

4IT Erasmus+Project, 2025-1-RO01-KA121-VET-000319082

Beneficiary:"Gheorghe Vranceanu" National College, Bacau

Sending organization: "Costache Negri" National College Targu Ocna

MOBILITY PORTFOLIO
Paphos, Cyprus

Mobility period: 01.04.2026-14.04.2026

Practice period 02.04.2026-13.04.2026

RECEIVING COMPANY: LAZAROU ICT PROFESSIONALS

Practice tutor: Lazaros Lazarou

Participant: Ștefan Dragoș-Florin

DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

2025-1-RO01-KA121-VET-000319082

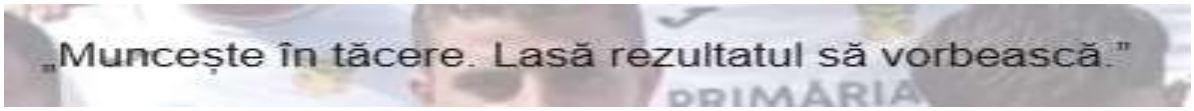
SECTIUNEA I

➢ Pagina principală



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- **Motto**



- **Logo**



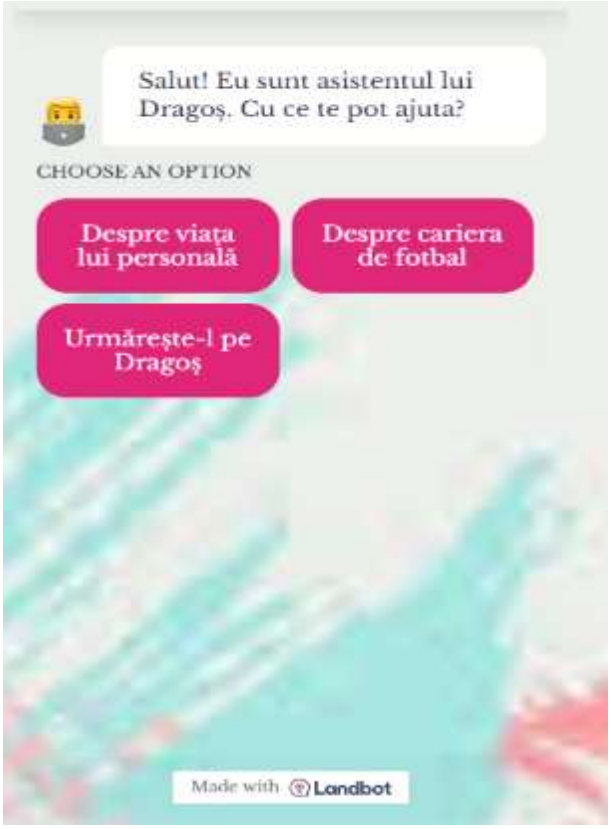


- **Un argument**



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- **Un chatbot**

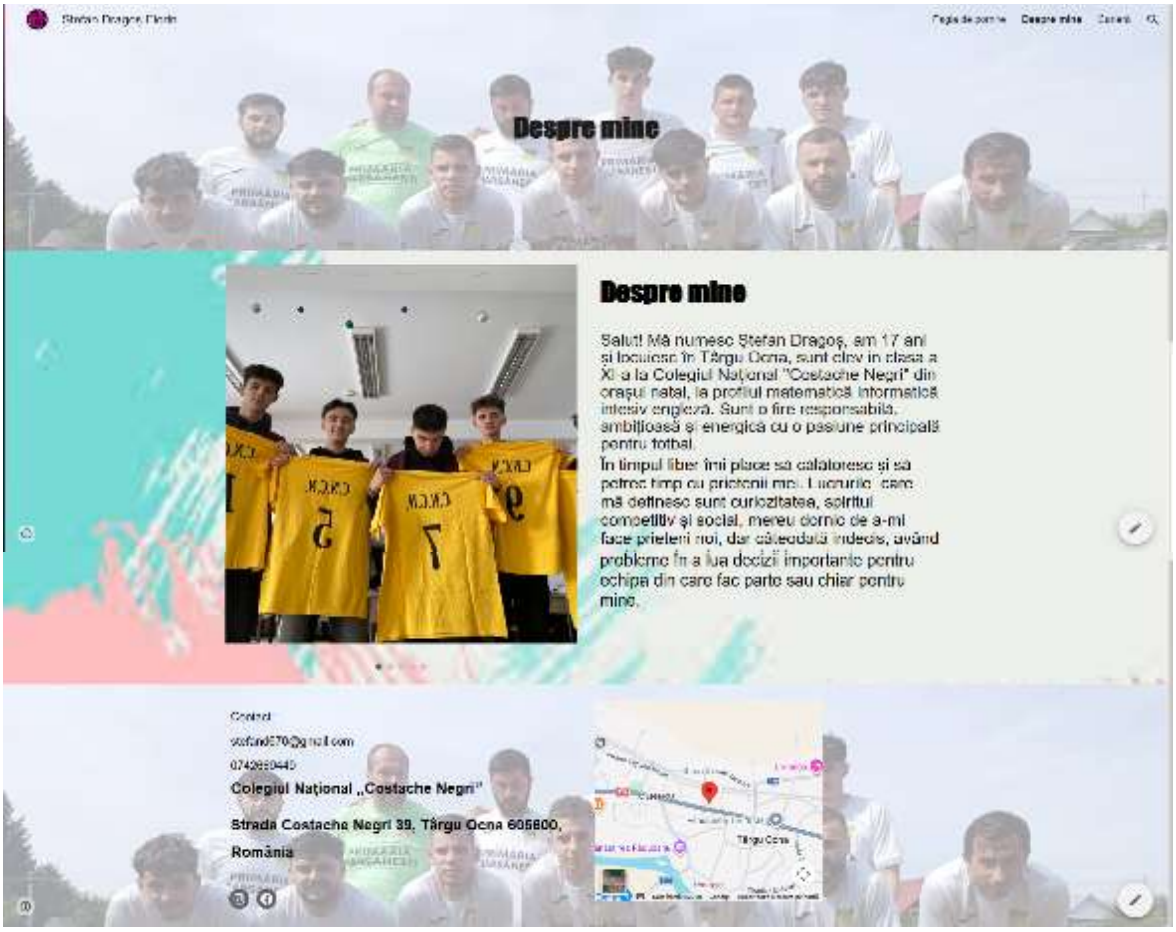


- **Un avatar/GIF asociat**



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

➢ Pagina care reflectă personalitatea,aspiratiile,pasiunile mele



➢ Galerie interactive de imagini originale





DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

➢ Clima orașului Paphos, Cipru



TEMPERATURILE MEDII LUNARE

Lună	Medie max (°C)
Ianuarie	17
Februarie	16.9
Martie	18.5
Aprilie	21.3
Mai	24.4
Iunie	27.7
Iulie	29.9
August	30.4
Septembrie	28.8
Octombrie	26.6
Noiembrie	22.4
Decembrie	18.6

Temperaturile sunt ridicate vara, atingând valori de peste 30°C în lunile iulie și august. Iarna, temperaturile sunt blânde, cu medii de aproximativ 16–18°C.

