Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 4, 5**

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

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

***Виконав:***

студент групи ШІ-11

Рєвков Володимир

# **Тема роботи:**

Ознайомлення з розділом «Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.» та виконання лабораторних робіт ВНС №4 та №5, лабораторних робіт з Алготестеру №2 та №3, практичної та самостійної практичної роботи.

# **Мета роботи:**

Ознайомитись з розділом «Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.» та виконати лабораторні роботи ВНС №4 та №5, лабораторні роботи з Алготестеру №2 та №3, практичну та самостійну практичну роботи.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Одновимірні масиви.
* Тема №2: Двовимірні Масиви.
* Тема №3: Вказівники та Посилання.
* Тема №4: Динамічні масиви.
* Тема №5: Структури даних.
* Тема №6: Вкладені структури.
* Тема №7: Алгоритми обробки та робота з масивами та структурами.

1. Індивідуальний план опрацювання теорії:

* Тема №1: Одновимірні масиви.
  + Джерела Інформації
    - Відео.

<https://youtu.be/T76E09hnEuo?feature=shared>

* + - Стаття.

<https://www.geeksforgeeks.org/cpp-arrays/>

* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023
* Тема №2: Двовимірні масиви.
  + Джерела Інформації
    - Відео.

<https://youtu.be/eVSbSeJPkDA?feature=shared>

* + - Стаття.

<https://www.geeksforgeeks.org/multidimensional-arrays-in-c/>

* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023
* Тема №3: Вказівники та Посилання.
  + Джерела Інформації
    - Відео.

<https://youtu.be/eNofmKYzje4?feature=shared>

* + - Стаття.

<https://www.geeksforgeeks.org/pointers-vs-references-cpp/>

* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023
* Тема №4: Динамічні масиви.
  + Джерела Інформації
    - Відео.

<https://youtu.be/axsplPtoQF0?feature=shared>

* + - Стаття.

<https://www.geeksforgeeks.org/how-do-dynamic-arrays-work/>

* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023
* Тема №5: Структури даних.
  + Джерела Інформації
    - Відео.

<https://youtu.be/9j5KkOC_goQ?feature=shared>

* + - Стаття.

<https://www.geeksforgeeks.org/structures-in-cpp/>

* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023
* Тема №6: Вкладені структури.
  + Джерела Інформації
    - Відео.

<https://youtu.be/9j5KkOC_goQ?feature=shared>

* + - Стаття.

<https://www.geeksforgeeks.org/nested-structure-in-c-with-examples/>

* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023
* Тема №7: Алгоритми обробки та робота з масивами та структурами.
  + Джерела Інформації
    - Відео.

<https://www.youtube.com/watch?v=Dv4qLJcxus8>

* + - Стаття.

<https://www.geeksforgeeks.org/useful-array-algorithms-in-c-stl/>you

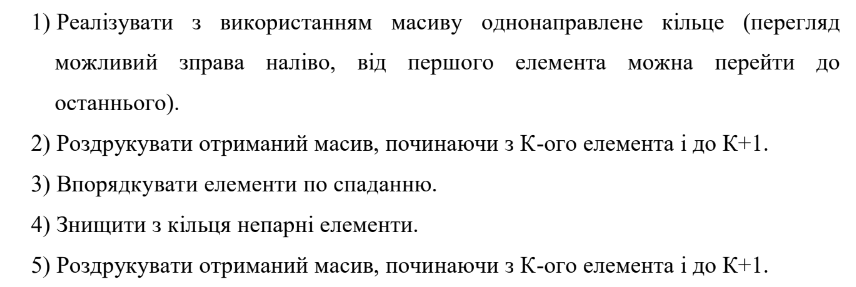
* + Що опрацьовано:
    - Опрацьовано вміст відео та статті.
  + Статус: Ознайомлений частково
  + Початок опрацювання теми: 01.12.2023
  + Звершення опрацювання теми: 01.12.2023

# **Виконання роботи:**

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

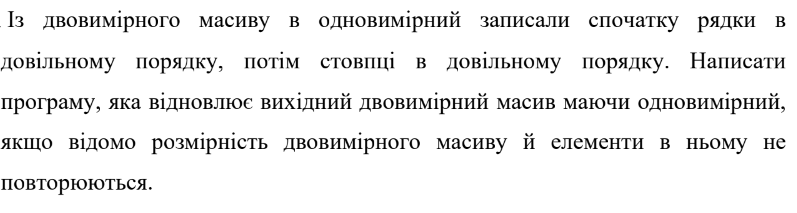
Завдання №1: Лабораторна робота №4

* Варіант завдання: 25
* Деталі завдання



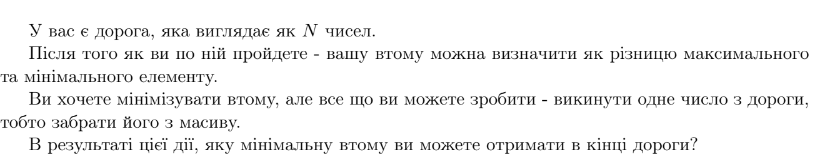
Завдання №2: Лабораторна робота №5

* Варіант завдання: 25
* Деталі завдання



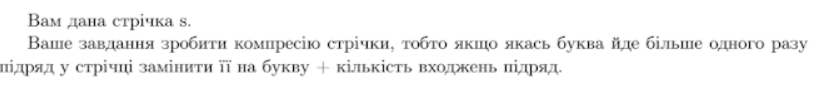
Завдання №3: Лабораторна робота з Алготестеру №2

* Варіант завдання: 1
* Деталі завдання



Завдання №4: Лабораторна робота з Алготестеру №3

* Варіант завдання: 1
* Деталі завдання



Завдання №5: Практична робота

* Варіант завдання: Паліндром
* Деталі завдання



Завдання №6: Самостійна практична робота

* Варіант завдання: Студентський журнал
* Деталі завдання

Реалізувати програму для додавання та видалення студентів до масиву, виводу студентів, їх оцінок, сортування за успішність, знаходження студента за іменем та прізвищем та встановлення оцінок.

