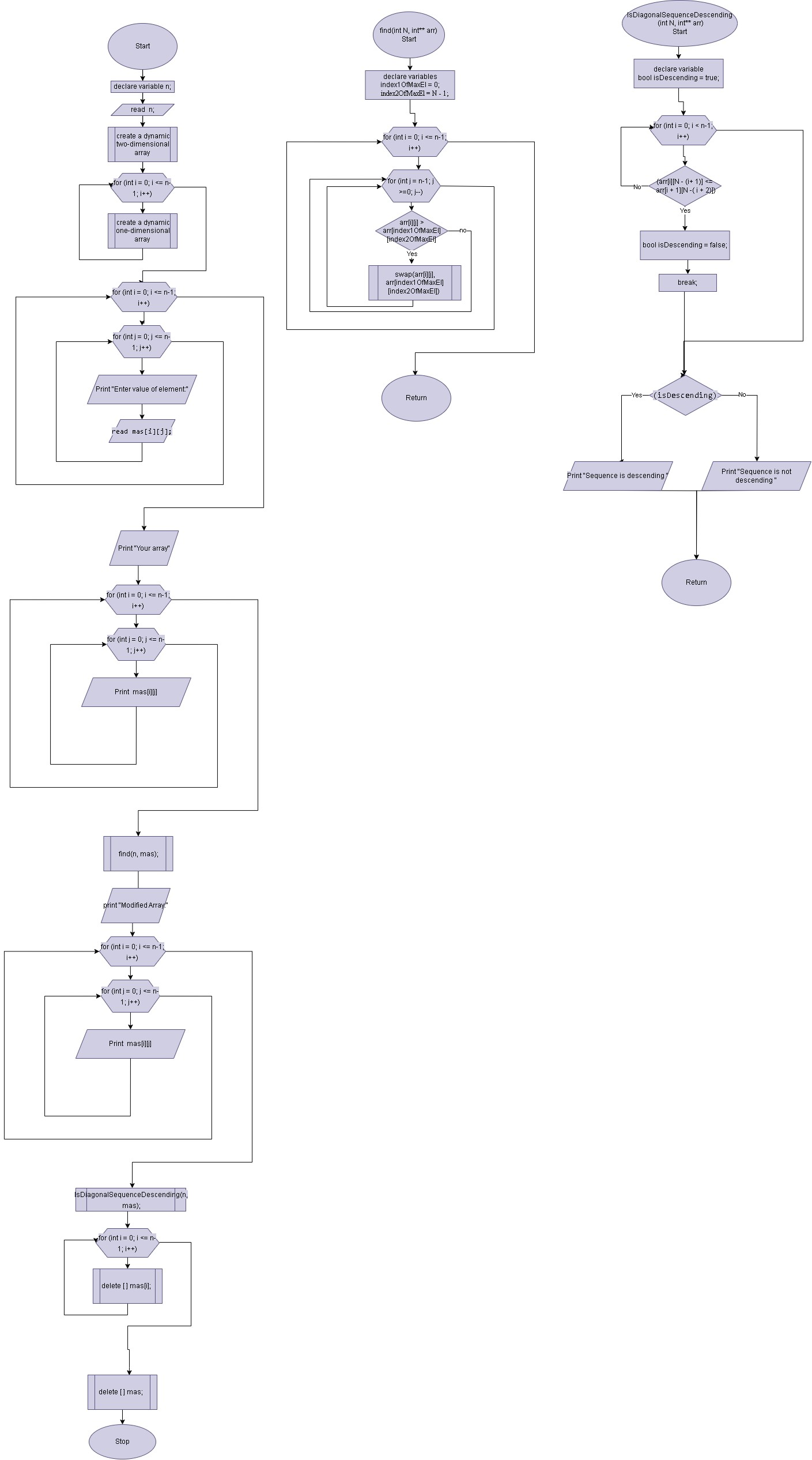
## **2. Дизайн та планована оцінка часу виконання завдань:**