PRECIPIATII MEDII LUNARE

Lună	Precipitații
Jan	80.2
Feb	64.2
Mar	34.3
Apr	18.7
Mai	5.1
Iun	1.6
Iul	0.1
Aug	0
Sep	3.8
Oct	18
Nov	56.4
Dec	93.9

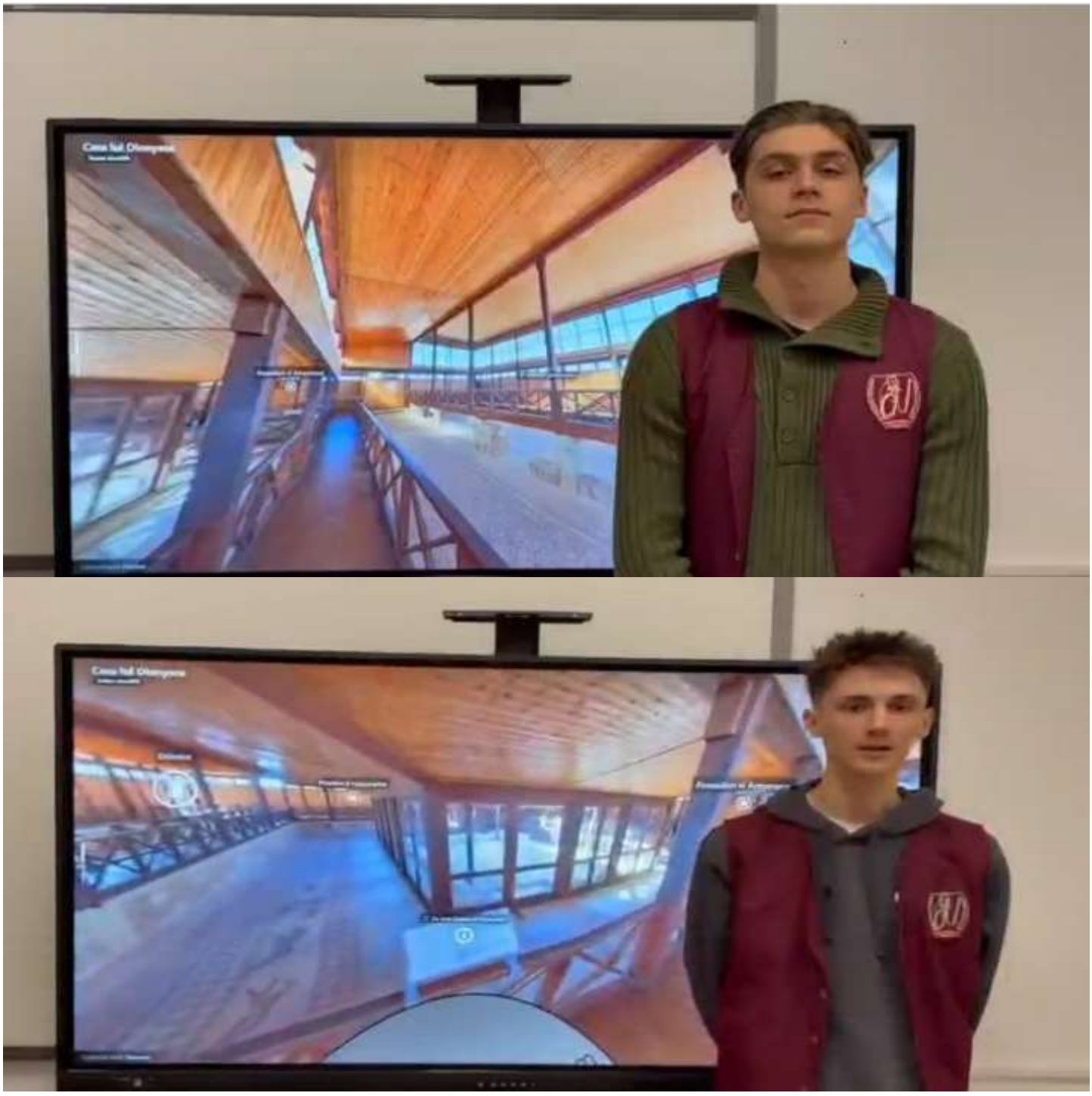
Precipitațiile sunt reduse în timpul verii, iar cele mai mari cantități de ploaie se înregistrează iarna, în special în lunile decembrie și ianuarie.

DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

➢ Proiect Pregătire culturală și Green Society

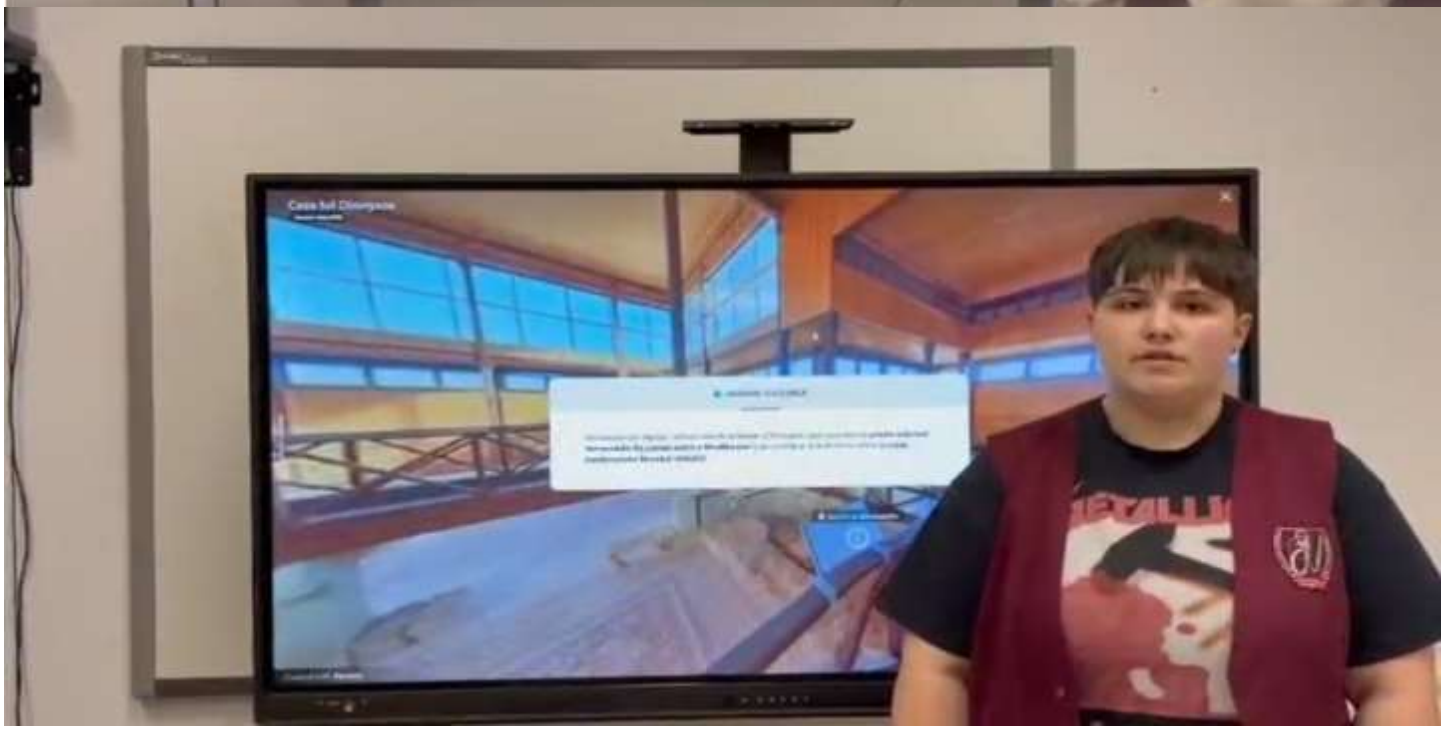


DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.





DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

SECTIUNEA II

➢ Practica in C++

○ Alocarea dinamică a memoriei

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


struct student{int age;};
int main(){
    int ARRAYSIZE = 0;
    student* studentArr;
    student* temp;
    int  n = 0, tmpN=0;

    cout<<"How many students would you like to enter?:";
    cin>>ARRAYSIZE;
    studentArr = new student[ARRAYSIZE];

    do{
        cout<<"Enter student "<<n+1<<" age -> ";
        cin >> studentArr[n].age;
        cout<<"Number saved!"<<endl;
        n++;
        if (n >= ARRAYSIZE){
            cout<<"Out of Space!!!! would you like to enter more? Please enter how many(0 to Exit): ";
            cin>>tmpN;
            if (tmpN > 0){
                ARRAYSIZE = ARRAYSIZE + tmpN;
                temp = new student[ARRAYSIZE];
                 for (int i=0; i<n; i++) {
                     temp[i] = studentArr[i];
                 }
                 delete [] studentArr;
                 studentArr = temp;

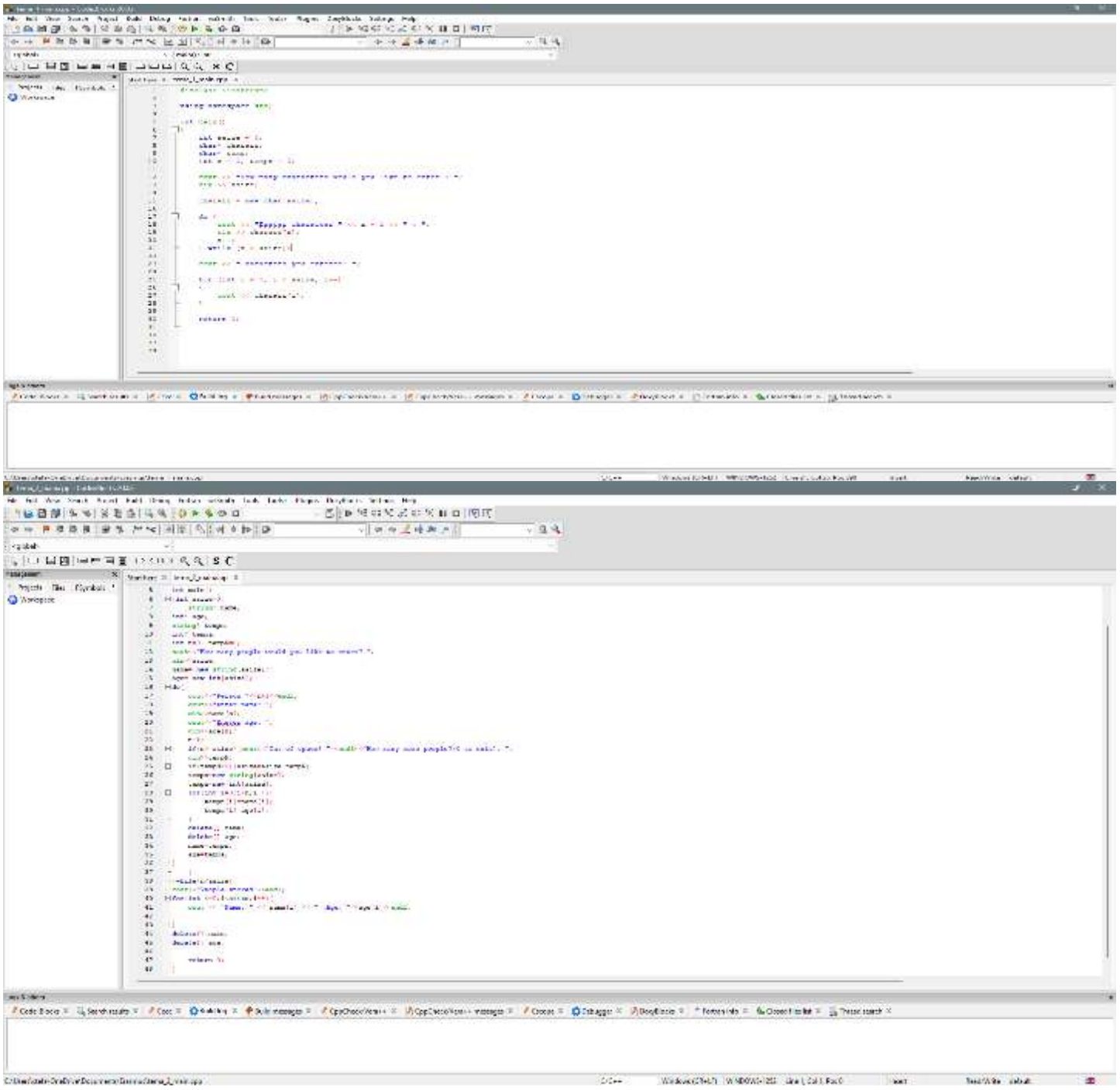
            }
        }

    }while (n<ARRAYSIZE);