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

Програма №1: Лабораторна робота №4

* Блок-схема

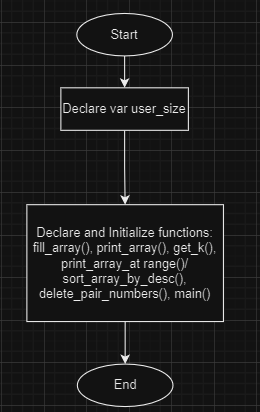
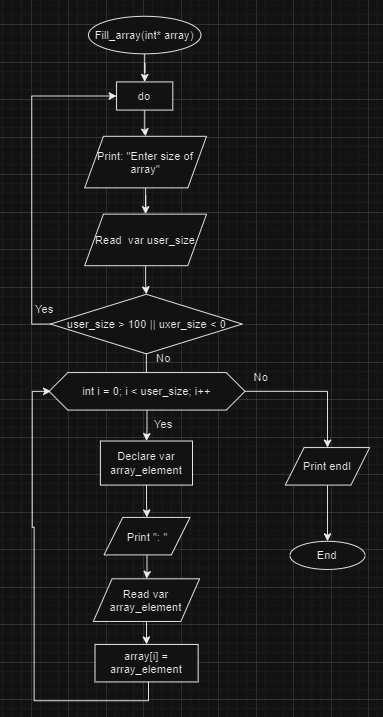
 

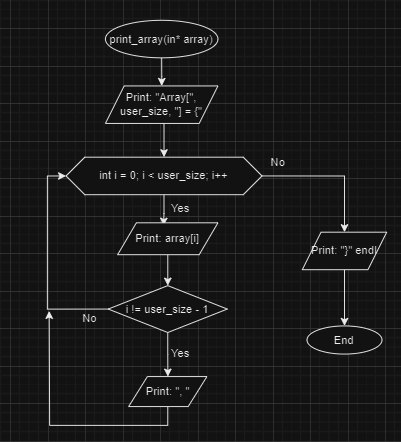
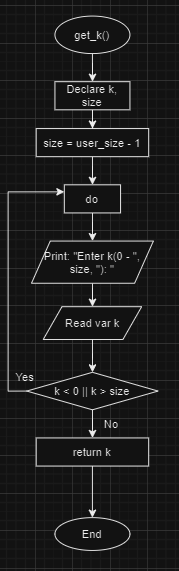
Рисунок 1: Main Program Рисунок 2: Function fill\_array 

Рисунок 3: Function print\_array Рисунок 4: Function get\_k

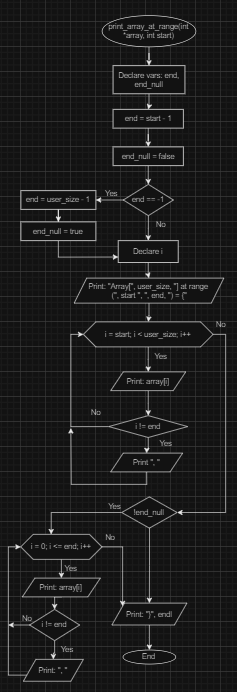
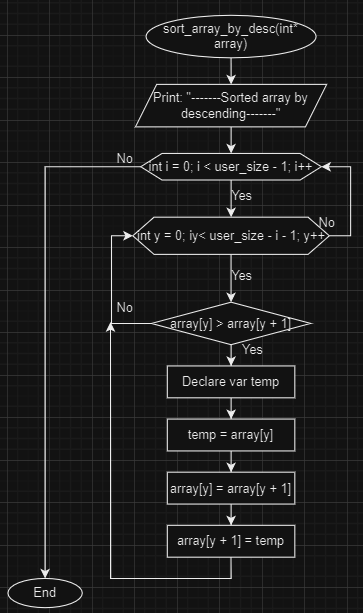
 

Рисунок 5: Function print\_array\_at\_range Рисунок 6: Function sort\_array\_by\_desc

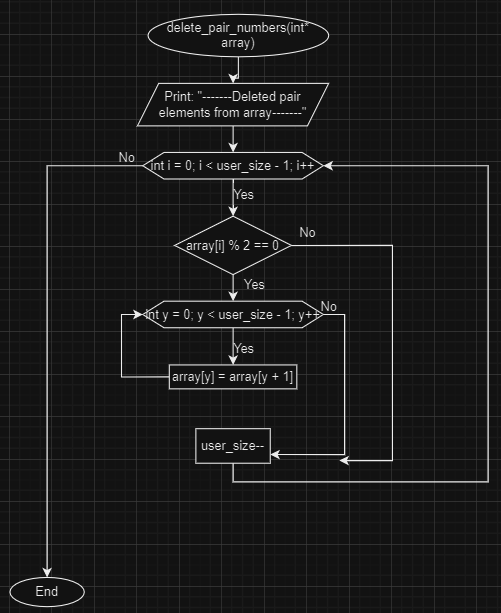
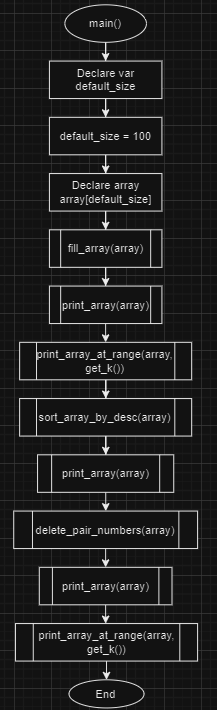
 

Рисунок 7: Function delete\_pair\_numbers Рисунок 8: Function main

* Планований час на реалізацію – 20хв.
* Реальний час на реалізацію – 90хв.
* Важливі деталі для врахування в імплементації:

Ця програма була обрана для побудови діаграми у зв’язку з тим, що здалась мені найбільш тяжкою з усіх.

