

4IT Erasmus+Project 2024-1-RO01-KA121-VET-000214264
Beneficiary Colegiul National Gheorghe Vranceanu, Bacau, Romania

Sending organization: Colegiul National ”Roman-Vodă” Roman, Romania

MOBILITY PORTFOLIO
Paphos, Cyprus

Mobility period:
PAPHOS, CYPRUS, 13.04.2025-27.04.2025 – practice period 14.04.2025-25.04.2025

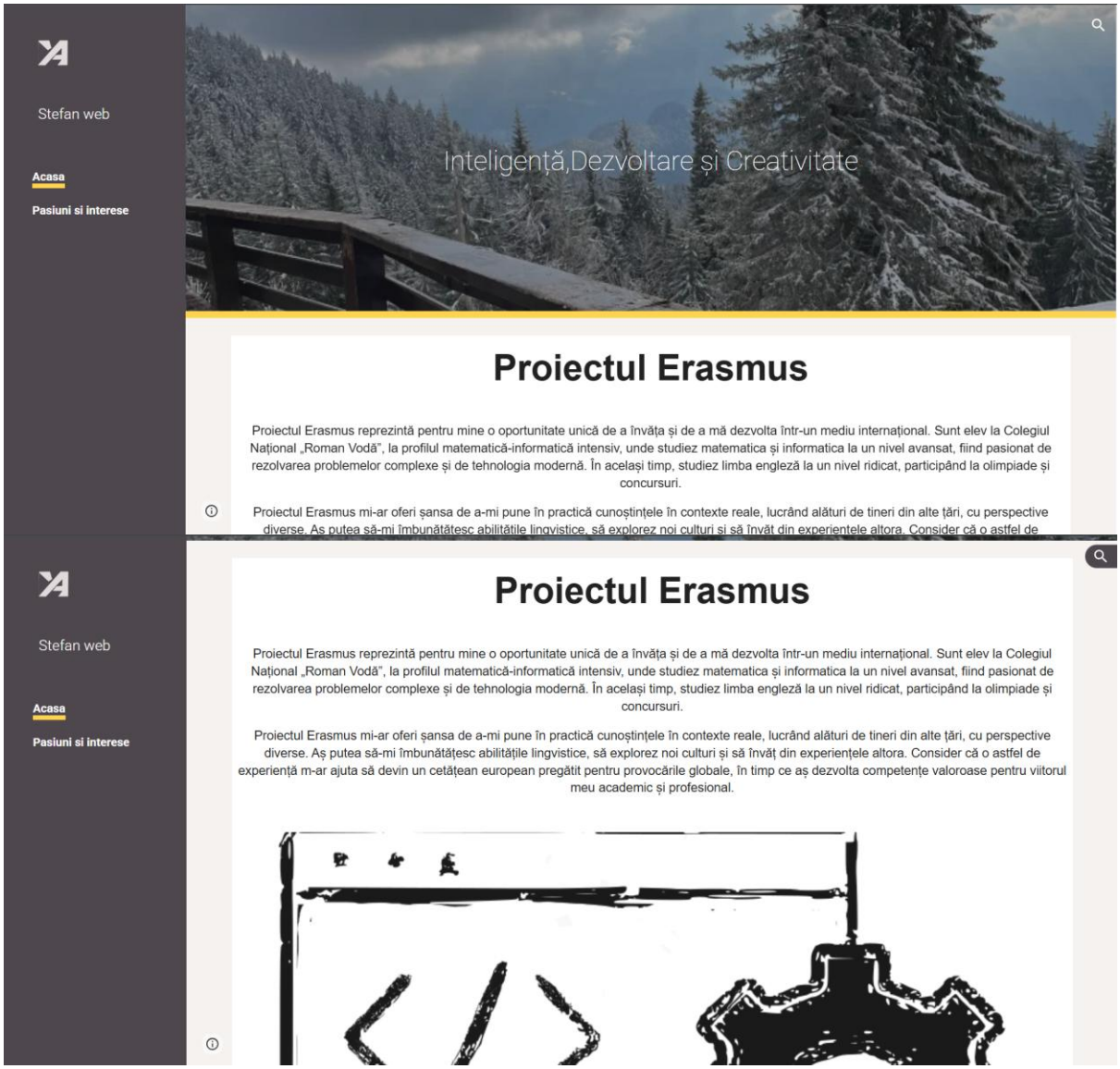
Receiving Lazarou ICT Professionals

Practice tutor: Lazaros Lazarou

Participant: Banu Stefan

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.

Stefan web - Personal website

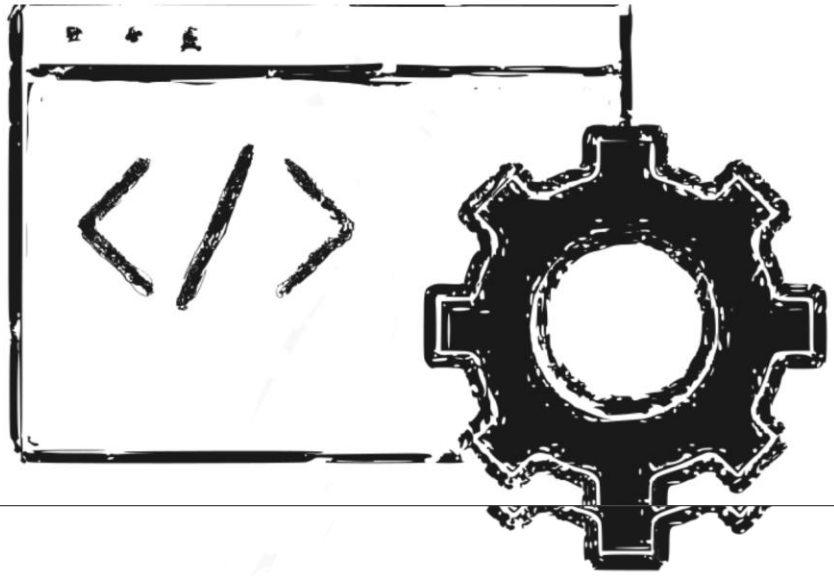


Stefan web

Acasa

Pasiuni si interese

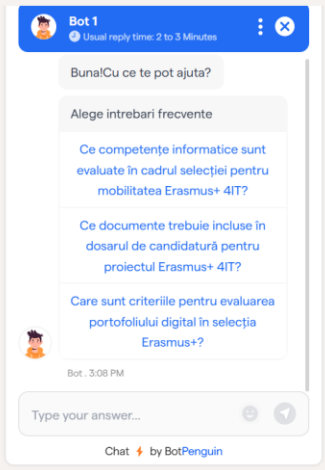
experiență m-ar ajuta să devin un cetățean european pregătit pentru provocările globale, în timp ce aș dezvolta competențe valoroase pentru viitorul meu academic și profesional.