Програма №1 ***«Однонапрямлене кільце»***

* Планований час на реалізацію: до 3 днів

Програма №2 ***«Послідовність спадна?»***

* Планований час на реалізацію: до 4 годин



*Блок-схема до завдання №2*

Програма №3 ***«Дрони»***

* Планований час на реалізацію: до 2 годин

Програма №4 ***«Компресія стрічки»***

* Планований час на реалізацію: до 2 годин

Програма №5 ***«Перевірка чи слово або число є паліндромом»***

* Планований час на реалізацію: 3 години

Програма №6 ***«Найбільша зростаюча послідовність»***

* Планований час на реалізацію: 2 години

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

Завдання №1 ***«Однонапрямлене кільце»***

[Pull-request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/716/files%23diff-31628ba25b1078b7fac0eee6d80cb9bc817bc06bd78fdcfb60d35eb0c07dacf0)

#include <iostream>

using namespace std;

struct Node

{

    int data;

    Node \*next;

};

class LinkedList

{

private:

    Node \*head;

    Node \*end;

public:

    LinkedList()

    {

        head = NULL;

        end = NULL;

    }

    void insertStart(int data)

    {

        Node \*newNode = new Node();

        newNode->data = data;

        if (head == NULL)

        {

            newNode->next = newNode;

            head = newNode;

            end = newNode;

        }

        else

        {

            newNode->next = head;

            end->next = newNode;

            head = newNode;

        }

    }

void insertLast(int data)

{

    Node \*newNode = new Node();

    newNode->data = data;

    if (head == NULL)

    {

        newNode->next = newNode;

        head = newNode;

        end = newNode;

    }

    else

    {

        end->next = newNode;

        end = newNode;

        end->next = head;

}

    void display()

    {

        if (head == NULL)

        {

            cout << "List is empty \n";

            return;

        }

        Node \*node = head;

        do

        {

            cout << node->data << " ";

            node = node->next;

        } while (node != head);

        cout <<"\n";

    }

    void displayFromK(int k)

    {

        if (head == NULL)

        {

            cout << "List is empty \n";

            return;

        }

        Node \*firstNode = head;

        for (int i = 0; i < k; ++i){

            firstNode = firstNode->next;

            if (firstNode == head)

            {

                display();

                return;

            }

        }

        Node \*node = firstNode;

        do

        {

            cout << node->data << " ";

            node = node->next;

        } while (node != head);

        node = head;

        while (node != firstNode)

        {

            cout << node->data << " ";

            node = node->next;

        }

        cout << "\n";

    }

    void deleteNode(Node \*del)

    {

        if (del == head)

        {

            if (head->next == head)

            {

                delete del;

                head = NULL;

                end = NULL;

                return;

            }

            head = head->next;

            end->next = head;

        }

        else

        {

            Node \*temp = head;

            while (temp->next != head && temp->next != del)

            {

                temp = temp->next;

            }

            if (temp->next == head)

            {

                return;

            }

            temp->next = del->next;

            if (del == end){

                end = temp;}

        }

        delete del;

    }

    void deleteEvenNodes(){

        Node \*ptr = head;

        while (ptr != NULL){

            Node \*next = ptr->next;

            if (ptr->data % 2 == 0) {

                deleteNode(ptr); }

            ptr = next;

            if (ptr == head){

                break;}

        }

    }

};

int main()

{

    LinkedList \*list = new LinkedList();

    list->insertStart(5);

    list->insertStart(10);

    list->insertStart(15);

    list->insertStart(20);

    list->insertStart(25);

    list->insertStart(30);

    list->insertStart(35);

    cout << "Original List: ";

    list->display();

    cout << "Original List from K to (K-1): ";

    list->displayFromK(3);

    cout << "List after deleting even nodes: ";

    list->deleteEvenNodes();

    list->displayFromK(2);

    list->insertLast(40);

    list->insertLast(45);

    list->insertStart(6);

    cout << "List after appending elements to the end: ";

    list->display();

    return 0;