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

Завдання №1: Лабораторна робота №4

#include<iostream>

using namespace std;

int user\_size;

void fill\_array(int\* array)

{

    do

    {

        cout << "Enter size of array: ";

        cin >> user\_size;

    } while (user\_size > 100 || user\_size < 0);

    for(int i = 0; i < user\_size; i++)

    {

        int array\_element;

        cout << ": ";

        cin >> array\_element;

        array[i] = array\_element;

    }

    cout << endl;

}

void print\_array(int\* array)

{

    cout << "Array[" << user\_size << "] = {";

    for(int i = 0; i < user\_size; i++)

    {

        cout << array[i];

        if(i != user\_size - 1)

        {

            cout << ", ";

        }

    }

    cout << "}" << endl;

}

int get\_k()

{

    int k;

    int size = user\_size - 1;

    do

    {

        cout << "Enter k(0 - " << size << "): ";

        cin >> k;

    } while (k < 0 || k > size);

    return k;

}

void print\_array\_at\_range(int\* array, int start)

{

    int end = start - 1;

    bool end\_null = false;

    if(end == -1)

    {

        end = user\_size - 1;

        end\_null = true;

    }

    int i;

    cout << "Array[" << user\_size << "] at range (" << start << ", " << end<< ") = {";

    for(i = start; i < user\_size; i++)

    {

        cout << array[i];

        if(i != end)

        {

            cout << ", ";

        }

    }

    if(!end\_null)

    {

        for(i = 0; i <= end; i++)

        {

            cout << array[i];

            if(i != end)

            {

                cout << ", ";

            }

        }

    }

    cout << "}" << endl;

}

void sort\_array\_by\_desc(int\* array)

{

    cout << "-------Sorted array by descending-------" << endl;

    for (int i = 0; i < user\_size - 1; i++)

    {

        for (int y = 0; y < user\_size - i - 1; y++)

        {

            if (array[y] < array[y + 1])

            {

                int temp = array[y];

                array[y] = array[y + 1];

                array[y + 1] = temp;

            }

        }

    }

}

void delete\_pair\_numbers(int\* array)

{

    cout << "-------Deleted pair elements from array-------" << endl;

    for(int i = 0; i < user\_size; i++)

    {

        if(array[i] % 2 == 0)

        {

            for(int y = i; y < user\_size - 1; y++)

            {

                array[y] = array[y + 1];

            }

        }

        user\_size--;

    }

}

int main()

{

    int default\_size = 100;

    int array[default\_size];

    fill\_array(array);

    print\_array(array);

    print\_array\_at\_range(array, get\_k());

    sort\_array\_by\_desc(array);

    print\_array(array);

    delete\_pair\_numbers(array);

    print\_array(array);

    print\_array\_at\_range(array, get\_k());

}

<https://github.com/volodymyr-rievkov/VolodymyrR_Reprository/blob/main/Epic%204/Lab4T1/vns_lab_4_task_1_variant_25_volodymyr_rievkov.cpp>

Завдання №2: Лабораторна робота №5

#include<iostream>

using namespace std;

const int rows = 3;

const int columns = 4;

int matrix[rows][columns];

const int array\_size = 2 \* (rows \* columns);

int matrix\_array[array\_size];

bool student\_exists(int number)

{

    for(int i = 0; i < rows; i++)

    {

        for(int y = 0; y < columns; y++)

        {

            if(number == matrix[i][y])

            {

                return true;

            }

        }

    }

    return false;

}

void fill\_matrix()

{

    cout << "------Enter number to fill matrix except zero------" << endl;

    int number;

    for(int i = 0; i < rows; i++)

    {

        for(int y = 0; y < columns; y++)

        {

            do

            {

                cout << "Element[" << i << "][" << y << "] = ";

                cin >> number;

            } while (student\_exists(number));

            matrix[i][y] = number;

        }

    }

    cout << endl;

}

void print\_matrix()

{

    cout << "Matrix[" << rows << "][" << columns << "]: " << endl;

    for(int i = 0; i < rows; i++)

    {

        for(int y = 0; y < columns; y++)

        {

            cout << "[" << matrix[i][y] << "] ";

        }

        cout << endl;

    }

    cout << endl;

}

void convert\_matrix\_to\_array()

{

    cout << "------Convertation matrix to array------" << endl;

    int index = 0;

    int i, y;

    for(i = 0; i < rows; i++)

    {

        for(y = 0; y < columns; y++)

        {

            matrix\_array[index] = matrix[i][y];

            index++;

        }

    }

    for(i = 0; i < columns; i++)

    {

        for(y = 0; y < rows; y++)

        {

            matrix\_array[index] = matrix[y][i];

            index++;

            matrix[y][i] = 0;

        }

    }

    cout << endl;

}

void print\_array()

{

    cout << "Array[" << array\_size << "] = ";

    for(int i = 0; i < array\_size; i++)

    {

        cout << "{" << matrix\_array[i] << "}";

        if(i != array\_size - 1)

        {

            cout << ", ";

        }

    }

    cout << endl << endl;

}

void convert\_array\_to\_matrix()

{

    cout << "------Convertation array to matrix------" << endl;

    int index = 0;

    int i, y;

    for(i = 0; i < rows; i++)

    {

        for(y = 0; y < columns; y++)

        {

            matrix[i][y] = matrix\_array[index];

            matrix\_array[index] = 0;

            index++;