Stefan web

Acasa

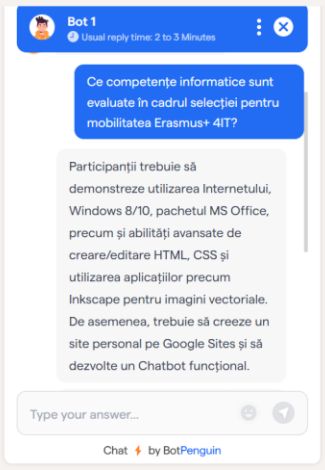
Pasiuni si interese



Stefan web

Acasa

Pasiuni si interese

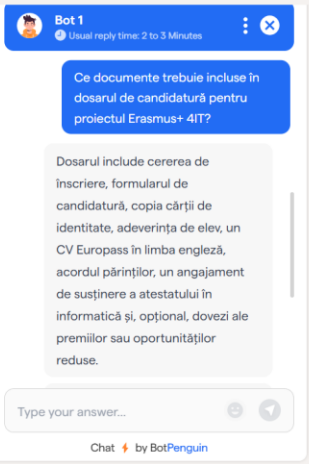


Stefan web

Acasa

Pasiuni si interese



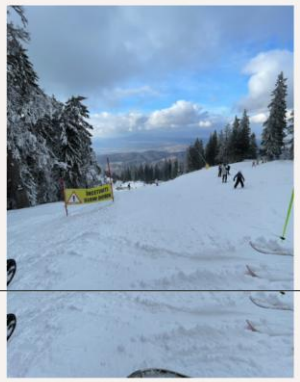


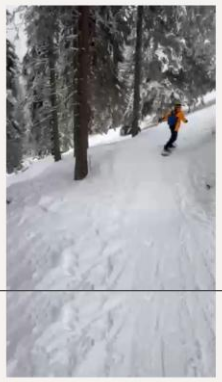
Stefan web

Acasa

Pasiuni si interese









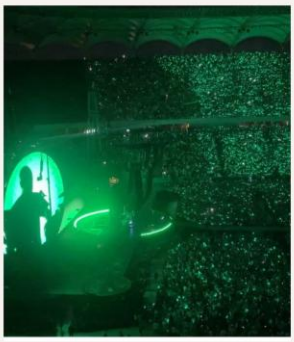
Snowboarding

Stefan web

Acasa

Pasiuni si interese

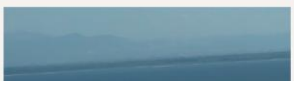






Concerte





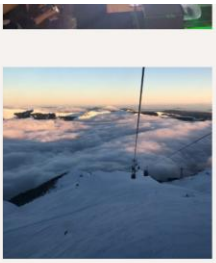


Peisaje

Stefan web

Acasa

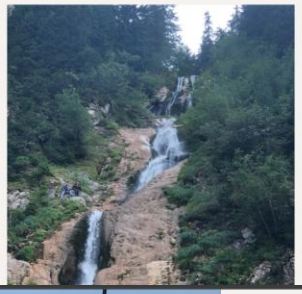
Pasiuni si interese





Peisaje

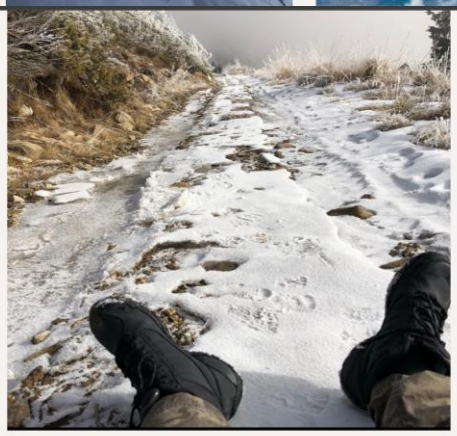




Stefan web

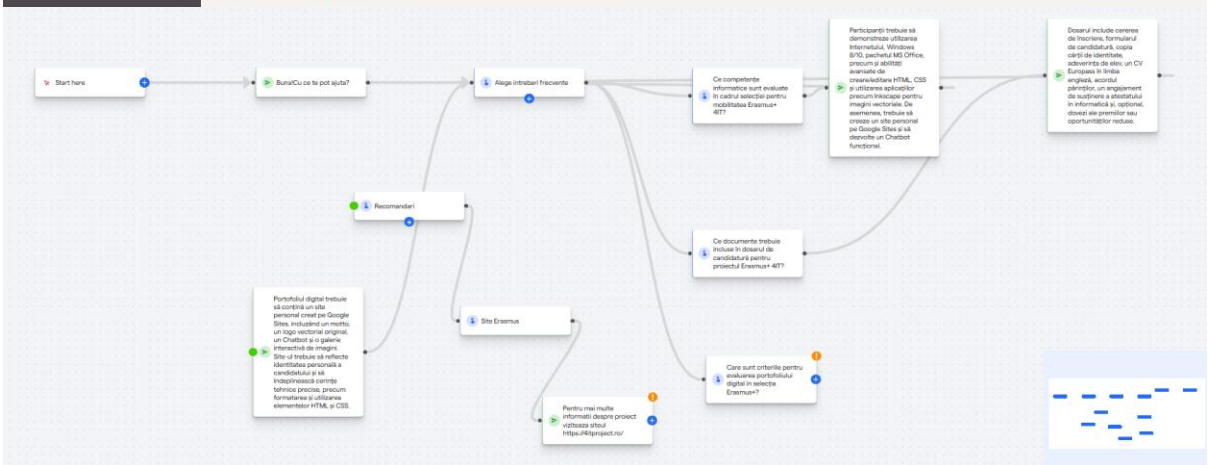
Acasa

Pasiuni si interese






COLEGIUL NAȚIONAL ROMAN VODĂ(C.N.R.V)



Teme realizate

Ziua 1 temă

```cpp
//////////////HOMEWORK DAY 1/////////////////
#include <iostream>
using namespace std;

int main() {
    int capacity = 0;
    int size = 0;
    int number;

    cout << "The initial capacity of the array will be: ";
    cin >> capacity;

    char* array = new char[capacity];

    while (true) {
        if (size == capacity) {
            cout << "You have reached the end of capacity." << endl;
            cout << "Do you want to add more? (0 to stop / 1 to continue): ";
            int answer;
            cin >> answer; // citim răspunsul

            if (answer == 0) {
                break;
            } else {
                int extra = 0;
                cout << "How much more space do you want to add? ";
                cin >> extra;

                int newCapacity = capacity + extra;
                char* newArray = new char[newCapacity];

                for (int i = 0; i < size; i++) {
                    newArray[i] = array[i];
                }

                delete[] array;
                array = newArray;
                capacity = newCapacity;
```

```
        }

        cout << "Enter a number between 65 and 90 (-1 to stop): ";
        cin >> number;

        if (number == -1) {
            break;
        }

        if (number >= 65 && number <= 90) {
            array[size] = (char)number;
            size++;
        } else {
            cout << "Invalid input. Please enter a number between 65 and 90." << endl;
        }
    }

    cout << "\nCharacters you entered in ASCII: ";
    for (int i = 0; i < size; i++) {
        cout << (char)array[i] << " ";
    }

    cout << endl;

    delete[] array;
    return 0;
}
```

Ziua 1 practică

```
int age;
int *aPtr; (saving a memory address)
int *bPtr; (saves the address of b )
age=5
aPtr=&age; (saves age inside aPtr)
cout<<age;//result is 5;

cout<<aPtr;// 0x 75AB34
cout<<*aPtr; //result:5



//the program;


#include<iostream>
using namespace std;

int main(){
    int age;
    char b;
    int *aPtr;
    char *bPtr;
    age=5;
    aPtr=&age;
    cout<<age;
    cout<<endl;
    cout<<aPtr;
    cout<<endl;
    cout<<*aPtr;
}



////////////////////////////////
```

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

int main(){

int *dinamicarray;//pointer to an integer,which is going to be dinamic array;
int arSize=0;
cout<<"enter the size of the array :";
cin>>arSize;
dinamicarray=new int[arSize];//the memory allocated for arsize integers
for(int i=0;i<arSize;i++){//use the for loop to fill in the dynamicarray with 50 integers
    dinamicarray[i]=i+1;

}
cout<<endl;
for(int i=0;i<arSize;i++){// display the contents of the dynamicarray
    cout<<dinamicarray[i]<<"-";

}
cout<<endl;
delete dinamicarray; //release the dynamicarray allocated space back to the operating system
return 0;

}

//////////////////////////


for(int i=0;i<arSize;i++){// display the contents of the dynamicarray
    cout<<dinamicarray[i]<<"-";

}
cout<<endl;
delete dinamicarray; //release the dynamicarray allocated space back to the operating system
return 0;

}

//////////////////////////////




//65-90 ascii alfabet



/*write a program where you will use a dynamic array to savre characters
the user will give you numbers(intrgers ) that range from 65 to 90 as soon as
you read the number you will ocnvert it ti character and save ity into a dynamic array
,your program will be able to make the array bigger if it is full at the
end you will display your result that you have read from the user

*/
```

Ziua 2 temă

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

int main(){

int *dinamicarray;//pointer to an integer,which is going to be dinamic array;
int arSize=0;
cout<<"enter the size of the array :";
cin>>arSize;
dinamicarray=new int[arSize];//the memory allocated for arsize integers
for(int i=0;i<arSize;i++){//use the for loop to fill in the dynamicarray with 50 integers
    dinamicarray[i]=i+1;

}
cout<<endl;
for(int i=0;i<arSize;i++){// display the contents of the dynamicarray
    cout<<dinamicarray[i]<<"-";

}
cout<<endl;
delete dinamicarray; //release the dynamicarray allocated space back to the operating system
return 0;

}

//////////////////////////
```