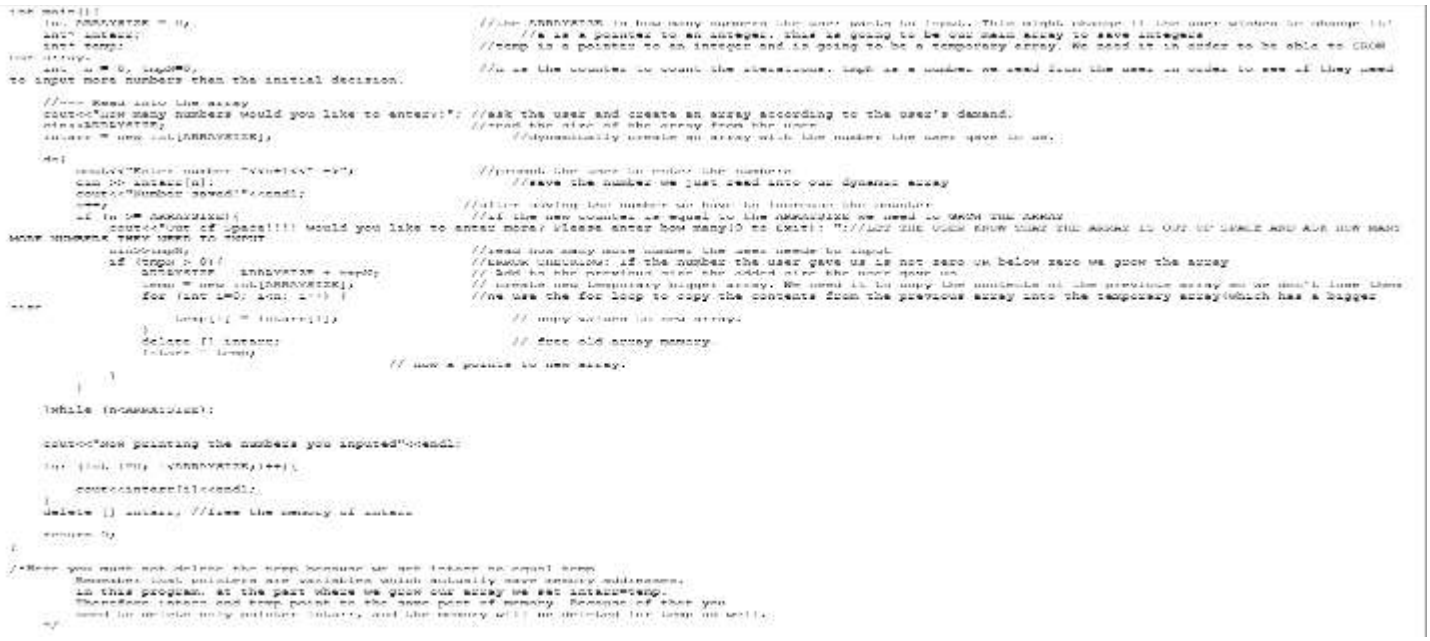

    cout<<"Now printing the students ages you inputed"<<endl;

    for (int i=0; i<ARRAYSIZE;i++){
        cout<<studentArr[i].age<<endl;
    }
    delete [] studentArr;
    return 0;
}
```

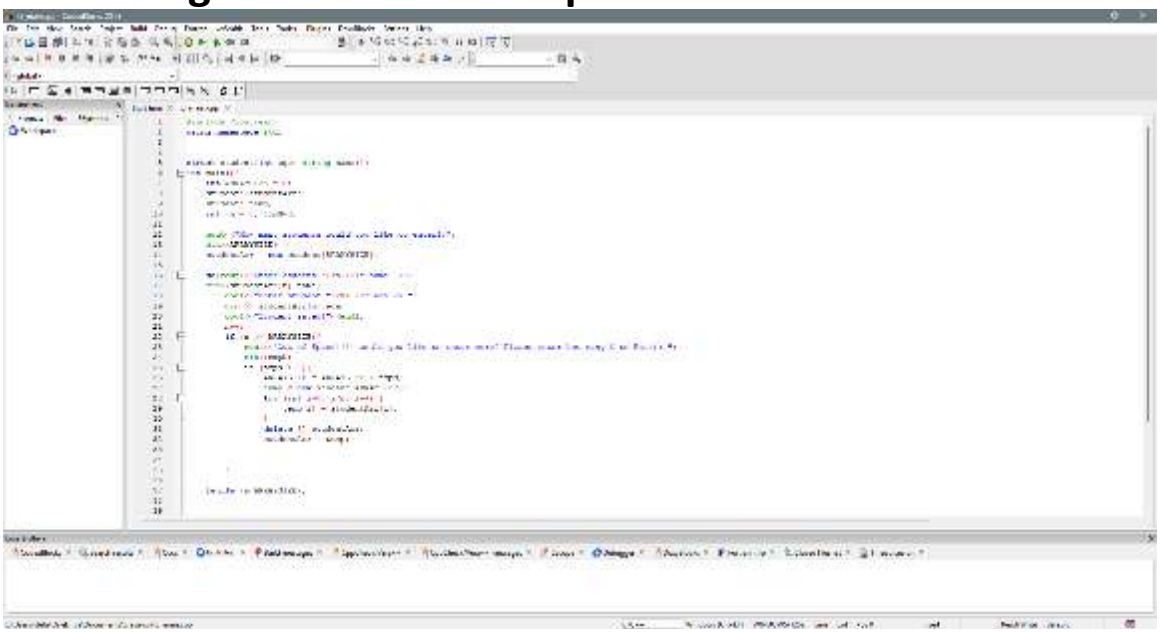
DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



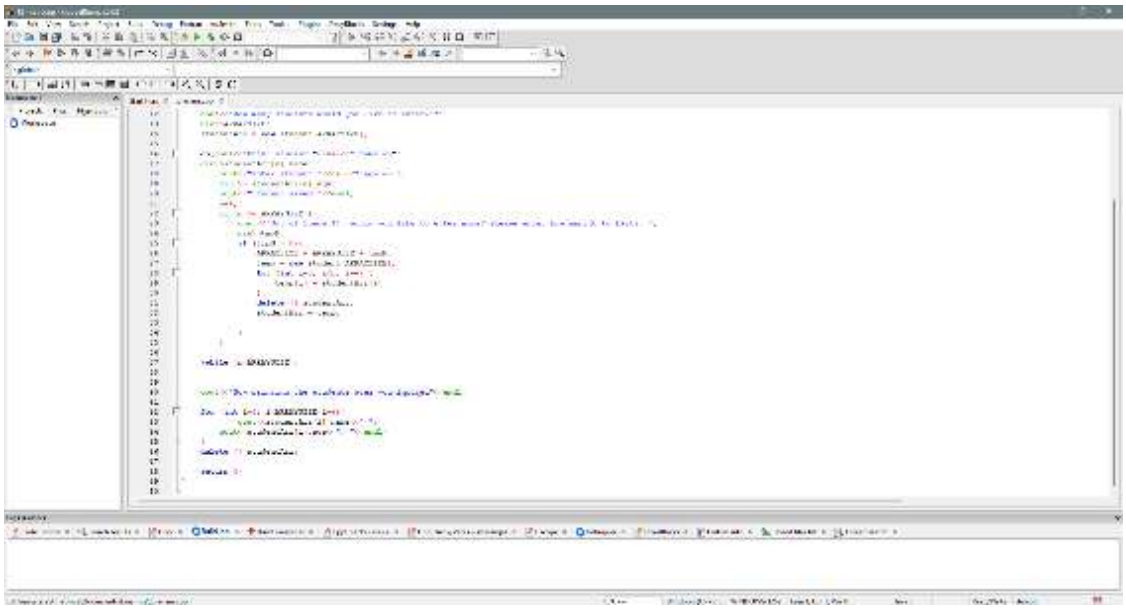
DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



- **Programare orientată pe obiect**



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