        }

    }

    for(i = 0; i < columns; i++)

    {

        for(y = 0; y < rows; y++)

        {

            if(matrix[y][i] != matrix\_array[index])

            {

                matrix[y][i] = matrix\_array[index];

            }

            matrix\_array[index] = 0;

            index++;

        }

    }

    cout << endl;

}

int main()

{

    fill\_matrix();

    print\_matrix();

    convert\_matrix\_to\_array();

    print\_array();

    print\_matrix();

    convert\_array\_to\_matrix();

    print\_matrix();

}

<https://github.com/volodymyr-rievkov/VolodymyrR_Reprository/blob/main/Epic%204/Lab5T1/vns_lab_5_task_1_variant_25_volodymyr_rievkov.cpp>

Завдання №3: Лабораторна робота з Алготестеру №2

#include<iostream>

#include<sstream>

using namespace std;

int get\_size()

{

    int size;

    cin >> size;

    return size;

}

void fill\_array(int\* array, int size)

{

    for(int i = 0; i < size; i++)

    {

        cin >> array[i];

    }

}

int get\_max\_array\_number(int\* array, int size)

{

    int max = 0;

    for(int i = 0; i < size; i++)

    {

        if(array[i] > max)

        {

            max = array[i];

        }

    }

    return max;

}

int get\_min\_array\_number(int\* array, int size, int max)

{

    int min = max;

    for(int i = 0; i < size; i++)

    {

        if(array[i] < min && array[i] != -1)

        {

            min = array[i];

        }

    }

    return min;

}

int get\_count\_of\_array\_number(int\* array, int size, int number)

{

    int count = 0;

    for(int i = 0; i < size; i++)

    {

        if(number == array[i])

        {

            count++;

        }

    }

    return count;

}

void delete\_array\_number(int\* array, int size, int number)

{

    for(int i = 0; i < size; i++)

    {

        if(array[i] == number)

        {

            array[i] = -1;

        }

    }

}

int get\_result(int\* array, int size)

{

    if(size == 1)

    {

        return 0;

    }

    int max = get\_max\_array\_number(array, size);

    int min = get\_min\_array\_number(array, size, max);

    int max\_count = get\_count\_of\_array\_number(array, size, max);

    int min\_count = get\_count\_of\_array\_number(array, size, min);

    if(max\_count == min\_count)

    {

        delete\_array\_number(array, size, max);

    }

    else

    {

        if(max\_count < min\_count)

        {

            delete\_array\_number(array, size, max);

        }

        else

        {

            delete\_array\_number(array, size, min);

        }

    }

    max = get\_max\_array\_number(array, size);

    min = get\_min\_array\_number(array, size, max);

    return  max - min;

}

int main()

{

    int size = get\_size();

    int a\_road[size];

    fill\_array(a\_road, size);

    cout << get\_result(a\_road, size);

}

<https://github.com/volodymyr-rievkov/VolodymyrR_Reprository/blob/main/Epic%204/AlgtL2T1/algotester_lab_2_task_1_rievkov_volodymyr.cpp>

Завдання №4: Лабораторна робота з Алготестеру №3

#include<iostream>

using namespace std;

string get\_string()

{

    string str;

    cin >> str;

    return str;

}

string compress\_string(string str)

{

    string compressed\_string;

    int counter = 1;

    int index = 1;

    for(int i = 0; i < str.length() - 1; i++)

    {

        compressed\_string += str[i];

        if(str[i] == str[i + index])

        {

            while(str[i] == str[i + index])

            {

                index++;

                counter++;

            }

            compressed\_string += to\_string(counter);

            i += (index - 1);

            counter = 1;

            index = 1;

        }

    }

    if(str[str.length() - 1] != str[str.length() - 2])

    {

        compressed\_string += str[str.length() - 1];

    }

    return compressed\_string;

}

int main()

{

    string str = get\_string();

    cout << compress\_string(str);

}

<https://github.com/volodymyr-rievkov/VolodymyrR_Reprository/blob/main/Epic%204/AlgtL3T3/algotester_lab_3_task_3_rievkov_volodymyr.cpp>

Завдання №5: Практична робота

#include<iostream>

#include<string>

using namespace std;

string get\_string()

{

    string str;

    cout << "Enter string: ";

    cin >> str;

    return str;

}

int get\_int()

{

    int number;

    cout << "Enter number: ";

    cin >> number;

    return number;

}

string check\_lever(bool lever)

{

    if( lever == 1)

    {

        return "Palindrome";

    }

    else

    {

        return "not Palindrome";

    }

}

bool is\_palindrome(const string &str, int start, int end)

{

    bool palindrome = true;

    int tail = end;

    int head = start;

    if(start == end || start > tail && end < head)

    {

        return palindrome;

    }

    if(str[start] == str[end])

    {

        palindrome = true;

    }

    else

    {

        palindrome = false;

        return palindrome;

    }

    return is\_palindrome(str, ++start, --end);

}

bool is\_palindrome(const int &number, int start, int end)

{

    const string str = to\_string(number);

    return is\_palindrome(str, start, end);

}

int main()

{

    const string str = get\_string();

    const int number = get\_int();

    cout << str << " is " << check\_lever(is\_palindrome(str, 0, str.length() - 1)) << endl;

    cout << number <<" is " << check\_lever(is\_palindrome(number, 0, to\_string(number).length() - 1)) << endl;

}

<https://github.com/volodymyr-rievkov/VolodymyrR_Reprository/blob/main/Epic%204/Practise1Palindrome/practise_task_1_volodymyr_rievkov.cpp>