Ziua 2 practică

```cpp
/////////////////////////////////////////////*
WRITE A PROGRAM TO COLLECT NAME ANT TELEPHONE NUMBER from students using 2 dynamic arrays (parallel arrays using simple notions;

*/
/////////////////////////////////////////////

#include <iostream>
using namespace std;
int main(){
 int count;
 cout<<"How many students do you want to enter? ";
 cin>>count;

 string* names=new string[count];
 string* phonenumbers=new string[count];
 for(int i=0;i<count;i++){
    cout<<"enter a name for student"<<i+1<<":";
    cin>>names[i];

    cout<<"enter a the phone number of student "<<i+1<<":";
    cin>>phonenumbers[i];

 }
 cout<<"\nStudent list:\n";
 for(int i=0;i<count;i++){
    cout<<"Name:"<<names[i]<<" "<<"Phone number:"<<" "<<phonenumbers[i]<<endl;

 }

 //clean  up
 delete[] names;
 delete[] phonenumbers;
 return 0;
}

////////////////////////////////
```

```cpp
//////////////////adaugam persoane+date///////////
#include <iostream>
using namespace std;

int main() {
    int count;
    cout << "How many students do you want to enter? ";
    cin >> count;

    int capacity = count;
    int size = 0;

    string* names = new string[capacity];
    string* phonenumbers = new string[capacity];

    while (true) {
        if (size == capacity) {
            cout << "Out of space. Do you want to add more? (0 to stop / 1 to continue): ";
            int answer;
            cin >> answer;

            if (answer == 0) {
                break;
            } else {
                int extra;
                cout << "How many students do you want to add? ";
                cin >> extra;

                int newcapacity = capacity + extra;
                string* newnames = new string[newcapacity];
                string* newphonenumbers = new string[newcapacity];

                for (int i = 0; i < size; i++) {
                    newnames[i] = names[i];
                    newphonenumbers[i] = phonenumbers[i];
                }

                for (int i = 0; i < size; i++) {
                    newnames[i] = names[i];
                    newphonenumbers[i] = phonenumbers[i];
                }

                delete[] names;
                delete[] phonenumbers;

                names = newnames;
                phonenumbers = newphonenumbers;
                capacity = newcapacity;
            }
        }

        cout << "Enter name for student " << size + 1 << ": ";
        cin >> names[size];

        cout << "Enter phone number for student " << size + 1 << ": ";
        cin >> phonenumbers[size];

        size++;
    }

    cout << "\nStudent list:\n";
    for (int i = 0; i < size; i++) {
        cout << "Name: " << names[i] << " | Phone number: " << phonenumbers[i] << endl;
    }

    // Clean up
    delete[] names;
    delete[] phonenumbers;

    return 0;
}
//////////////////////////////////cauta numele si afiseaza numarul de telefon in functie de nume///////////
#include <iostream>
```

```
}
//////////////////////////////////cauta numele si afiseaza numarul de telefon in functie de nume//////////
#include <iostream>
using namespace std;

int main() {
    int count;
    cout << "How many students do you want to enter? ";
    cin >> count;

    int capacity = count;
    int size = 0;

    string* names = new string[capacity];
    string* phonenumbers = new string[capacity];

    while (true) {
        if (size == capacity) {
            cout << "Out of space. Do you want to add more? (0 to stop / 1 to continue): ";
            int answer;
            cin >> answer;

            if (answer == 0) {
                break;
            } else {
                int extra;
                cout << "How many students do you want to add? ";
                cin >> extra;

                int newcapacity = capacity + extra;
                string* newnames = new string[newcapacity];
                string* newphonenumbers = new string[newcapacity];

                for (int i = 0; i < size; i++) {
                    newnames[i] = names[i];
                    newphonenumbers[i] = phonenumbers[i];
                }
                }

                delete[] names;
                delete[] phonenumbers;

                names = newnames;
                phonenumbers = newphonenumbers;
                capacity = newcapacity;
            }
        }

        cout << "Enter name for student " << size + 1 << ": ";
        cin >> names[size];

        cout << "Enter phone number for student " << size + 1 << ": ";
        cin >> phonenumbers[size];

        size++;
    }

    cout << "\nStudent list:\n";
    for (int i = 0; i < size; i++) {
        cout << "Name: " << names[i] << " | Phone number: " << phonenumbers[i] << endl;
    }

//SEARCH SECTION:
cout<<"DO you want to search a phone number by introducing the name?(0=no/1=yes)";
int searchuser;
cin>>searchuser;
while(searchuser==1){
    string searchname;
    cout<<"enter the name you search for";
    cin>>searchname;
    bool found=false;
    for(int i=0;i<size;i++){
        if(names[i]==searchname){
            cout<<"the phone number of "<<searchname<<" is : "<<phonenumbers[i]<<endl;
```

```
//SEARCH SECTION:
cout<<"DO you want to search a phone number by introducing the name?(0=no/1=yes)";
int searchuser;
cin>>searchuser;
while(searchuser==1){
   string searchname;
   cout<<"enter the name you search for";
   cin>>searchname;
   bool found=false;
   for(int i=0;i<size;i++){
      if(names[i]==searchname){
         cout<<"the phone number of "<<searchname<<" is : "<<phonenumbers[i]<<endl;
         found=true;
         break;
      }
   }
   if(!found){
      cout<<"name not found on the list"<<endl;

   }

cout<<"do you waht to search again??(0 to stop/1 to continue)";
cin>>searchuser;
}
    // Clean up
    delete[] names;
    delete[] phonenumbers;

    return 0;
}
```

```
////////////////////////////////////////////////////////////////////////////////
```

```
}
    // Clean up
    delete[] names;
    delete[] phonenumbers;

    return 0;
}
```

```
////////////////////////////////////////////////////////////////////////////////////////////////
```

```
//////////////////////argv+argc
#include <iostream>
using namespace std;
int main(int argc,char* argv[])//argc ->argument count
{
    cout<<"you have entered"<<argc
    <<"arguments:"<<endl;
    //using a while loop to iterate through arguments
    int i=0;
    while(i<argc){
        cout<<"argument "<<i+1<<": "<<argv[i]<<"\n";
        i++;
    }
    return 0;
}
```

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

int main() {
    int count;
    cout << "How many students do you want to enter? ";
    cin >> count;

    int capacity = count;
    int size = 0;

    string* names = new string[capacity];
    string* phonenumbers = new string[capacity];

    while (true) {
        if (size == capacity) {
            cout << "Out of space. Do you want to add more? (0 to stop / 1 to continue): ";
            int answer;
            cin >> answer;

            if (answer == 0) {
                break;
            } else {
                int extra;
                cout << "How many students do you want to add? ";
                cin >> extra;

                int newcapacity = capacity + extra;
                string* newnames = new string[newcapacity];
                string* newphonenumbers = new string[newcapacity];

                for (int i = 0; i < size; i++) {
                    newnames[i] = names[i];
                    newphonenumbers[i] = phonenumbers[i];
                }

                delete[] names;
                delete[] phonenumbers;
```

```cpp
                names = newnames;
                phonenumbers = newphonenumbers;
                capacity = newcapacity;
            }
        }

        cout << "Enter name for student " << size + 1 << ": ";
        cin >> names[size];

        cout << "Enter phone number for student " << size + 1 << ": ";
        cin >> phonenumbers[size];

        size++;
    }

    cout << "\nStudent list:\n";
    for (int i = 0; i < size; i++) {
        cout << "Name: " << names[i] << " | Phone number: " << phonenumbers[i] << endl;
    }

    // SEARCH SECTION TO FIND A PHONE NUMBER BY NAME
    cout << "\nDo you want to search a phone number by entering the name? (0 = no / 1 = yes): ";
    int searchuser;
    cin >> searchuser;

    while (searchuser == 1) {
        string searchname;
        cout << "Enter the name you want to search for: ";
        cin >> searchname;

        bool found = false;
        for (int i = 0; i < size; i++) {
            if (names[i] == searchname) {
                cout << "The phone number of " << searchname << " is: " << phonenumbers[i] << endl;
                found = true;
                break;
            }
        }
```

```
        if (!found) {
            cout << "Name not found in the list." << endl;
        }

        cout << "Do you want to search again? (0 = no / 1 = yes): ";
        cin >> searchuser;
    }

    // SEARCH SECTION TO FIND A NAME BY PHONE NUMBER
    cout << "\nDo you want to search a name by entering the phone number? (0 = no / 1 = yes): ";
    int searchphonenumber;
    cin >> searchphonenumber;

    while (searchphonenumber == 1) {
        string searchnumber;
        cout << "Enter the phone number you want to search for: ";
        cin >> searchnumber;

        bool found = false;
        for (int i = 0; i < size; i++) {
            if (phonenumbers[i] == searchnumber) {
                cout << "The name of " << searchnumber << " is: " << names[i] << endl;
                found = true;
                break;
            }
        }

        if (!found) {
            cout << "Phone number not found in the list." << endl;
        }

        cout << "Do you want to search again? (0 = no / 1 = yes): ";
        cin >> searchphonenumber;
    }


    delete[] names;
    delete[] phonenumbers;
            }
        }

        if (!found) {
            cout << "Phone number not found in the list." << endl;
        }

        cout << "Do you want to search again? (0 = no / 1 = yes): ";
        cin >> searchphonenumber;
    }


    delete[] names;
    delete[] phonenumbers;

    return 0;
}
```

Ziua 3 practică

```
//////////////////////////////////////////////////////////////////////


#include <iostream>
#include <string.h>
using namespace std;
int main(int argc,char* argv[])//argc ->argument count
{
    cout<<"you have entered"<<argc
    <<"arguments:"<<endl;
    //using a while loop to iterate through arguments
    int i=1;
    while(i<argc){
        cout<<"argument "<<i+1<<": "<<argv[i]<<"\n";
           if strcmp(argv[1],"/h"){

              //write code to show help
           }

              if strcmp(argv[1],"/s"){
                 //write code to implement to show /s option

              }
                 if strcmp(argv[1],"/t"){


                     //write code to implement to show /t option

                 }
        i++;
    }
    // /h /s /t
    if strcmp(argv[1],"/h")

    return 0;
}


//////////////////////////////CLASE ///////////////


#include <iostream>
using namespace std;

class student{
    private:
        string name;
        string lname;
        float exam1;
        float exam2;
    public:
        student(){//constructor: We use constructors to initialize the object's variables
            name="EMPTY";
            lname="EMPTY";
            exam1=1.1;
            exam2=1.1;
        }
    void setName(string n){  name=n; }
    void setLname(string l){ lname=l; }
    void setExam1(float e1){ exam1=e1;}
    void setExam2(float e2){ exam2=e2;}
    string getName(){ return name;}
    string getLname(){ return lname;}
    float getExam1(){ return exam1;}
    float getExam2(){ return exam2;}
};

int main(){
    const int MAXSTUDENTS=5;
    student myclass[MAXSTUDENTS];
    string tname="", tlname="";
    float texam1=0.0, texam2=0.0;