```
#include <iostream>
using namespace std;
/*
EXERCISE
Write a program which takes input from the user
the information for cars.
you need to save the following information
BRAND
MODEL
YEAR
WT
When you are done reading print your information on the screen

*/
class student{
    private:
        string name;
        int age;
        float ex1,ex2;

    public:
        void setName(string n){
            name=n;
        }
        void setAge(int a){age =a;}
        void setEx1(float e1){ex1=e1;}
        void setEx2(float e2){ex2=e2;}

        string getName(){return name;}
        int getAge(){return age;}
        float getEx1(){return ex1;}
        float getEx2(){return ex2;}
};

int main(){

    student st;
    student myClass[5];
    string name;
    int age;
    float e1,e2;

    st.setName("Lazarca");
    st.setAge(42);
    st.setEx1(7);
    st.setEx2(8);

    cout<<"Name: "<<st.getName()<<endl;
    cout<<"Age: "<<st.getAge()<<endl;
    cout<<"Exam 1: "<<st.getEx1()<<endl;
    cout<<"Exam 2: "<<st.getEx2()<<endl;
    cout<<"Avg: "<<(st.getEx1()+st.getEx2())/2<<endl;
```

DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
    for (int i=0;i<5;i++){
        cout<<"Name: ";
        cin>>name;
        myClass[i].setName(name);
        cout<<"Age: ";
        cin>>age;
        myClass[i].setAge(age);
        cout<<"Examl: ";
        cin>>e1;
        myClass[i].setEx1(e1);
        cout<<"Exam 2: ";
        cin>>e2;
        myClass[i].setEx2(e2);
    }
    for (int i=0;i<5;i++){
        cout<<myClaaa[i].getName()<<endl;
        cout<<myClaaa[i].getAge()<<endl;
        cout<<myClaaa[i].getEx1()<<endl;
        cout<<myClaaa[i].getEx2()<<endl;
        cout<<"----------------------"<<endl<<endl;
    }
    return 0;
}
```

```
#include <fstream>
#include <iostream>
using namespace std;
/*
    Part 1
    -------
    create a program that reads the
    following information from the user:
    Patient Name   string
    Patient Age    int
    Patient Phone  string
    Patient Medicine string
    you will create an array of objects since a hospital
    has so many patients.
    The program must use a "dynamic" array to a patient object.

    Part 2
    ------
    Create a menu for your program in order to give the user options.
    Give the following options:
    1. Add a new user
    2. Edit a User
    3. Delete a User
    4. Search by Name
    5. Search by Medicine
    6. Print All data (you can use it for testing)
    0. Exit
    With the creation of the menu, you are now
    required to change your program in such a way
    that whatever choice the user makes you will
    call a function to accomplish the selected task.
    PAGE 3

    Write the Search function for your program
*/

//We will use the patient class to create the data structure
//to save our patients information. the data structure is going to
//be a dynamic array of patients
class patient{
    private:
        string name;
        int age;
        string phone;
        string medicine;

    public:
        //All the following functions are used to
        //save information for the objects that we create in
        //the object array. All SetFunctions are used to assign
        //values into the objects.
        //All GetFunctions are used to retrieve the values
        //from the objects
```

DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
        void setName(string n) {name=n;}
        void setAge(int a)     {age=a;}
        void setPhone(string p)   {phone=p;}
        void setMedicine(string m)  {medicine=m;}

        string getName()     {return name;}
        int getAge()         {return age;}
        string getPhone()    {return phone;}
        string getMedicine(){return medicine;}
};
//the addPatient function is used to
//add one patient into our array of patient objects
void addPatient(patient *cl,int index);
//growArray is a function to make the array bigger. it takes
//a patient pointer and the arraySize
patient *growArray(patient *cl,int arraySize,int index);
//printArray is a function to print all the data
//that are saved into the array.
//it takes as parameters the clinic array and the index
void printArray(patient *clinic, int index);
//searchByName(patient *, index)
//this function takes for parameters a patient pointer(which is the dynamic array)
//and also the index(the last position we saved at in the  array).
//it uses a for loop to go through the array and print all the information
//that we are looking for.
bool searchByName(patient *cl,int index);
//searchByMedicine(patient *, index)
//this function takes for parameters a patient pointer(which is the dynamic array)
//and also the index(the last position we saved at in the  array).
//it uses a for loop to go through the array and print all the information
//that we are looking for.
void searchByMedicine(patient *cl,int index);
void editPatient(patient *cl,int index);
void deletePatient(patient *cl,int index);
void saveToFile(patient *cl,int index,ofstream &out);
patient* readFromFile(patient *cl,int &index, ifstream &inf);