Завдання №6: Самостійна практична робота

#include<iostream>

#include<ctime>

using namespace std;

const int students\_amount = 30;

const int grades\_amount = 20;

const int max\_grade = 5;

const int min\_grade = 1;

const int subject\_amount = 3;

int student\_counter = 0;

string subjects[subject\_amount] = {"Math   ", "English", "History"};

bool exit\_s\_r = false;

struct Student

{

    string name;

    string surname;

    int age;

    string study\_email;

    int number;

    int s\_register[subject\_amount][grades\_amount];

    float average\_grade = 0;

};

Student students[students\_amount];

Student sorted\_students[students\_amount];

string set\_name\_value(string &name)

{

    cout << "Enter name: ";

    cin >> name;

    return name;

}

string set\_surname\_value(string &surname)

{

    cout << "Enter surname: ";

    cin >> surname;

    return surname;

}

int set\_age\_value(int &age)

{

    cout << "Enter age: ";

    cin >> age;

    return age;

}

int get\_age()

{

    int age;

    cout << "Enter age: ";

    cin >> age;

    return age;

}

string convert\_uppercase\_to\_lowercase(string str)

{

    str[0] = tolower(str[0]);

    return str;

}

string generate\_study\_email(Student &student)

{

    return convert\_uppercase\_to\_lowercase(student.name) + "." + convert\_uppercase\_to\_lowercase(student.surname) + "@gmail.com";;

}

void print\_student(Student &student)

{

    cout << "-----Student №" << student.number << "-----" << endl;

    cout << "Name: " << student.name << endl;

    cout << "Surame: " << student.surname << endl;

    cout << "Age: " << student.age << endl;

    cout << "Study email: " << student.study\_email << endl;

    cout << endl;

}

void print\_all\_students(Student\* students)

{

    cout << "-----------Students list-----------" << endl;

    for(int i = 0; i < students\_amount; i++)

    {

        if(students[i].name.empty())

        {

            break;

        }

        print\_student(students[i]);

    }

}

Student\* find\_student(string &name, string &surname)

{

    for(int i = 0; i < students\_amount; i++)

    {

        if(name == students[i].name && surname == students[i].surname)

        {

             return &students[i];

        }

    }

    return nullptr;

}

bool student\_exists(string &name, string &surname)

{

    for(int i = 0; i < students\_amount; i++)

    {

        if(name == students[i].name && surname == students[i].surname)

        {

           return true;

        }

    }

    return false;

}

void add\_student()

{

    if(student\_counter == students\_amount)

    {

        cout << "Error: Student list is full!" << endl;

    }

    else

    {

    int action;

    string name;

    string surname;

    int age;

    do

    {

        cout << "Press [1] for manual filling or [2] for auto filling: ";

        cin >> action;

    } while (action != 1 && action != 2);

    if(action == 1)

    {

        set\_name\_value(name);

        set\_surname\_value(surname);

        set\_age\_value(age);

    }

    else

    {

        string names[students\_amount] = {

        "Aiden", "Brooklyn", "Carter", "Delilah", "Elijah",

        "Faith", "Grayson", "Harper", "Isaiah", "Jasmine",

        "Kaden", "Layla", "Mason", "Nevaeh", "Owen",

        "Peyton", "Quinn", "Riley", "Sophia", "Tyler",

        "Upton", "Vanessa", "Wyatt", "Ximena", "Yasmine",

        "Zane", "Ava", "Brayden", "Chloe", "Daniel"};

        string surnames[students\_amount] = {

        "Anderson", "Baker", "Clark", "Davis", "Edwards",

        "Fisher", "Garcia", "Hill", "Irwin", "Johnson",

        "Keller", "Lopez", "Martinez", "Nelson", "Owens",

        "Perez", "Quinn", "Rodriguez", "Smith", "Taylor",

        "Upton", "Valdez", "Watson", "Xavier", "Young",

        "Zimmerman", "Adams", "Brown", "Chavez", "Dixon"};

        do

        {

            name = names[rand() % 30];

            surname = surnames[rand() % 30];

        } while (student\_exists(name, surname));

        age = rand() % 13 + 7;

    }

    students[student\_counter].name = name;

    students[student\_counter].surname = surname;

    students[student\_counter].age = age;

    students[student\_counter].study\_email = generate\_study\_email(students[student\_counter]);

    students[student\_counter].number = student\_counter + 1;

    student\_counter++;

    cout << endl;

    cout << "You added: " << endl;

    print\_student(\*find\_student(name, surname));

    }

}

void delete\_student()

{

    if(student\_counter == 0)

    {

        cout << "Error: Student list is empty!" << endl;

    }

    else

    {

        string name;

        string surname;

        int index;

        set\_name\_value(name);

        set\_surname\_value(surname);

        cout << endl;

        Student\* student = find\_student(name, surname);

        if(student == nullptr)

        {

            cout << "Error: Student was not founded!" << endl;

            cout << endl;

        }

        else

        {

            cout << "You deleted: " << endl;

            print\_student(\*student);

            index = student->number - 1;

            for(index; index < students\_amount - 1; index++)

            {

                students[index] = students[index + 1];

                if(students[index].number != index + 1)

                {

                    students[index].number--;

                }

            }

            student\_counter--;

        }

    }

}

int choose\_subject()

{

    int subject;

    do

    {

        for(int i = 0; i < subject\_amount; i++)

        {

            cout << "---" << subjects[i] << "[" << i + 1 << "]" <<  endl;

        }

        cout << "Choose subject: ";

        cin >> subject;

    } while (subject < 0 || subject > subject\_amount);

    return subject - 1;

}

void print\_subject\_grades(int\* student\_register)

{

    for(int i = 0; i < grades\_amount; i++)

    {

        if(student\_register[i] == 0)

        {

            cout << "|0";

        }

        else

        {

            cout << "|" << student\_register[i];

        }

    }

    cout << "|" << endl;