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

```
    float getExam1(){ return exam1;}
    float getExam2(){ return exam2;}
};

int main(){
    const int MAXSTUDENTS=5;
    student myclass[MAXSTUDENTS];
    string tname="", tlname="";
    float texam1=0.0, texam2=0.0;

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

        cout<<"Name "<<i<<": ";
        cin>>tname;
        myclass[i].setName(tname);

        cout<<"Last Name "<<i<<": ";
        cin>>tlname;
        myclass[i].setLname(tlname);

        cout<<"1st Exam "<<i<<": ";
        cin>>texam1;
        myclass[i].setExam1(texam1);


        cout<<"2nd Exam "<<i<<": ";
        cin>>texam2;
        myclass[i].setExam2(texam2);

    }
    cout<<endl<<endl<<endl;
    cout<<"---------YOU HAVE SAVED THE FOLLOWING INFO ------------"<<endl;
    for (int i=0;i<MAXSTUDENTS;i++){

        cout<<myclass[i].getName()<<" "<<myclass[i].getLname()<<endl;
        cout<<"Exam 1: "<<myclass[i].getExam1()<<" - Exam2: "<<myclass[i].getExam2()<<endl;
    }
    return 0;
        myclass[i].setExam2(texam2);

    }
    cout<<endl<<endl<<endl;
    cout<<"---------YOU HAVE SAVED THE FOLLOWING INFO ------------"<<endl;
    for (int i=0;i<MAXSTUDENTS;i++){

        cout<<myclass[i].getName()<<" "<<myclass[i].getLname()<<endl;
        cout<<"Exam 1: "<<myclass[i].getExam1()<<" - Exam2: "<<myclass[i].getExam2()<<endl;
    }
    return 0;
}
/*
-EXERCISE
Write a program using a class which is going to take information
about cars save them inside an array of objects.
Your program must also print everything you have saved inside the array.
You will collect the following information from the user:
Car Brand
Car Model
Plate number (ex. ABC 999)
Name of owner

-EXERCISE
Write a program for a supermarket which is going to
take the information for the different products a supermarket sells
and it will save the following information:
-Brand of product
-Type of product
-Price
-Stock Remaining
-Barcode->return the brand,price,stock of the product
Your program must be able to search for products by name and by Type
*/
```

Ziua 3 temă

PROGAM 1

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

class Car {
private:
    string brand;
    string model;
    string plate;
    string owner;
public:
    Car() {
        brand = "EMPTY";
        model = "EMPTY";
        plate = "EMPTY";
        owner = "EMPTY";
    }

    void setBrand(string b) { brand = b; }
    void setModel(string m) { model = m; }
    void setPlate(string p) { plate = p; }
    void setOwner(string o) { owner = o; }

    string getBrand() { return brand; }
    string getModel() { return model; }
    string getPlate() { return plate; }
    string getOwner() { return owner; }
};

int main() {
    const int MAXCARS = 3;
    Car garage[MAXCARS];

    string tbrand, tmodel, tplate, towner;

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

```cpp
    for (int i = 0; i < MAXCARS; i++) {
        cout << "Car " << i + 1 << " Brand: ";
        cin >> tbrand;
        garage[i].setBrand(tbrand);

        cout << "Car " << i + 1 << " Model: ";
        cin >> tmodel;
        garage[i].setModel(tmodel);

        cout << "Car " << i + 1 << " Plate Number: ";
        cin.ignore(); // pentru spații în input
        getline(cin, tplate);
        garage[i].setPlate(tplate);

        cout << "Car " << i + 1 << " Owner Name: ";
        getline(cin, towner);
        garage[i].setOwner(towner);

        cout << endl;
    }

    cout << "\n------ SAVED CAR DATA ------\n";
    for (int i = 0; i < MAXCARS; i++) {
        cout << "Car " << i + 1 << ": " << garage[i].getBrand() << " " << garage[i].getModel() << endl;
        cout << "Plate: " << garage[i].getPlate() << " | Owner: " << garage[i].getOwner() << endl;
        cout << "----------------------------\n";
    }

    return 0;
}
```

PROGRAM 2

```cpp
#include <iostream>
```

PROGRAM 2

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

class Product {
private:
    string brand;
    string type;
    float price;
    int stock;
    string barcode;
public:
    Product() {
        brand = "EMPTY";
        type = "EMPTY";
        price = 0.0;
        stock = 0;
        barcode = "000000";
    }

    void setBrand(string b) { brand = b; }
    void setType(string t) { type = t; }
    void setPrice(float p) { price = p; }
    void setStock(int s) { stock = s; }
    void setBarcode(string bc) { barcode = bc; }

    string getBrand() { return brand; }
    string getType() { return type; }
    float getPrice() { return price; }
    int getStock() { return stock; }
    string getBarcode() { return barcode; }
};

int main() {
    const int MAXPRODUCTS = 5;
```

```cpp
int main() {
    const int MAXPRODUCTS = 5;
    Product store[MAXPRODUCTS];

    string tbrand, ttype, tbarcode;
    float tprice;
    int tstock;

    // Introducere produse
    for (int i = 0; i < MAXPRODUCTS; i++) {
        cout << "Product " << i + 1 << " Brand: ";
        cin >> tbrand;
        store[i].setBrand(tbrand);

        cout << "Product " << i + 1 << " Type: ";
        cin >> ttype;
        store[i].setType(ttype);

        cout << "Product " << i + 1 << " Price: ";
        cin >> tprice;
        store[i].setPrice(tprice);

        cout << "Product " << i + 1 << " Stock Remaining: ";
        cin >> tstock;
        store[i].setStock(tstock);

        cout << "Product " << i + 1 << " Barcode: ";
        cin >> tbarcode;
        store[i].setBarcode(tbarcode);

        cout << endl;
    }

    // Afișare produse
    cout << "\n------ SAVED PRODUCT DATA ------\n";
    for (int i = 0; i < MAXPRODUCTS; i++) {
        cout << "Product " << i + 1 << ": " << store[i].getBrand() << " - " << store[i].getType() << endl;
        cout << "Price: $" << store[i].getPrice() << " | Stock: " << store[i].getStock() << " units" << endl;
```

```
        cout << endl;
    }

    // Afișare produse
    cout << "\n------ SAVED PRODUCT DATA ------\n";
    for (int i = 0; i < MAXPRODUCTS; i++) {
        cout << "Product " << i + 1 << ": " << store[i].getBrand() << " - " << store[i].getType() << endl;
        cout << "Price: $" << store[i].getPrice() << " | Stock: " << store[i].getStock() << " units" << endl;
        cout << "Barcode: " << store[i].getBarcode() << endl;
        cout << "-----------------------------\n";
    }

    // Căutare după cod de bare
    string searchBarcode;
    cout << "\nEnter a barcode to search for product info (brand, price, stock): ";
    cin >> searchBarcode;

    bool found = false;
    for (int i = 0; i < MAXPRODUCTS; i++) {
        if (store[i].getBarcode() == searchBarcode) {
            cout << "\nProduct Found!" << endl;
            cout << "Brand: " << store[i].getBrand() << endl;
            cout << "Price: $" << store[i].getPrice() << endl;
            cout << "Stock: " << store[i].getStock() << " units" << endl;
            found = true;
            break;
        }
    }

    if (!found) {
        cout << "\nNo product found with barcode: " << searchBarcode << endl;
    }