}

*Програмний код до завдання №1*

Завдання №2 ***«Послідовність спадна?»***

[Pull-request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/716/files%23diff-4768918d7957f5dafc3b54cd86b38a9f97025183148eebcebf15e978a68a22d0)

#include <iostream>

using namespace std;

void find(int N, int\*\* arr);

void IsDiagonalSequenceDescending (int N, int\*\* arr);

int main() {

    int n;

    cout << "Enter n:\n ";

    cin >> n;

    int\*\* mas = new int\*[n];

    for (int i = 0; i <= n-1; i++) {

        mas[i] = new int[n];

    }

    for (int i = 0; i <= n-1; i++) {

        for (int j = 0; j <= n-1; j++) {

            cout << "Enter value of element[" << i+1 << "][" << j+1 << "]: \n";

            cin >> mas[i][j];

        }

    }

    cout << "Your Array:\n";

    for (int i = 0; i <= n-1; i++) {

        for (int j = 0; j <= n-1; j++) {

            cout << mas[i][j] << " ";

        }

        cout <<"\n";

    }

    find(n, mas);

    cout << "Modified Array:\n";

    for (int i = 0; i <= n-1; i++) {

        for (int j = 0; j <= n-1; j++) {

            cout << mas[i][j] << " ";

        }

        cout <<"\n";

    }

IsDiagonalSequenceDescending(n, mas);

    for (int i = 0; i <= n-1; i++) {

        delete[] mas[i];

    }

    delete[] mas;

    return 0;

}

void find(int N, int\*\* arr) {

    int index1OfMaxEl = 0;

    int index2OfMaxEl = N - 1;

    for (int i = 0; i <= N-1; ++i) {

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

            if (arr[i][j] > arr[index1OfMaxEl][index2OfMaxEl]) {

                swap(arr[i][j], arr[index1OfMaxEl][index2OfMaxEl]);

            }

        }

    }

}

void IsDiagonalSequenceDescending(int N, int\*\* arr) {

    bool isDescending = true;

    for (int i = 0; i < N - 1; ++i) {

        if (arr[i][N - (i+ 1)] < arr[i + 1][N -( i + 2)]) {

            isDescending = false;

            break;

        }

    }

    if (isDescending) {

        cout << "Sequence is descending " ;

    } else {

        cout << "Sequence is not descending ";

    }

}

*Програмний код до завдання №2*

Завдання №3 ***«Дрони»***

[Pull-request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/716/files%23diff-fa274c88dc864cf5af1043437d910273f354205847b1f21729f23ab4ad3891de)

#include <iostream>

using namespace std;

int main(){

    int N ;

    cin>>N;

    int speed1=0;

    int speed2=N-1;

    const int SIZE\_ARRAY = N;

    int arr1[SIZE\_ARRAY];

    for (int i=0;i<N;i++){

          cin>>arr1[i];

}

while ((speed1<=N-1)&&(speed2>=0)){

if(speed1==speed2){

    cout<<speed1+1<<" "<<speed2+1<<endl;

    cout<<"Collision";

    break;

}else if(speed1>speed2){

    cout<<speed1+1<<" "<<speed2+1<<endl;

    cout<<"Miss";

    break;

}else if(speed1==speed2-1){

    cout<<speed1+1<<" "<<speed2+1<<endl;

    cout<<"Stopped";

    break;

}

speed1+=arr1[speed1];

speed2-=arr1[speed2];

}

    return 0;

    }

*Програмний код до завдання №3*

Завдання №4 ***«Компресія стрічки»***

[Pull-request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/716/files%23diff-23b7b629caac8ff379b252d3abdd709c2e62112f162a3a30877a86fc300b7fa5)

#include <iostream>

using namespace std;

int main(){

string s;

string sCompressed;

int numOfRepetitions=1;

cin>>s;

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

  if(s[i]==s[i+1]){

numOfRepetitions++;

}else{

sCompressed+=s[i]+(numOfRepetitions>1?to\_string(numOfRepetitions):"");

numOfRepetitions=1;

}

}

cout<<sCompressed;

     return 0;