}

void give\_grade(Student &student)

{

    int subject = choose\_subject();

    int grade;

    if(student.s\_register[subject][grades\_amount - 1] != 0)

    {

        cout << "Error: Register is full!" << endl;

    }

    else

    {

        do

        {

            cout << "Enter grade(" << min\_grade << "-" << max\_grade << "): ";

            cin >> grade;

        } while (grade > max\_grade || grade < min\_grade);

        cout << endl;

        for(int i = 0; i < grades\_amount; i++)

        {

            if(student.s\_register[subject][i] == 0)

            {

                student.s\_register[subject][i] = grade;

                break;

            }

        }

    }

}

void set\_average\_grade(Student &student)

{

    float grades\_counter;

    student.average\_grade = 0;

    for(int i = 0; i < subject\_amount; i++)

    {

        for(int y = 0; y < grades\_amount; y++)

        {

            if(student.s\_register[i][y] != 0)

            {

                student.average\_grade += student.s\_register[i][y];

                grades\_counter++;

            }

        }

    }

    if(grades\_counter != 0)

    {

        student.average\_grade /= grades\_counter;

    }

    else

    {

        student.average\_grade =  grades\_counter;

    }

}

void print\_student\_average\_grade(Student &student)

{

    set\_average\_grade(student);

    cout << "Average grade: " << student.average\_grade << endl << endl;;

}

void print\_all\_subject\_grades(Student &student)

{

    for(int i = 0; i < subject\_amount; i++)

    {

        cout << "---" << subjects[i];

        print\_subject\_grades(student.s\_register[i]);

        cout << endl;

    }

}

void print\_all\_students\_grades()

{

    cout << "-------------Students grades-------------" << endl;

    for(int i = 0; i < students\_amount; i++)

    {

        if(students[i].name.empty())

        {

            break;

        }

        cout << students[i].name << " " << students[i].surname << endl;

        print\_all\_subject\_grades(students[i]);

        print\_student\_average\_grade(students[i]);

        cout << endl;

    }

}

Student\* sort\_by\_progress()

{

    int i;

    Student temp;

    for(i = 0; i < students\_amount; i++)

    {

        sorted\_students[i] = students[i];

    }

    for (i = 0; i < students\_amount - 1; i++)

    {

        for (int y = 0; y < students\_amount - i - 1; y++)

        {

            if (sorted\_students[y].average\_grade < sorted\_students[y + 1].average\_grade)

            {

                temp = sorted\_students[y];

                sorted\_students[y] = sorted\_students[y + 1];

                sorted\_students[y + 1] = temp;

            }

        }

    }

    return sorted\_students;

}

void exit\_student\_register()

{

    exit\_s\_r = true;

}

int choose\_action()

{

    int action;

    do

    {

        cout << " - Find student[1]\n - Add student[2]\n - Delete student[3]\n - Print all student[4]\n - Print all grades[5]\n - Sort all students by progress[6]\n - Exit[7]\n: ";

        cin >> action;

    } while (action < 1  || action > 7);

    cout << endl;

    return action;

}

int choode\_action2()

{

    int action;

    do

    {

        cout << " - Print student[1]\n - Give grade[2]\n - Print grades[3]\n - Print average grade[4]\n - Exit[5]\n: ";

        cin >> action;

    } while (action < 1  || action > 5);

    cout << endl;

    return action;

}

void student\_register\_app()

{

    string name;

    string surname;

    Student \*student;

    srand(time(nullptr));

    while(!exit\_s\_r)

    {

        switch(choose\_action())

        {

            case 1:

                    set\_name\_value(name);

                    set\_surname\_value(surname);

                    cout << endl;

                    student = find\_student(name, surname);

                    if(student == nullptr)

                    {

                        cout << "Error: Student was not founded!" << endl << endl;

                        break;

                    }

                    while(!exit\_s\_r)

                    {

                        switch(choode\_action2())

                        {

                            case 1: print\_student(\*student);

                                    break;

                            case 2:

                                    give\_grade(\*student);

                                    break;

                            case 3: print\_all\_subject\_grades(\*student);

                                    break;

                            case 4: print\_student\_average\_grade(\*student);

                                    break;

                            case 5: exit\_student\_register();

                                    break;

                        }

                    }

                    exit\_s\_r = false;

                    break;

            case 2: add\_student();

                    break;

            case 3: delete\_student();

                    break;

            case 4: print\_all\_students(students);

                    break;

            case 5: print\_all\_students\_grades();

                    break;

            case 6: sort\_by\_progress();

                    print\_all\_students(sorted\_students);

                    break;

            case 7: exit\_student\_register(); break;

        }

    }

}

int main()

<https://github.com/volodymyr-rievkov/VolodymyrR_Reprository/blob/main/Epic%204/Practise2StudentRegisterApp/practise_task_2_volodymyr_rievkov.cpp>