    return 0;
}
```

Ziua 4 temă

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

const int MAXOBJS = 4;

class Car {
private:
    string brand;
    string model;
    string plateNum;
    string owner;
public:
    Car() : brand("EMPTY"), model("EMPTY"), plateNum("EMPTY"), owner("EMPTY") {}

    void setBrand(const string& B) { brand = B; }
    void setModel(const string& M) { model = M; }
    void setPlate(const string& P) { plateNum = P; }
    void setOwner(const string& N) { owner = N; }

    string getBrand() const { return brand; }
    string getModel() const { return model; }
    string getPlate() const { return plateNum; }
    string getOwner() const { return owner; }

    void display() const {
        cout << "Brand: " << brand << endl;
        cout << "Model: " << model << endl;
        cout << "Plate: " << plateNum << endl;
        cout << "Owner: " << owner << endl;
    }
};
```

```
};

// Subprogram pentru introducerea datelor
void inputCars(Car cars[]) {
    string userInput, tmp2;
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- CAR " << i + 1 << " ----------" << endl;

        cout << "Car Brand: ";
        cin.ignore();
        getline(cin, userInput);
        cars[i].setBrand(userInput);

        cout << "Car Model: ";
        getline(cin, userInput);
        cars[i].setModel(userInput);

        cout << "Plate Number (ex. ABC 123): ";
        cin >> userInput >> tmp2;
        userInput += " " + tmp2;
        cars[i].setPlate(userInput);
        cin.ignore();

        cout << "Owner Name: ";
        getline(cin, userInput);
        cars[i].setOwner(userInput);
    }
}

// Subprogram pentru afișare
void displayCars(const Car cars[]) {
    cout << "\n---------- SAVED CAR INFO ----------" << endl;
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- CAR " << i + 1 << " ----------" << endl;
        cars[i].display();
    }
}
```

```
}

// Subprograme de căutare
void searchByBrand(const Car cars[]) {
    string query;
    cout << "Which brand are you looking for? ";
    getline(cin, query);
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getBrand() == query)
            cars[i].display();
}

void searchByModel(const Car cars[]) {
    string query;
    cout << "Which model are you looking for? ";
    getline(cin, query);
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getModel() == query)
            cars[i].display();
}

void searchByOwner(const Car cars[]) {
    string query;
    cout << "Which owner are you looking for? ";
    getline(cin, query);
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getOwner() == query)
            cars[i].display();
}

void searchByPlate(const Car cars[]) {
    string part1, part2;
    cout << "Which plate number are you looking for? ";
    cin >> part1 >> part2;
    cin.ignore();
    string query = part1 + " " + part2;
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getPlate() == query)
```

```
void searchByPlate(const Car cars[]) {
    string part1, part2;
    cout << "Which plate number are you looking for? ";
    cin >> part1 >> part2;
    cin.ignore();
    string query = part1 + " " + part2;
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getPlate() == query)
            cars[i].display();
}

// Subprogram pentru meniu de căutare
void searchMenu(const Car cars[]) {
    int option = -1;

    while (option != 0) {
        cout << "\n========== SEARCH MENU ==========" << endl;
        cout << "1 - Search by brand" << endl;
        cout << "2 - Search by model" << endl;
        cout << "3 - Search by owner" << endl;
        cout << "4 - Search by plate number" << endl;
        cout << "0 - Exit" << endl;
        cout << "Option: ";
        cin >> option;
        cin.ignore();

        switch (option) {
            case 1: searchByBrand(cars); break;
            case 2: searchByModel(cars); break;
            case 3: searchByOwner(cars); break;
            case 4: searchByPlate(cars); break;
            case 0: cout << "Exiting search...\n"; break;
            default: cout << "Invalid option. Try again!\n"; break;
        }
    }
}
```

```
        cin.ignore();

        switch (option) {
            case 1: searchByBrand(cars); break;
            case 2: searchByModel(cars); break;
            case 3: searchByOwner(cars); break;
            case 4: searchByPlate(cars); break;
            case 0: cout << "Exiting search...\n"; break;
            default: cout << "Invalid option. Try again!\n"; break;
        }
    }
}

int main() {
    Car carsArray[MAXOBJS];

    inputCars(carsArray);
    displayCars(carsArray);
    searchMenu(carsArray);

    cout << "\nProgram ended. Goodbye!\n";
    return 0;
}
```

Ziua 4 practică

```
PROGAM 1


#include <iostream>
using namespace std;

class Car {
private:
    string brand;
    string model;
    string plate;
    string owner;
public:
    Car() {
        brand = "EMPTY";
        model = "EMPTY";
        plate = "EMPTY";
        owner = "EMPTY";
    }

    void setBrand(string b) { brand = b; }
    void setModel(string m) { model = m; }
    void setPlate(string p) { plate = p; }
    void setOwner(string o) { owner = o; }

    string getBrand() { return brand; }
    string getModel() { return model; }
    string getPlate() { return plate; }
    string getOwner() { return owner; }
};

int main() {
    const int MAXCARS = 3;
    Car garage[MAXCARS];

    string tbrand, tmodel, tplate, towner;
```

```
};

int main() {
    const int MAXCARS = 3;
    Car garage[MAXCARS];

    string tbrand, tmodel, tplate, towner;

    for (int i = 0; i < MAXCARS; i++) {
        cout << "Car " << i + 1 << " Brand: ";
        cin >> tbrand;
        garage[i].setBrand(tbrand);

        cout << "Car " << i + 1 << " Model: ";
        cin >> tmodel;
        garage[i].setModel(tmodel);

        cout << "Car " << i + 1 << " Plate Number: ";
        cin.ignore(); // pentru spații în input
        getline(cin, tplate);
        garage[i].setPlate(tplate);

        cout << "Car " << i + 1 << " Owner Name: ";
        getline(cin, towner);
        garage[i].setOwner(towner);

        cout << endl;
    }

    cout << "\n------ SAVED CAR DATA ------\n";
    for (int i = 0; i < MAXCARS; i++) {
        cout << "Car " << i + 1 << ": " << garage[i].getBrand() << " " << garage[i].getModel() << endl;
        cout << "Plate: " << garage[i].getPlate() << " | Owner: " << garage[i].getOwner() << endl;
        cout << "----------------------------\n";
    }

    return 0;
}
```

PROGRAM 2

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

class Product {
private:
    string brand;
    string type;
    float price;
    int stock;
    string barcode;
public:
    Product() {
        brand = "EMPTY";
        type = "EMPTY";
        price = 0.0;
        stock = 0;
        barcode = "000000";
    }

    void setBrand(string b) { brand = b; }
    void setType(string t) { type = t; }
    void setPrice(float p) { price = p; }
    void setStock(int s) { stock = s; }
    void setBarcode(string bc) { barcode = bc; }

    string getBrand() { return brand; }
    string getType() { return type; }
    float getPrice() { return price; }
    int getStock() { return stock; }
    string getBarcode() { return barcode; }
};
```

```cpp
int main() {
    const int MAXPRODUCTS = 5;
    Product store[MAXPRODUCTS];

    string tbrand, ttype, tbarcode;
    float tprice;
    int tstock;

    // Introducere produse
    for (int i = 0; i < MAXPRODUCTS; i++) {
        cout << "Product " << i + 1 << " Brand: ";
        cin >> tbrand;
        store[i].setBrand(tbrand);

        cout << "Product " << i + 1 << " Type: ";
        cin >> ttype;
        store[i].setType(ttype);

        cout << "Product " << i + 1 << " Price: ";
        cin >> tprice;
        store[i].setPrice(tprice);

        cout << "Product " << i + 1 << " Stock Remaining: ";
        cin >> tstock;
        store[i].setStock(tstock);

        cout << "Product " << i + 1 << " Barcode: ";
        cin >> tbarcode;
        store[i].setBarcode(tbarcode);

        cout << endl;
    }

    // Afișare produse
    cout << "\n------ SAVED PRODUCT DATA ------\n";
    for (int i = 0; i < MAXPRODUCTS; i++) {
        cout << "Product " << i + 1 << ": " << store[i].getBrand() << " - " << store[i].getType() << endl;
```

```cpp
        cout << endl;
    }

    // Afișare produse
    cout << "\n------ SAVED PRODUCT DATA ------\n";
    for (int i = 0; i < MAXPRODUCTS; i++) {
        cout << "Product " << i + 1 << ": " << store[i].getBrand() << " - " << store[i].getType() << endl;
        cout << "Price: $" << store[i].getPrice() << " | Stock: " << store[i].getStock() << " units" << endl;
        cout << "Barcode: " << store[i].getBarcode() << endl;
        cout << "-----------------------------\n";
    }

    // Căutare după cod de bare
    string searchBarcode;
    cout << "\nEnter a barcode to search for product info (brand, price, stock): ";
    cin >> searchBarcode;

    bool found = false;
    for (int i = 0; i < MAXPRODUCTS; i++) {
        if (store[i].getBarcode() == searchBarcode) {
            cout << "\nProduct Found!" << endl;
            cout << "Brand: " << store[i].getBrand() << endl;
            cout << "Price: $" << store[i].getPrice() << endl;
            cout << "Stock: " << store[i].getStock() << " units" << endl;
            found = true;
            break;
        }
    }

    if (!found) {
        cout << "\nNo product found with barcode: " << searchBarcode << endl;
    }

    return 0;
}
```

Ziua 5 practică

```
int Suma()
    {
        return a + b;
    }
```

```
///oop_2
int Max()
{
    int maxim = t[0];
    for (int i = 1; i < n; i++)
    {
        if (t[i] > maxim)
            maxim = t[i];
    }
    return maxim;
}

int Cauta(int val)
{
    for (int i = 0; i < n; i++)
    {
        if (t[i] == val)
            return i;
    }
    return -1;
}
```

```cpp
//oop_3
class Stiva
{
private:
    int v[100000];
    int maxv[100000]; // stivă pentru maxime
    int n;

public:
    Stiva()
    {
        n = 0;
    }

    void Push(int x)
    {
        v[n] = x;
        if (n == 0)
            maxv[n] = x;
        else
            maxv[n] = (x > maxv[n - 1] ? x : maxv[n - 1]);
        n++;
    }

    void Pop()
    {
        if (n > 0)
        {
            n--;
        }
    }

    int Top()
    {
        if (n == 0)
            return -1;
        return v[n - 1];
```