}

*Програмний код до завдання №4*

Завдання №5 ***«Перевірка чи слово або число є паліндромом»***

[Pull-request](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/716/files%23diff-e3b7e66fcf4bdcf5f05d8e8d5ff7bdf97ee02a846b117ee4780310d83119c07f)

#include <iostream>

#include <string>

using namespace std;

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

bool isPalindrome(int number);

int main()

{

string radar="radar";

string level="level";

string notPalindrome="orange";

cout<<"Is 'radar' palindrome? "<<(isPalindrome(radar,0,(radar.size()-1))?"Yes :)":"No :(")<<endl;

cout<<"Is 'level' palindrome? "<<(isPalindrome(level,0,(level.size()-1))?"Yes :)":"No :(")<<endl;

cout<<"Is 'orange' palindrome? "<<(isPalindrome(notPalindrome,0,(notPalindrome.size()-1))?"Yes :)":"No :(")<<endl;

cout<<"Is 564 palindrome? "<<(isPalindrome(564)?"Yes :)":"No :(")<<endl;

cout<<"Is 12321 palindrome? "<<(isPalindrome(12321)?"Yes :)":"No :(")<<endl;

    return 0;

}

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

    if(str[start]!=str[end]){

        return false;

    }

    if(start>=end){

        return true;

    }

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

}

bool isPalindrome(int number){

    string str\_number=to\_string(number);

    return isPalindrome(str\_number,0,(str\_number.size()-1));

}

Програмний код до завдання №5

Завдання №6 ***«Найбільша зростаюча послідовність»***

[**Pull-request**](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/716/files%23diff-a56692219fce8ce3ae179eafa2201f8dc2f7595d46a30dc23764a00f0179fb78)

#include <iostream>

#include <algorithm>

using namespace std;

int FindDecreasingSequence(int \*mas,int N);

int main(){

int n;

cin>>n;

int \*arr=new int[n];

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

    cin>>arr[i];

}

 cout<<FindDecreasingSequence(arr,n);

delete[] arr;

    return 0;

}

int FindDecreasingSequence(int \*mas,int N){

int f[N];

f[0]=1;

for(int i=1;i<N;i++){

    f[i]=1;

    for (int j=0;j<i;j++){

        if(mas[j]<mas[i]){

            f[i]= max(f[i],f[j]+1);

        }

    }

}

int length=1;

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

length = max(length, f[i]);