int main(){
    int ArraySize=1;//set the array size to 1 and then we can grow the array if needed
    int index=0; //index is the last index that we are using in the array(the last) of the array)
    patient *clinic, *temp; //a pointer to a patient to create a dynamic array
    clinic=new patient[ArraySize]; //allocate space for the array
    int userChoice=1; //needed to read the choice the user is giving us.
    ifstream inputfile;
    ofstream outfile;

    inputfile.open("data.txt");
    if (!inputfile.is_open()){//check if the file exists
        cout<<"Data file does not exist at this moment."<<endl;
        cout<<"Enter Data and the file will automatically be created"<<endl;
    }else{ //the file exists the file is already opened for input
        //first open the data file for reading
        cout<<"the file exists! Opening for reading!"<<endl;
        clinic=readFromFile(clinic,index,inputfile);
        ArraySize=index;
    }

    while (userChoice >0){
        //print the menu for the user
        cout<<"1. Add a new user"<<endl;
        cout<<"2. Edit a User"<<endl;
        cout<<"3. Delete a User"<<endl;
        cout<<"4. Search by Name"<<endl;
        cout<<"5. Search by Medicine"<<endl;
        cout<<"6. Print The Array"<<endl;
        cout<<"0. Exit"<<endl;
        cout<<"What would you like to do? ";
        cin>>userChoice;
        cout<<"-----------------------------------"<<endl;

        if (userChoice == 1){
            cout<<"Add a new Patient"<<endl<<"-----------------"<<endl;
            if (index>=ArraySize){
                ArraySize=ArraySize+1;//grow the array by 1 position
                clinic=growArray(clinic,ArraySize,index); //call function to grow array
            }
            addPatient(clinic,index);//call the function to add a patient
            index++;//after we added a patient we must increment the index
        }else if(userChoice==2){
            cout<<"Edit a Patient"<<endl<<"-----------------"<<endl;
            editPatient(clinic,index);
        }else if(userChoice==3){
            cout<<"Delete a Patient"<<endl<<"-----------------"<<endl;
            deletePatient(clinic,index);
            index--;
        }else if (userChoice==4){
            cout<<"Search By Name"<<endl<<"-----------------"<<endl;
            searchByName(clinic,index);
        }else if(userChoice==5){
            cout<<"Search By Medicine"<<endl<<"-----------------"<<endl;
            searchByMedicine(clinic,index);
        }else if(userChoice==6){
            printArray(clinic,index);

        }else if (userChoice==0){
            outfile.open("data.txt",ios::trunc);
            saveToFile(clinic,index,outfile);
            outfile.close();
            cout<<"Thank you For using the program!"<<endl<<"-----------------"<<endl;
        }else{
            cout<<"Wrong Choice! Try again"<<endl;
            userChoice=10;
        }
    }
    return 0;

}
patient* readFromFile(patient *cl,int &index, ifstream &inf){
    string temp;
    inf>>temp;//read the first line of the file (how many patients are saved
    index=stoi(temp);//convert the string you read from the file into a number
```

DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
    ol=new patient[index];

    for (int i=0;i<index;i++){
        inf>>temp;
        cl[i].setName(temp);
        inf>>temp;
        ol[i].setAge(stoi(temp));
        inf>>temp;
        cl[i].setPhone(temp);
        inf>>temp;
        cl[i].setMedicine(temp);
    }
    return cl;//return the pointer that points onto the array back to main()
}
void saveToFile(patient *cl,int index,ofstream &outf){

    outf<<index<<endl;
    for (int i=0;i<index;i++){
        outf<<cl[i].getName()<<" ";
        outf<<cl[i].getAge()<<" ";
        outf<<ol[i].getPhone()<<" ";
        outf<<cl[i].getMedicine()<<endl;
    }
    cout<<"Data Saved!"<<endl;


}
void deletePatient(patient *cl, int index){
    /*
    1. call the search function so the user
    will find out the position they want to delete.
    2. ask the user about the position they want to delete
    3. use a for loop to move everything from position +1 all the way
    to index, back on position.
    */
    int arrSize=index; //index is confusing us! lets make it something
                       //we can understand
    int pos;
    if (!searchByName(cl,arrSize)){
        cout<<"Try again with a different name!"<<endl;
    }else{

        cout<<"Position to delete: ";
        cin>>pos;

        if (arrSize <1){
            cout<<"Array is empty"<<endl;
            return;
        }
        if (pos<0 || pos >arrSize ){
            cout<<"The position is wrong!";
            return;
        }
        for (int i=pos; i<arrSize-1;i++){
            ol[i].setName(ol[i+1].getName());
            cl[i].setAge(cl[i+1].getAge());
            cl[i].setPhone(cl[i+1].getPhone());
            cl[i].setMedicine(cl[i+1].getMedicine());

        }
        [illegible]
    }

}
void editPatient(patient *cl,int index){

    /*
    1. Call the search function with whatever the user is looking for
    2. Get the position from the user that the patient is saved to
    3. Do the editing at the specific position
    */
    [illegible]
    int position;
    [illegible]
    cin>>position;
    //do the editing by calling addPatient() function.
    addPatient(cl,position);
}
[illegible]
    string query;
    int resultsFound=0;
    cout<<"Enter name to look for: ";
    cin>>query;

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

        if (cl[i].getName() == query){
            resultsFound++;
            cout<<"          Result "<<resultsFound<<"          "<<endl;
            cout<<cl[i].getName()<<endl;
            [illegible]
            [illegible]
            [illegible]
            cout<<"Position found: "<<i<<endl;
        }

    }
    if (resultsFound==0){
        cout<<"No Results Found!"<<endl;
        return false;
    }else{
        return true;
    }

}
void searchByMedicine(patient *cl,int index){
    string query;
    int resultsFound=0;
    cout<<"Enter Medicine to look for: ";
    cin>>query;
```

DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
    for (int i=0;i<index;i++){

        if (cl[i].getMedicine() == murry){
            counter++;
            cout<<"---------- Result "<<counter<<" ----------"<<endl;
            cout<<cl[i].getName()<<endl;
            cout<<cl[i].getAge()<<endl;
            cout<<cl[i].getPhone()<<endl;
            cout<<cl[i].getMedicine()<<endl;
        }
    }
}
/*
function growArray
the function is used to grow the clinic array by 1.
it takes as parameters, the clinic array (it is a patient pointer),
the arraySize (which we incremented in the main function,
and the index which we will increment in the main function,
after we make the array bigger).
The function returns a patient pointer which is the new bigger array.
*/
patient *growArray(patient *clinic,int ArraySize, int index){
    patient *temp;
    temp=new patient[ArraySize];
    //copy the data from the original in temp array
    for (int i=0;i<index;i++){
        temp[i].setAge(clinic[i].getAge());
        temp[i].setMedicine(clinic[i].getMedicine());
        temp[i].setName(clinic[i].getName());
        temp[i].setPhone(clinic[i].getPhone());
    }
    delete [] clinic;//delete the array with the old size
    return temp; //return temp(array with new size) back to the main function
}
/*
function addPatient
  takes as parameters a patient pointer (the dynamic array)
  the index, which is the next available position to save the data
  and the arraySize because we need to check if the array is full
-the function asks the user about the patient information and saves the data
  from the keyboard into the dynamic array, at position index.
-The function will check if the array is full and grow the array dynamically.
*/
void addPatient(patient *cl,int index){

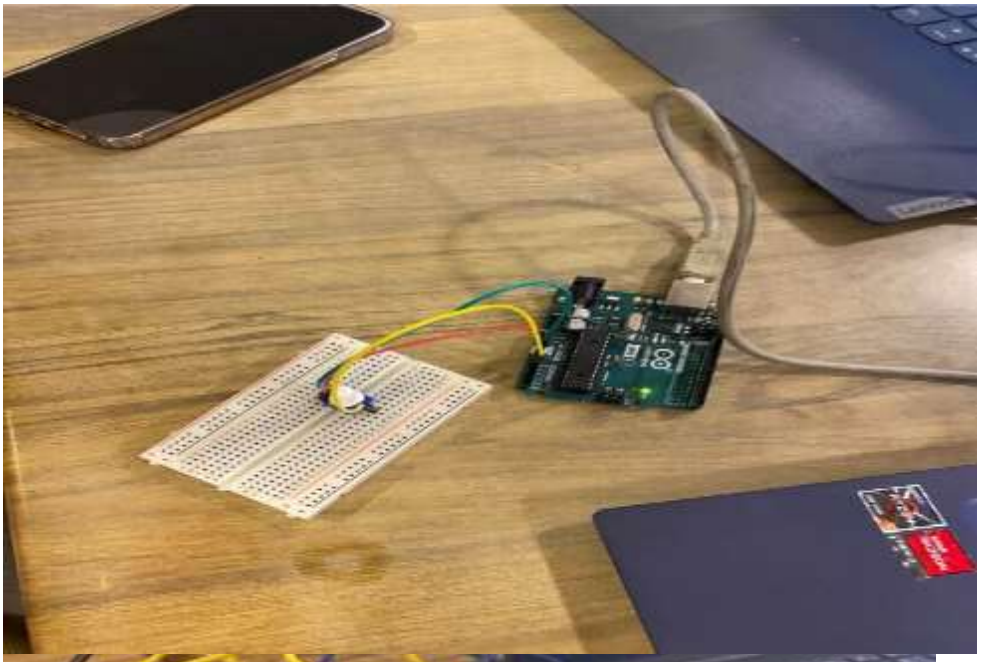
    string name,phone,med;
    int age;

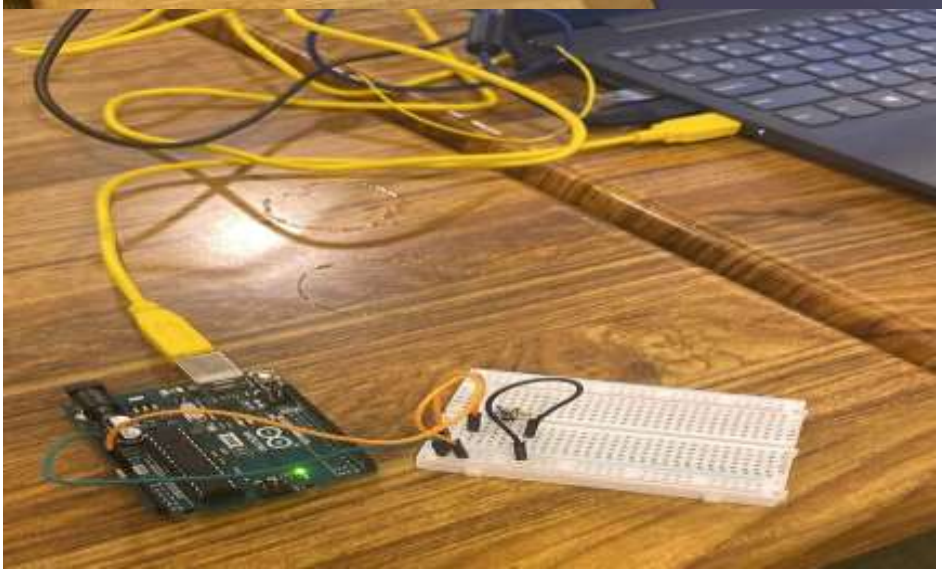
    cout<<"Name: ";
    cin>>name;
    cl[index].setName(name);
    cout<<"Age: ";
    cin>>age;
    cl[index].setAge(age);
    cout<<"phone: ";
      cin>>phone;
      cl[index].setPhone(phone);
      cout<<"Medicine Given: ";
      cin>>med;
      cl[index].setMedicine(med);
      cout<<"Patient Successfully Added!"<<endl;