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

Завдання №1: Лабораторна робота №4

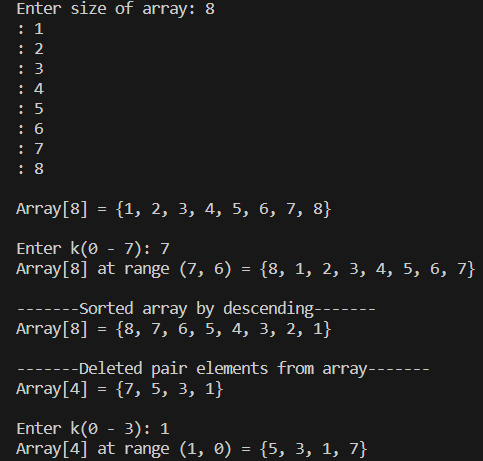


Рисунок 9: Лабораторна робота №4

Час затрачений на виконання завдання: 6 годин

Завдання №2: Лабораторна робота №5

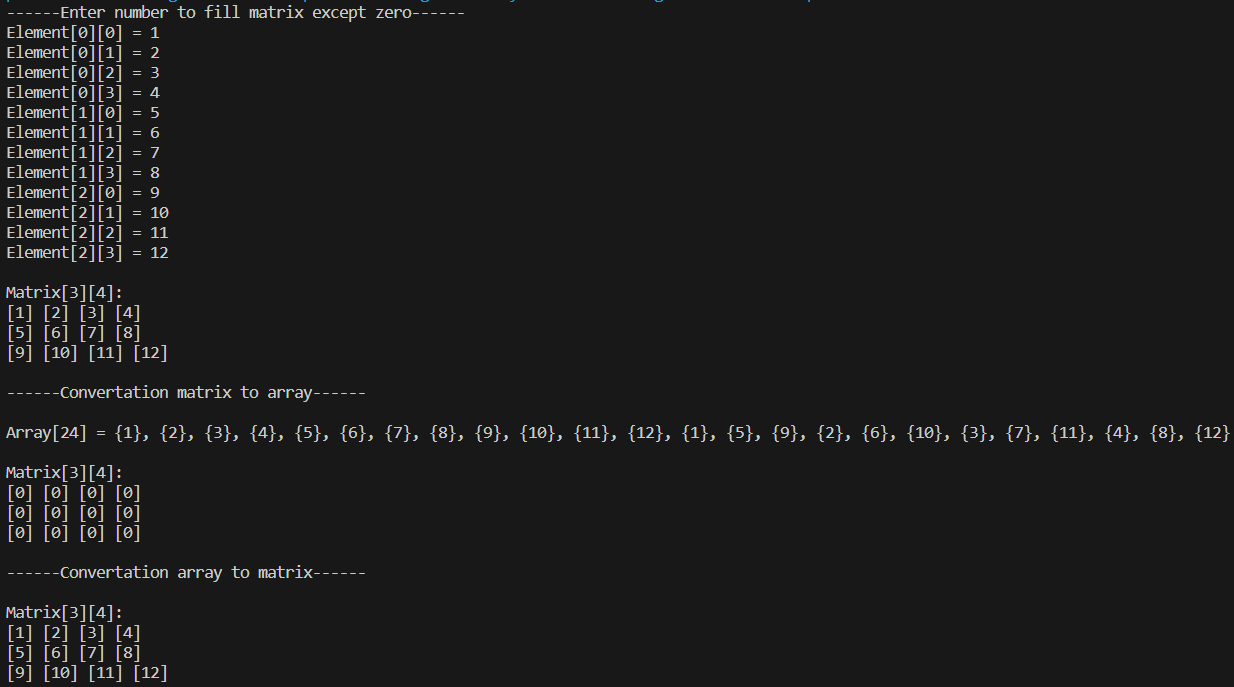


Рисунок 10: Лабораторна робота №5

Час затрачений на виконання завдання: 2 години

Завдання №3: Лабораторна робота з Алготестера №2

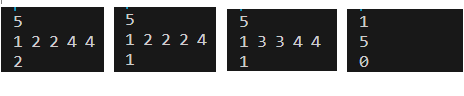


Рисунок 11: Лабораторна робота з Алготестера №2, результат в VS Code

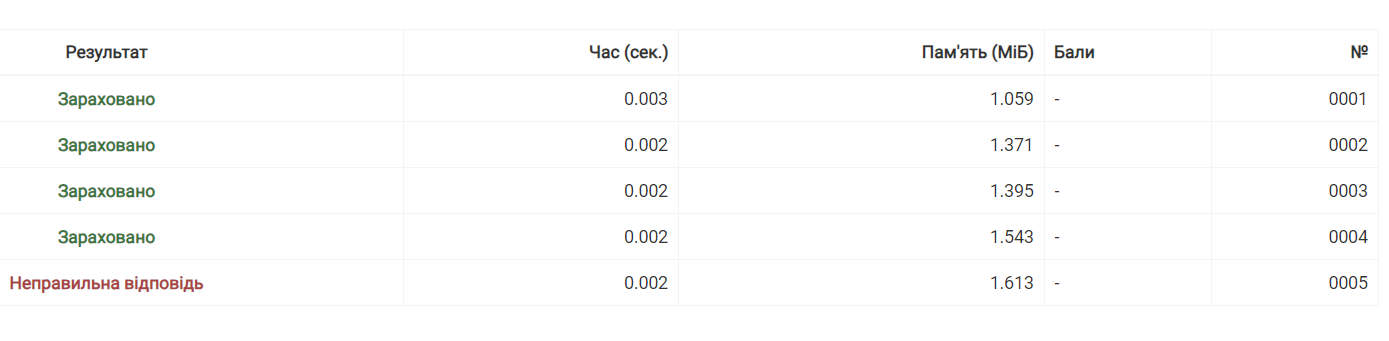


Рисунок 12: Лабораторна робота з Алготестера №2, результат в Алготестері

Час затрачений на виконання завдання: 30 хвилин

Завдання №4: Лабораторна робота з Алготестера №3



Рисунок 13: Лабораторна робота №3 з Алготестера

Час затрачений на виконання завдання: 20 хвилин

Завдання №5: Практична робота

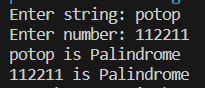
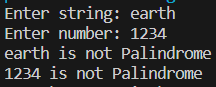
 