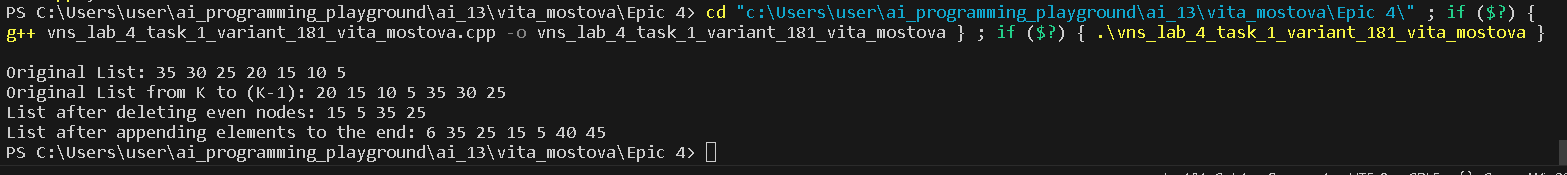
}

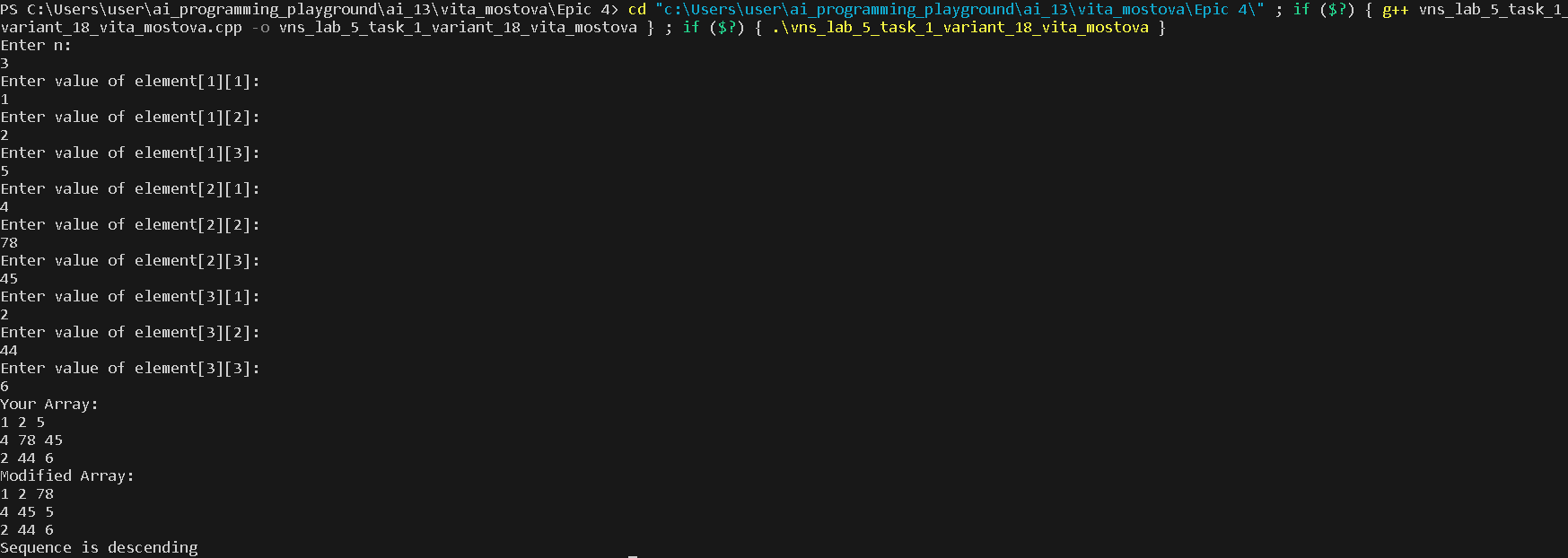
return length;}

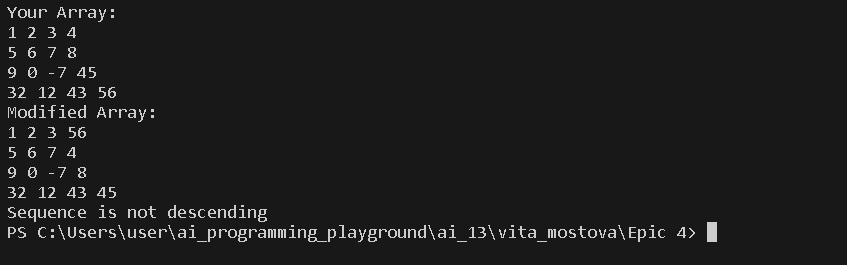
Програмний код до завдання №6

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

Завдання №1 ***«Однонапрямлене кільце»***

*Рисунок 1: Приклад виконання завдання №1*

Завдання №2 ***«Послідовність спадна?»****Рисунок 2.1: Приклад виконання завдання №2*