```cpp
    int Top()
    {
        if (n == 0)
            return -1;
        return v[n - 1];
    }

    int Max()
    {
        if (n == 0)
            return -1;
        return maxv[n - 1];
    }

    int Empty()
    {
        if (n == 0)
            return 1;
        return 0;
    }
};
```

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

// the function that gets a pointer to an array and finds a value

int searchInArray(int* array, int size, int valueToFind) {

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

        if ( *(array + i) == valueToFind) {

            return i; // Found,return the index
        }
    }
    return -1; // Was not found
}

int main() {
    int size;
    cout << "Introduce the array's dimension: ";
    cin >> size;

    // Declare the array
    int myArray[100]; //  max de 100 elemente

    cout << "Introduce " << size << " elements:\n";
    for (int i = 0; i < size; ++i) {
        cin >> myArray[i];
    }

    int value;
    cout << "Introduce the value that you want to read: ";
    cin >> value;

    // Send the pointer to the function
    int index = searchInArray(myArray, size, value);

    if (index != -1) {
        cout << "Value " << value << " was found at index" << index << "." << endl;
    } else {
        cout << "Value " << value << " wasn't found in the array" << endl;
    }

    return 0;
}
```

Ziua 6 temă

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

const int MAXOBJS = 4;

class Product {
private:
    string brand;
    string type;
    double price;
    int stock;
    string barcode;
public:
    Product() {
        brand = "empty";
        type = "empty";
        price = 0;
        stock = 0;
        barcode = "empty";
    }

    void setBrand(string b) { brand = b; }
    void setType(string t) { type = t; }
    void setPrice(double p) { price = p; }
    void setStock(int s) { stock = s; }
    void setBarcode(string c) { barcode = c; }

    string getBrand() { return brand; }
    string getType() { return type; }
    double getPrice() { return price; }
    int getStock() { return stock; }
    string getBarcode() { return barcode; }

    void display() {
        cout << "Brand: " << brand << endl;
        cout << "Type: " << type << endl;
        cout << "Price: $" << price << endl;
        cout << "Stock: " << stock << " units" << endl;
```

```
        barcode = "empty";
    }

    void setBrand(string b) { brand = b; }
    void setType(string t) { type = t; }
    void setPrice(double p) { price = p; }
    void setStock(int s) { stock = s; }
    void setBarcode(string c) { barcode = c; }

    string getBrand() { return brand; }
    string getType() { return type; }
    double getPrice() { return price; }
    int getStock() { return stock; }
    string getBarcode() { return barcode; }

    void display() {
        cout << "Brand: " << brand << endl;
        cout << "Type: " << type << endl;
        cout << "Price: $" << price << endl;
        cout << "Stock: " << stock << " units" << endl;
        cout << "Barcode: " << barcode << endl;
    }

    void displayShort() {
        cout << "Brand: " << brand << endl;
        cout << "Price: $" << price << endl;
        cout << "Stock: " << stock << " units" << endl;
    }
};

void inputProducts(Product products[]) {
    string text;
    double val;
    int num;

    cin.ignore();
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- PRODUCT " << i + 1 << " ----------" << endl;
    int num;

    cin.ignore();
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- PRODUCT " << i + 1 << " ----------" << endl;

        cout << "Brand: ";
        getline(cin, text);
        products[i].setBrand(text);

        cout << "Type: ";
        getline(cin, text);
        products[i].setType(text);

        cout << "Price: ";
        cin >> val;
        products[i].setPrice(val);

        cout << "Stock remaining: ";
        cin >> num;
        products[i].setStock(num);

        cin.ignore();

        cout << "Barcode: ";
        getline(cin, text);
        products[i].setBarcode(text);
    }
}

void displayProducts(Product products[]) {
    cout << "\n---------- PRODUCT LIST ----------" << endl;
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- PRODUCT " << i + 1 << " ----------" << endl;
        products[i].display();
    }
}
```

```
}

void searchByBrand(Product products[]) {
    string search;
    cout << "Enter brand to search: ";
    getline(cin, search);

    for (int i = 0; i < MAXOBJS; i++) {
        if (products[i].getBrand() == search) {
            products[i].display();
        }
    }
}

void searchByType(Product products[]) {
    string search;
    cout << "Enter product type to search: ";
    getline(cin, search);

    for (int i = 0; i < MAXOBJS; i++) {
        if (products[i].getType() == search) {
            products[i].display();
        }
    }
}

void searchByBarcode(Product products[]) {
    string search;
    cout << "Enter barcode to search: ";
    getline(cin, search);

    for (int i = 0; i < MAXOBJS; i++) {
        if (products[i].getBarcode() == search) {
            products[i].displayShort();
        }
    }
}
}

void searchMenu(Product products[]) {
    int opt;

    do {
        cout << "\n========== SEARCH MENU ==========" << endl;
        cout << "1 - Search by brand" << endl;
        cout << "2 - Search by type" << endl;
        cout << "3 - Search by barcode" << endl;
        cout << "0 - Exit" << endl;
        cout << "Option: ";
        cin >> opt;
        cin.ignore();

        if (opt == 1) searchByBrand(products);
        else if (opt == 2) searchByType(products);
        else if (opt == 3) searchByBarcode(products);
    } while (opt != 0);
}

int main() {
    Product products[MAXOBJS];

    inputProducts(products);
    displayProducts(products);
    searchMenu(products);

    cout << "\nProgram ended. Goodbye!\n";
    return 0;
}
```

Ziua 6 practică

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

const int MAXOBJS = 4;

class Car {
private:
    string brand;
    string model;
    string plateNum;
    string owner;
public:
    Car() {
        brand = "EMPTY";
        model = "EMPTY";
        plateNum = "EMPTY";
        owner = "EMPTY";
    }

    void setBrand(string B) { brand = B; }
    void setModel(string M) { model = M; }
    void setPlate(string P) { plateNum = P; }
    void setOwner(string N) { owner = N; }

    string getBrand() { return brand; }
    string getModel() { return model; }
    string getPlate() { return plateNum; }
    string getOwner() { return owner; }

    void display() {
        cout << "Brand: " << brand << endl;
        cout << "Model: " << model << endl;
        cout << "Plate: " << plateNum << endl;
        cout << "Owner: " << owner << endl;
    }
};

void inputCars(Car cars[]) {
    string userInput, tmp2;
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- CAR " << i + 1 << " ----------" << endl;

void inputCars(Car cars[]) {
    string userInput, tmp2;
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- CAR " << i + 1 << " ----------" << endl;

        cout << "Car Brand: ";
        cin.ignore();
        getline(cin, userInput);
        cars[i].setBrand(userInput);

        cout << "Car Model: ";
        getline(cin, userInput);
        cars[i].setModel(userInput);

        cout << "Plate Number (ex. ABC 123): ";
        cin >> userInput >> tmp2;
        userInput += " " + tmp2;
        cars[i].setPlate(userInput);
        cin.ignore();

        cout << "Owner Name: ";
        getline(cin, userInput);
        cars[i].setOwner(userInput);
    }
}

void displayCars(Car cars[]) {
    cout << "\n---------- SAVED CAR INFO ----------" << endl;
    for (int i = 0; i < MAXOBJS; i++) {
        cout << "\n---------- CAR " << i + 1 << " ----------" << endl;
        cars[i].display();
    }
}
```

```
}

void searchByBrand(Car cars[]) {
    string query;
    cout << "Which brand are you looking for? ";
    getline(cin, query);
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getBrand() == query)
            cars[i].display();
}

void searchByModel(Car cars[]) {
    string query;
    cout << "Which model are you looking for? ";
    getline(cin, query);
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getModel() == query)
            cars[i].display();
}

void searchByOwner(Car cars[]) {
    string query;
    cout << "Which owner are you looking for? ";
    getline(cin, query);
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getOwner() == query)
            cars[i].display();
}

void searchByPlate(Car cars[]) {
    string part1, part2;
    cout << "Which plate number are you looking for? ";
    cin >> part1 >> part2;
    cin.ignore();
    string query = part1 + " " + part2;
    for (int i = 0; i < MAXOBJS; i++)
        if (cars[i].getPlate() == query)
            cars[i].display();
}
```

```
        if (cars[i].getPlate() == query)
            cars[i].display();
}

void searchMenu(Car cars[]) {
    int option = -1;

    while (option != 0) {
        cout << "\n========== SEARCH MENU ==========" << endl;
        cout << "1 - Search by brand" << endl;
        cout << "2 - Search by model" << endl;
        cout << "3 - Search by owner" << endl;
        cout << "4 - Search by plate number" << endl;
        cout << "0 - Exit" << endl;
        cout << "Option: ";
        cin >> option;
        cin.ignore();

        switch (option) {
            case 1: searchByBrand(cars); break;
            case 2: searchByModel(cars); break;
            case 3: searchByOwner(cars); break;
            case 4: searchByPlate(cars); break;
            case 0: cout << "Exiting search...\n"; break;
            default: cout << "Invalid option. Try again!\n"; break;
        }
    }
}

int main() {
    Car carsArray[MAXOBJS];

    inputCars(carsArray);
    displayCars(carsArray);
    searchMenu(carsArray);

    cout << "\nProgram ended. Goodbye!\n";
    return 0;
}
```

Ziua 7 temă

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

// CREATE THE CLASS
class car {
private:
    string brand;
    string model;
    string plateNumber;
    string owner;
public:
    car() {
        brand = "EMPTY";
        model = "EMPTY";
        plateNumber = "EMPTY";
        owner = "EMPTY";
    }

    void setBrand(string B) { brand = B; }
    void setModel(string M) { model = M; }
    void setPlateNumber(string P) { plateNumber = P; }
    void setOwner(string O) { owner = O; }

    string getBrand() { return brand; }
    string getModel() { return model; }
    string getPlateNumber() { return plateNumber; }
    string getOwner() { return owner; }
};