Рисунок 14: Практична робота Рисунок 15: Практична робота

Час затрачений на виконання завдання: 20 хвилин

Завдання №6: Самостійна практична робота

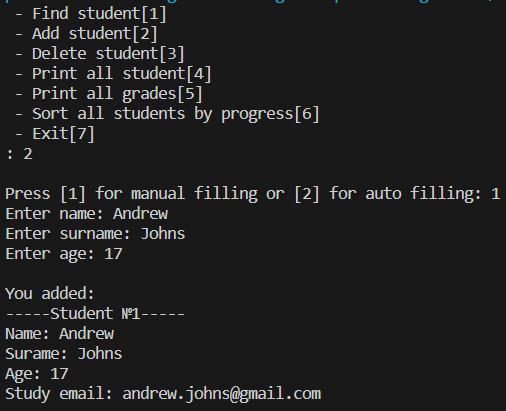
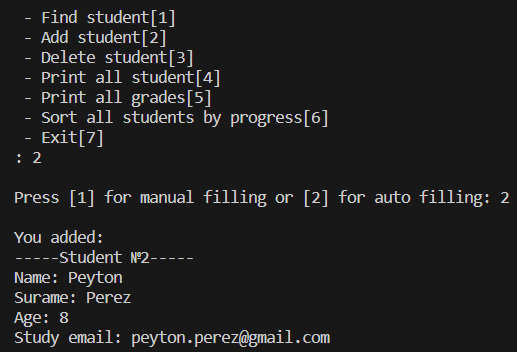
 

Рисунок 16: Самостійна практична робота Рисунок 17: Самостійна практична робота

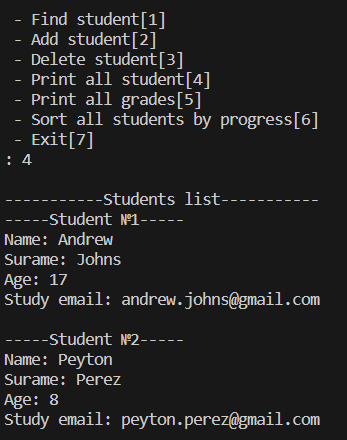
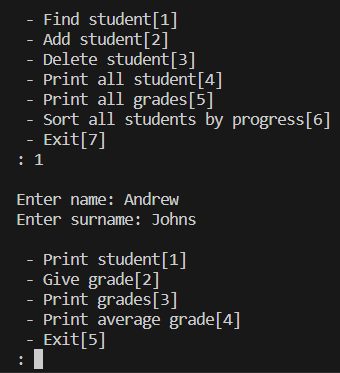
 

Рисунок 18: Самостійна практична робота Рисунок 19: Самостійна практична робота

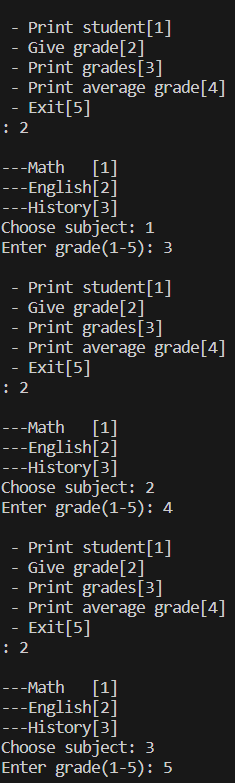
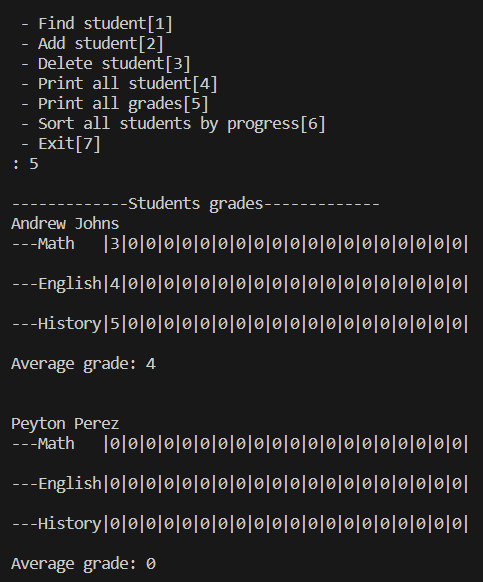
 

Рисунок 20: Самостійна практична робота Рисунок 21: Самостійна практична робота

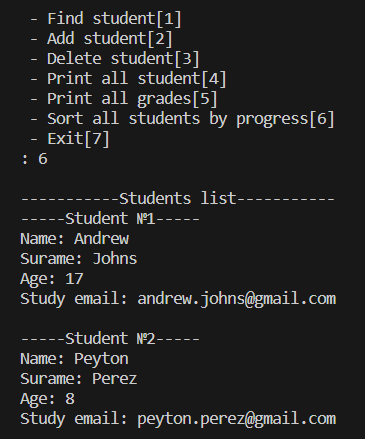
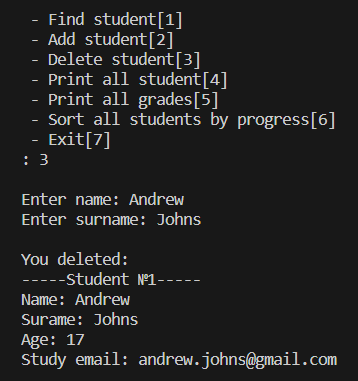
 

Рисунок 22: Самостійна практична робота Рисунок 23: Самостійна практична робота

Час затрачений на виконання завдання: 12 годин

**Висновки:**

Виконавши завдання Епіку №4, я ознайомився з розділом «Одновимірні масиви. Двовимірні Масиви. Вказівники та Посилання. Динамічні масиви. Структури даних. Вкладені структури. Алгоритми обробки та робота з масивами та структурами.» та виконав лабораторні роботи ВНС №4 та №5, лабораторні роботи з Алготестеру №2 та №3, практичну та самостійну практичну роботи.