 *Рисунок 2.2: Приклад виконання завдання №2*

Час затрачений на виконання завдання: близько 5 годин

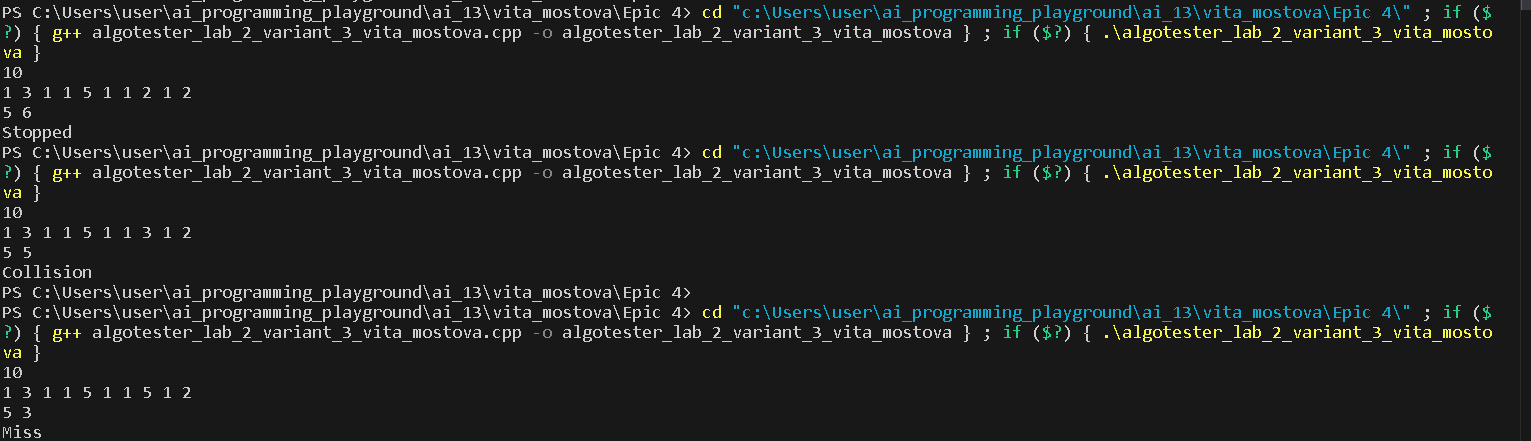
Завдання №3 ***«Дрони»***

Рисунок 3.1: Приклад виконання завдання №3

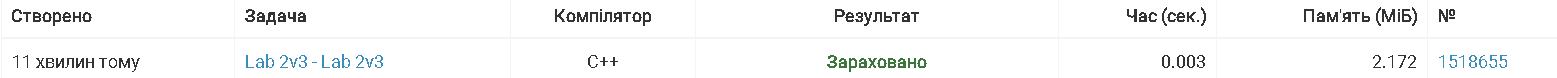
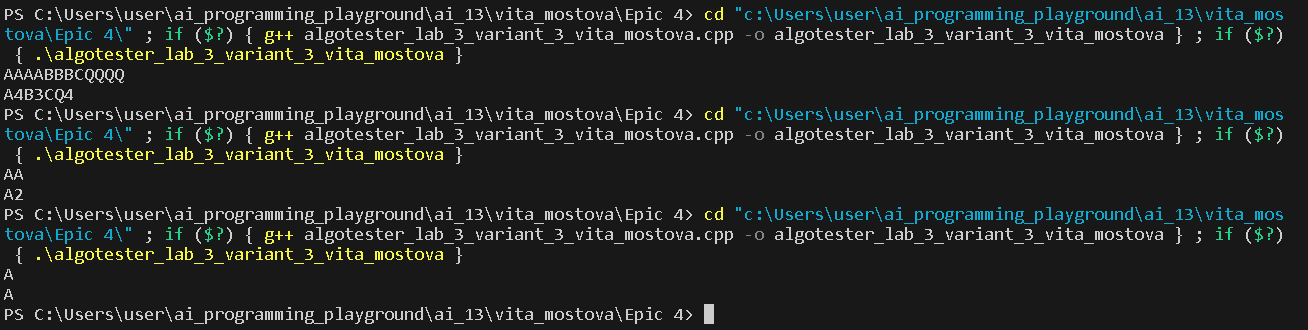