}
/*
function printArray(patient*,int)
takes parameters a patient pointer(the clinic array)
and an integer (the index)
once it is called it printe from zero to index the data that are
saved on the array.
*/
void printArray(patient *clinic, int index){
    cout<<"PRINTING ALL DATA"<<endl<<"--------------------"<<endl;
    for (int i=0;i<index;i++){
        cout<<"------ Patient "<<i+1<<"------"<<endl;
        cout<<clinic[i].getName()<<endl;
        cout<<clinic[i].getAge()<<endl;
        cout<<clinic[i].getPhone()<<endl;
        cout<<clinic[i].getMedicine()<<endl;
    }
}
```

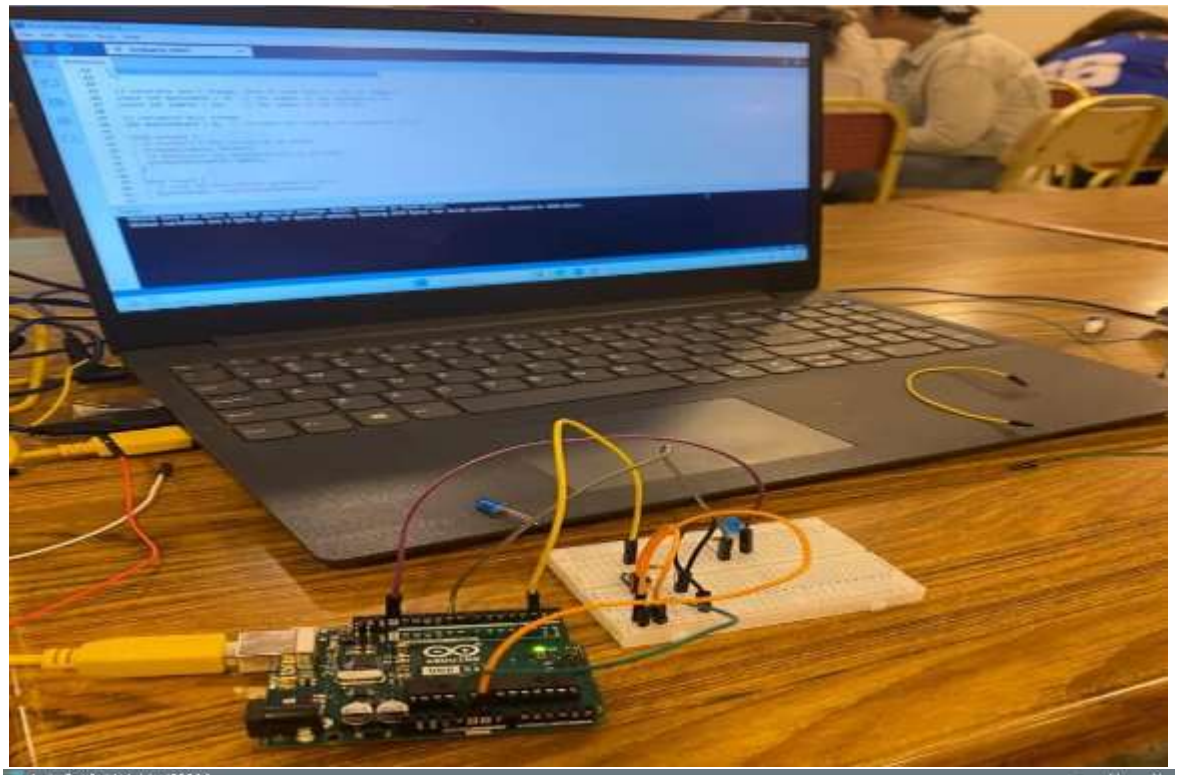
DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- **Practica in Arduino**



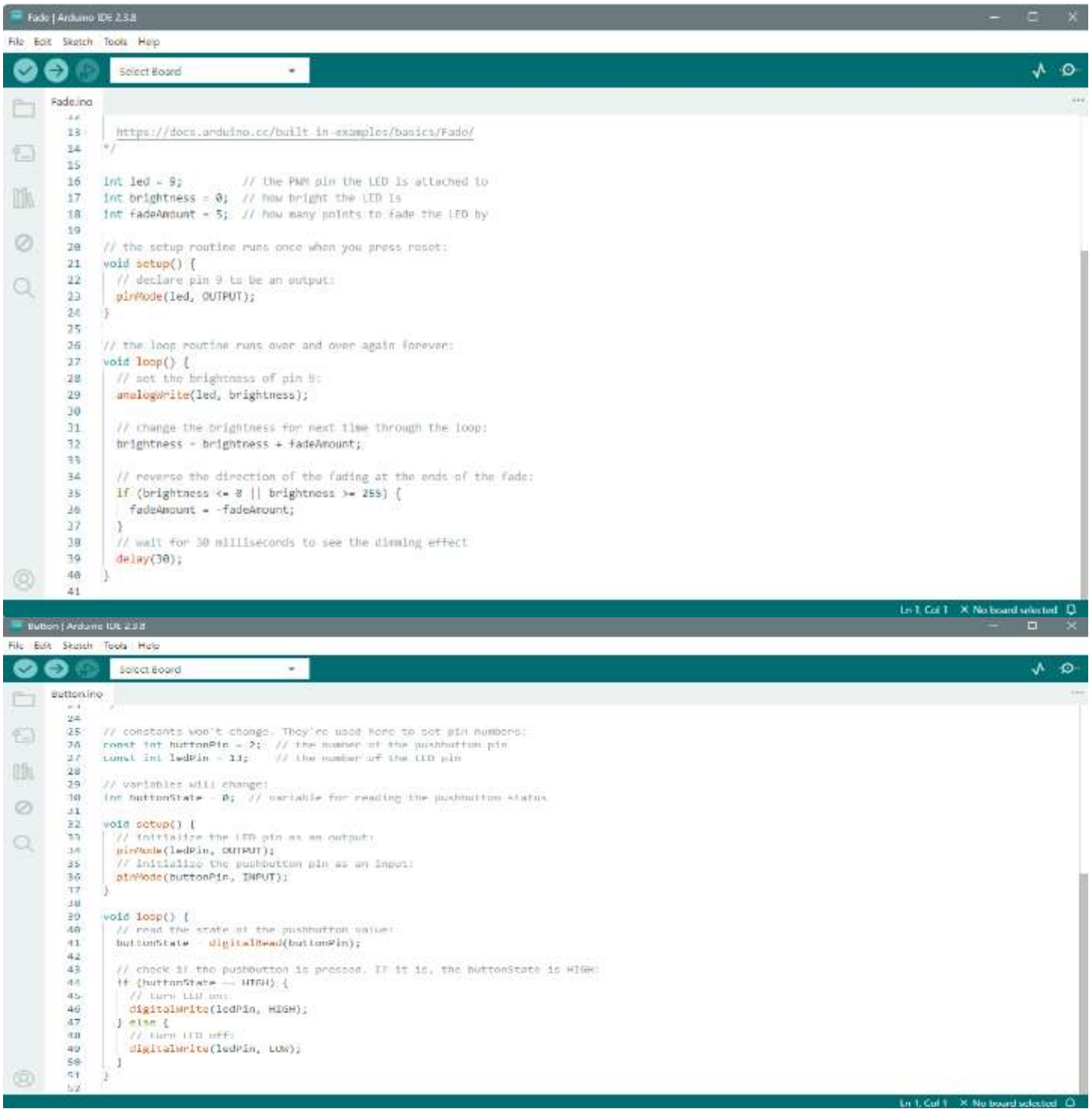


DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.





DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



DISCLAIMER: The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.