// Function declarations
car* readData(car*, int, int&);
void displayCars(car*, int);
void searchByBrand(car*, int);
```

```cpp
int main() {
    int ARRSIZE = 4;
    int nextAvailPos = 0;
    car* carsArray = new car[ARRSIZE];

    int choice = -1;
    do {
        cout << "\nPlease Choose from the menu below:\n";
        cout << "1. Enter data\n";
        cout << "2. Display data\n";
        cout << "3. Search by brand\n";
        cout << "0. Exit\n";
        cout << "Option: ";
        cin >> choice;

        switch (choice) {
        case 1:
            carsArray = readData(carsArray, ARRSIZE, nextAvailPos);
            break;
        case 2:
            displayCars(carsArray, nextAvailPos);
            break;
        case 3:
            searchByBrand(carsArray, nextAvailPos);
            break;
        case 0:
            cout << "Exiting program...\n";
            break;
        default:
            cout << "Invalid option.\n";
        }
    } while (choice != 0);

    delete[] carsArray;
    return 0;
}
```

```
}

// Function to read data
car* readData(car* arr, int arrSize, int& nextPos) {
    int numCars = 0;
    string input;
    car* tmp;

    cout << "How many cars are you entering: ";
    cin >> numCars;

    for (int i = 0; i < numCars; i++) {
        if (nextPos >= arrSize) {
            int newSize = arrSize + numCars;
            tmp = new car[newSize];

            for (int j = 0; j < arrSize; j++) {
                tmp[j] = arr[j];
            }

            arrSize = newSize;
            delete[] arr;
            arr = tmp;
        }

        cout << "\n--- Entering data for car [" << i + 1 << "] ---\n";

        cout << "Brand: ";
        cin >> input;
        arr[nextPos].setBrand(input);

        cout << "Model: ";
        cin >> input;
        arr[nextPos].setModel(input);

        cout << "Plate number: ";
        cin >> input;
        arr[nextPos].setPlateNumber(input);

        cout << "Owner: ";
        cin >> input;
        arr[nextPos].setOwner(input);
```

```
        nextPos++;
    }

    return arr;
}

// Function to display all cars
void displayCars(car* arr, int count) {
    if (count == 0) {
        cout << "No cars to display.\n";
        return;
    }

    cout << "\n--- List of Cars ---\n";
    for (int i = 0; i < count; i++) {
        cout << "Car [" << i + 1 << "]:\n";
        cout << "  Brand: " << arr[i].getBrand() << endl;
        cout << "  Model: " << arr[i].getModel() << endl;
        cout << "  Plate Number: " << arr[i].getPlateNumber() << endl;
        cout << "  Owner: " << arr[i].getOwner() << endl;
    }
}

// Function to search cars by brand
void searchByBrand(car* arr, int count) {
    if (count == 0) {
        cout << "No cars to search.\n";
        return;
    }

    string brandSearch;
    cout << "Enter brand to search: ";
    cin >> brandSearch;

    bool found = false;
    for (int i = 0; i < count; i++) {
        if (arr[i].getBrand() == brandSearch) {
            cout << "\n--- Found car ---\n";
            cout << "  Brand: " << arr[i].getBrand() << endl;
            cout << "  Model: " << arr[i].getModel() << endl;
            cout << "  Plate Number: " << arr[i].getPlateNumber() << endl;
            cout << "  Owner: " << arr[i].getOwner() << endl;
```

```
        cout << "No cars to search.\n";
        return;
    }

    string brandSearch;
    cout << "Enter brand to search: ";
    cin >> brandSearch;

    bool found = false;
    for (int i = 0; i < count; i++) {
        if (arr[i].getBrand() == brandSearch) {
            cout << "\n--- Found car ---\n";
            cout << "  Brand: " << arr[i].getBrand() << endl;
            cout << "  Model: " << arr[i].getModel() << endl;
            cout << "  Plate Number: " << arr[i].getPlateNumber() << endl;
            cout << "  Owner: " << arr[i].getOwner() << endl;
            found = true;
        }
    }

    if (!found) {
        cout << "No cars found with brand \"" << brandSearch << "\".\n";
    }
}
```

Ziua 7 practică

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


//CREATE THE CLASS
class car{
    private:
        string brand;
        string model;
    public:

        car(){
            brand="EMPTY";
            model="EMPTY";

        }
        void setBrand(string B){
            brand=B;
        }
        void setModel(string M){
            model=M;
        }
        string getBrand(){
            return brand;
        }
        string getModel(){
            return model;
        }
};
//function read data
//the function accepts a pointer to a car object (the array) and the size of the array the
//function return back to a pointer to a car object which is the array
```

```cpp
//the function accepts a pointer to a car object (the array) and the size of the array the
//function return back to a pointer to a car object which is the array

car *readData(car *,int ,int &);


int main(){

    int ARRSIZE=4;
    int nextAvailPos=0;
    car *carsArray;
    carsArray=new car[ARRSIZE];
    int choice=-1;
    do{
        cout<<"Please Choose form the menu below: "<<endl;
        cout<<"1.Enter data"<<endl;
        cout<<"2.Display data"<<endl;
        cout<<"3.Search by brand"<<endl;
        cout<<"0.exit"<<endl;
        cin>>choice;

        switch(choice){
            case 1:
          carsArray= readData(carsArray,ARRSIZE,nextAvailPos);
            break;
            case 2:
            break;
            case 3:
            break;
            case 0:
            break;

        }

    }while (choice!=0);



    return 0;
}
```

```
}

car *readData(car *arr,int arrSize,int &nextPos){
    int numcars=0;
    string tstring;
    car *tmp;

    cout<<"How many cars are you entering: ";
    cin>>numcars;//how many cars you need to input
     for(int i=0;i<numcars;i++){//repeat the reading process  *numCars *times
    if(nextPos>=arrSize){//check if the array is ful
        newSize=arrSize+numCars;//if the array is full we calculate the new size
        tmp=new car[newSize];//alocate temporary space for the new bigger array

        for(int j=0;j<newSize;j++){//copy everything from the small array form the new bigger one
            tmp[j]=arr[j];
        }
        arrSize=newSize;//save the newsize into arrsize update the arrsize
        delete []arr;// release the memory from the old array back to the os

        arr=tmp;//set the pointer of the array to point to the new allocated space
    }
    cout<<"Enter. brand for car ["<<i<<"]";
    cin>>tstring;
    arr[nextPos].setBrand(tstring);
    cout<<"Enter Model for car ["<<i<<"]";
    cin>>tstring;
    arr[nextPos].setModel(tString);
    nextPos++;


}
return arr;


}
```

Ziua 8 temă

```cpp
int main() {
    int ARRAYSIZE = 4;
    int nextAvailPos = 0;
    car* carsArray = new car[ARRAYSIZE];
    int choice = -1;
    int searchIndex = -1;

    do {
        cout << "Please Choose from the menu below: " << endl;
        cout << "1. Enter Data" << endl;
        cout << "2. Display Data" << endl;
        cout << "3. Search By Brand" << endl;
        cout << "4. Edit Car Data By Brand" << endl;
        cout << "0. Exit" << endl;
        cin >> choice;

        switch (choice) {
        case 1:
            carsArray = readData(carsArray, ARRAYSIZE, nextAvailPos);
            break;
        case 2:
            displayData(carsArray, nextAvailPos);
            break;
        case 3:
            searchIndex = searchByBrand(carsArray, nextAvailPos);
            break;
        case 4:
            searchIndex = searchByBrand(carsArray, nextAvailPos);
            if (searchIndex != -1) {
                editData(carsArray[searchIndex]);
            }
            else {
                cout << "Car not found!" << endl;
            }
            break;
        case 0:
            break;
        }
    } while (choice != 0);

    delete[] carsArray; // curățăm memoria
    return 0;
```

```
    delete[] carsArray; // curățăm memoria
    return 0;
}

int searchByBrand(car* arr, int arrSize) {
    string tstr;
    int index = -1;

    cout << "Enter the brand you are looking for: ";
    cin >> tstr;

    for (int i = 0; i < arrSize; i++) {
        if (arr[i].getBrand() == tstr) {
            cout << "Index found at position: " << i << endl;
            index = i;
            break;
        }
    }

    return index;
}

void displayData(car* arr, int arrSize) {
    for (int i = 0; i < arrSize; i++) {
        cout << "----- Car " << i + 1 << " -----" << endl;
        cout << "Brand: " << arr[i].getBrand() << endl;
        cout << "Model: " << arr[i].getModel() << endl;
    }
}

car* readData(car* arr, int& arrSize, int& nextPos) {
    int numcars = 0;
    int newSize = 0;
    string tstring;
    car* tmp;

    cout << "How many cars are you entering: ";
    cin >> numcars;

    for (int i = 0; i < numcars; i++) {
        if (nextPos >= arrSize) {
```

```
car* readData(car* arr, int& arrSize, int& nextPos) {
    int numcars = 0;
    int newSize = 0;
    string tstring;
    car* tmp;

    cout << "How many cars are you entering: ";
    cin >> numcars;

    for (int i = 0; i < numcars; i++) {
        if (nextPos >= arrSize) {
            newSize = arrSize + numcars;
            tmp = new car[newSize];
            for (int j = 0; j < arrSize; j++) {
                tmp[j] = arr[j];
            }
            arrSize = newSize;
            delete[] arr;
            arr = tmp;
        }

        cout << "Enter BRAND for car[" << nextPos << "]: ";
        cin >> tstring;
        arr[nextPos].setBrand(tstring);

        cout << "Enter MODEL for car[" << nextPos << "]: ";
        cin >> tstring;
        arr[nextPos].setModel(tstring);

        nextPos++;
    }

    return arr;
}
            nextPos++;
        }

        return arr;
    }

    // FUNCȚIA NOUĂ: EDITAREA UNUI OBIECT CAR
    void editData(car& c) {
        string tstring;

        cout << "Current BRAND: " << c.getBrand() << endl;
        cout << "Current MODEL: " << c.getModel() << endl;

        cout << "Enter NEW BRAND: ";
        cin >> tstring;
        c.setBrand(tstring);

        cout << "Enter NEW MODEL: ";
        cin >> tstring;
        c.setModel(tstring);

        cout << "Data updated successfully!" << endl;
    }
```