Рисунок 3.2: Скрін виконання завдання №3 на алготестері

Час затрачений на виконання завдання: близько 2 годин



*Рисунок 4.1: приклад виконання завдання №4*

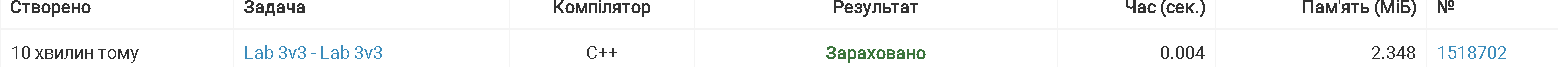
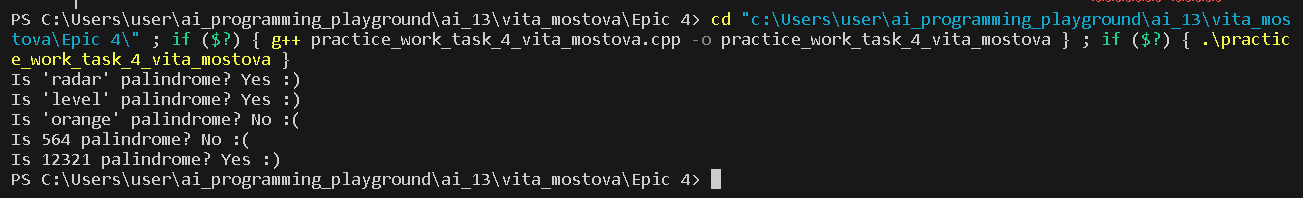


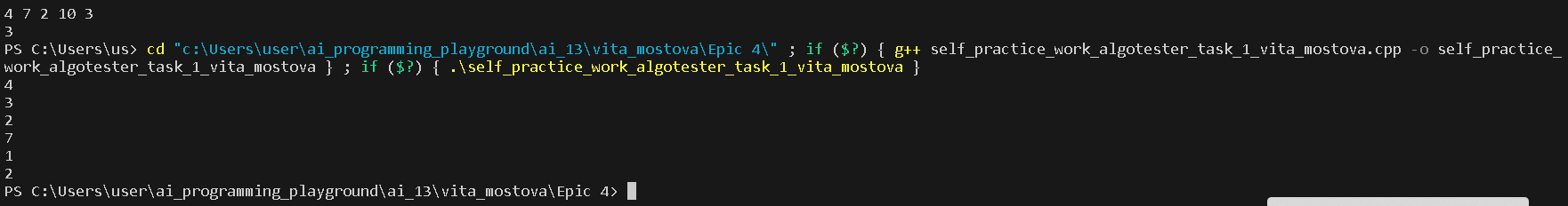
Рисунок 4.2: Скрін виконання завдання №4 на алготестері

Час затрачений на виконання завдання: близько 1 години

Завдання №5 ***«Перевірка чи слово або число є паліндромом»****Рисунок 5: приклад виконання завдання №5*

Час затрачений на виконання завдання: близько 2 годин

Завдання №6 ***«Найбільша зростаюча послідовність»***

*Рисунок 6.1: приклад виконання завдання №6*

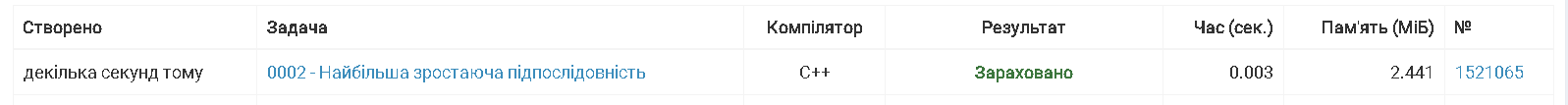


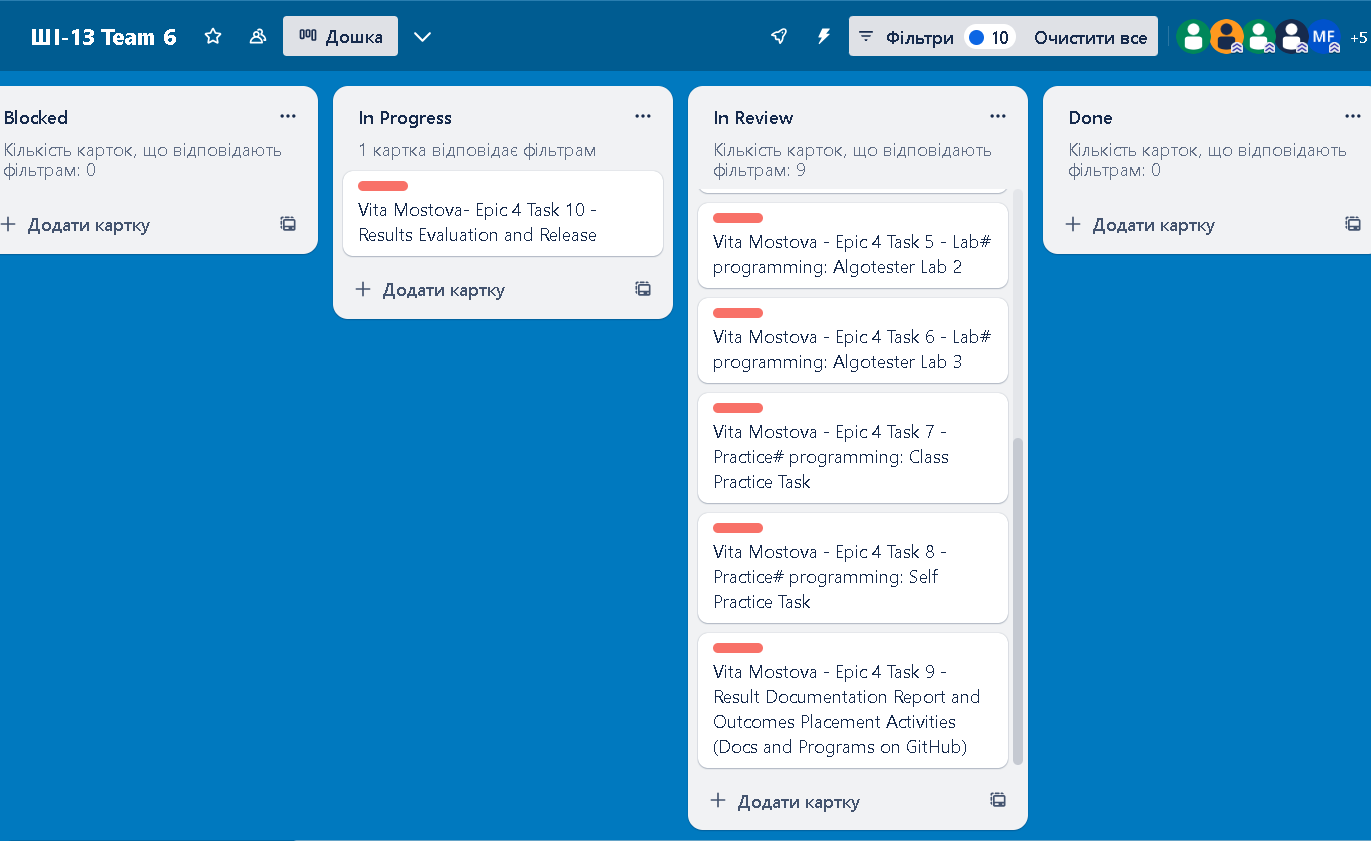
Рисунок 6.2: Скрін виконання завдання №6 на алготестері

Час затрачений на виконання завдання: близько 2 годин

## **5. Кооперація з командою:**

# 

Скрін з зустрічі з командою



Скрін з трелло

# **Висновки:**

Під час виконання завдань з цього епіка я попрактикувалась створювати одно- та двовимірні масиви( також змогла повзаємодіяти з динамічним створенням масивів). Досить складною для мене виявилось завдання з однонапрямленим кільцем, оскільки ніколи до того не використовувала структури і класи + потрібно було добре розібратись з вказівниками. З математичної точки зору було трішки незрозуміле друге завдання, проте після трішки довших роздумів все ж вдалось його реалізувати. Загалом, мені вдалось розібратись в роботі з масивами( а також масивами,як параметрами функції) та однонапрямленим кільцем.