Ziua 8 practică

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


//CREATE THE CLASS
class car{
    private:
        string brand;
        string model;
    public:

        car(){
            brand="EMPTY";
            model="EMPTY";

        }
        void setBrand(string B){
            brand=B;
        }
        void setModel(string M){
            model=M;
        }
        string getBrand(){
            return brand;
        }
        string getModel(){
            return model;
        }
};

car *readData(car *, int & ,int &);
void displayData(car*, int);
int searchByBrand(car*,int);

int main(){

    int ARRAYSIZE=4;
    int nextAvailPos=0;
    car *carsArray;
    carsArray=new car[ARRAYSIZE];
    int choice=-1;
    int searchIndex=-1;
```

```
    carsArray=new car[ARRAYSIZE];
    int choice=-1;
    int searchIndex=-1;

    do{
        cout<<"Please Choose from the menu below: "<<endl;
        cout<<"1. Enter Data"<<endl;
        cout<<"2. Display Data"<<endl;
        cout << "3. Search By Brand" << endl;
        cout<<"0. Exit"<<endl;
        cin>>choice;

        switch(choice){
            case 1:
                carsArray=readData(carsArray,ARRAYSIZE, nextAvailPos);
                break;
            case 2:
                //because we are growing the array dynamically,
                //the variable ARRAYSIZE is the TOTAL SIZE OF THE ARRAY,
                //which may contain empty positions.
                //The variable nextAvailPos tells us the last index which
                //contains data.
                displayData(carsArray,nextAvailPos);
                break;
            case 3:
                searchIndex=searchByBrand(carsArray,nextAvailPos);
                break;
            case 4: //edit data inside the array
                    //call the search function first and get
                    //the index that you want to change.
                    //once you have the index send it to the edit Data function
                    //the edit data function will first, show you what is
                    //saved at the current location and then give you the option
                    //to input new data.
                break;
            case 0:
                break;


        }
```

```
    }while (choice!=0);

    return 0;
}

int searchByBrand(car* arr,int arrSize){
    string tstr;//temporary string to read data from keyboard
    cout<<"Enter the brand you are looking for: ";
    cin>>tstr;
    for (int i=0;i<arrSize;i++){
        if (arr[i].getBrand()== tstr){
            cout<<"Index found at position: "<<i<<endl;
            index=i;
        }

    }

    return index;

}
//void displayData()
//the function receives as arguments
//a car object array and the size of the array.
//It displays all data that are saved on the car array
void displayData(car *arr, int arrSize){

    for (int i=0;i<arrSize;i++){
        cout<<"----- "<<"Car "<<i+1<<" -----"<<endl;
        cout<<arr[i].getBrand()<<endl;
        cout<<arr[i].getModel()<<endl;
    }

}
//function car *readData(car *, int)
//the function populates the array with information
//from the user
//the function accepts a pointer to a car object(the array)\
//and the size of the array
//the function return back a pointer to aa car object which is the array
//and is now populated with data.
car *readData(car *arr, int &arrSize,int &nextPos){
```

```
    }

}
//function car *readData(car *, int)
//the function populates the array with information
//from the user
//the function accepts a pointer to a car object(the array)\
//and the size of the array
//the function return back a pointer to aa car object which is the array
//and is now populated with data.
car *readData(car *arr, int &arrSize,int &nextPos){
    int numcars=0; //how many cars the user is going to input
    int newSize=0; //the new size of the array which is calculated by adding arraySize+numCars
    string tstring; //temporary string to read keyboard input
    car *tmp; //temporary car object to copy everything from the car array and grow the array;

    cout<<"How many cars are you entering: ";
    cin>>numcars; //read from the keyboard how many cars the user needs to input
    for (int i=0;i<numcars;i++){//repeat the reading process 'numCars' times

        if( nextPos>=arrSize){ //first check if the array is full
          newSize=arrSize+numcars;//if you got here the array is full calculate the new size
          tmp=new car[newSize]; //allocate temporary space for the new bigger array
          for (int j=0;j<arrSize;j++){//copy everything from the small array to the new bigger one
              tmp[j]=arr[j];
          }
           arrSize=newSize; //save the newsize into arrSize (update the arrSize).

           delete []arr; //release the memory from the old array back to the OS
           arr=tmp; //set the pointer of the array to point to the new allocated space

        }
        cout<<"Enter BRAND for car["<<nextPos<<"]";
        cin>>tstring;
        arr[nextPos].setBrand(tstring);
        cout<<"Enter MODEL for car["<<nextPos<<"]";
        cin>>tstring;
        arr[nextPos].setModel(tstring);
        nextPos++;

    }
```

```
          newSize=arrSize+numcars;//if you got here the array is full calculate the new size
          tmp=new car[newSize]; //allocate temporary space for the new bigger array
          for (int j=0;j<arrSize;j++){//copy everything from the small array to the new bigger one
              tmp[j]=arr[j];
          }
           arrSize=newSize; //save the newsize into arrSize (update the arrSize).

           delete []arr; //release the memory from the old array back to the OS
           arr=tmp; //set the pointer of the array to point to the new allocated space

        }
        cout<<"Enter BRAND for car["<<nextPos<<"]";
        cin>>tstring;
        arr[nextPos].setBrand(tstring);
        cout<<"Enter MODEL for car["<<nextPos<<"]";
        cin>>tstring;
        arr[nextPos].setModel(tstring);
        nextPos++;

    }
    return arr;
}
//void displayData(car *, int)
//function displayData(char*,int) will print all data
//that we saved in our object array at any given time.
```

Ziua 9 temă

```ino
/*
  DigitalReadSerial

  Reads a digital input on pin 2, prints the result to the Serial Monitor

  This example code is in the public domain.

  https://www.arduino.cc/en/Tutorial/BuiltInExamples/DigitalReadSerial
*/

// digital pin 2 has a pushbutton attached to it. Give it a name:
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);  // delay in between reads for stability
}
```

Ziua 9 practică

```ino
/*
  AnalogReadSerial

  Reads an analog input on pin 0, prints the result to the Serial Monitor.
  Graphical representation is available using Serial Plotter (Tools > Serial Plotter menu).
  Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.

  This example code is in the public domain.

  https://www.arduino.cc/en/Tutorial/BuiltInExamples/AnalogReadSerial
*/

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);  // delay in between reads for stability
}
```

Pregătire mobilitate - *Egalitate de*

șanse,toleranță și diversitate socială

EGALITATE DE SANSE, TOLERANTA SI DIVERSITATE SOCIALA



CUPRINS

- INTRODUCERE SI IMPORTANTA
- EDUCATIA PENTRU EGALITATE
- INCLUZIUNEA SOCIALA
- STEREOTIPURI SI PREJUDECATI
- PREVENIREA DISCRIMINARII
- ORGANIZATII SI IMPORTANTA LOR



INTRODUCERE

CE ESTE TOLERANȚA?

Toleranța este valoarea care stă la baza drepturilor universale ale omului și a libertăților fundamentale. Este vorba despre acceptarea și respectarea diferențelor dintre oameni, culturi, religii și opinii, fără discriminare sau prejudecăți.

CE ESTE EGALITATEA DE ȘANSE?

În general, termenul „egalitate de șanse" este utilizat pentru a se referi la condiția de paritate și egalitate introdusă pentru a garanta că toate persoanele sunt tratate în mod egal și pentru a preveni formele de discriminare pe baza anumitor aspecte (vârstă, etnie, handicap, orientare religioasă și politică etc.).

CE ESTE DIVERSITATEA SOCIALĂ?

Diversitatea socială este un proces prin care părți ale unei populații dobândesc treptat o identitate distinctă în ceea ce privește funcția, activitatea, structura, cultura, autoritatea, puterea sau alte caracteristici sociale semnificative și relevante.





AM UN VIS CĂ ÎNTR-O BUNĂ ZI CEI PATRU COPII AI MEI VOR TRĂI ÎNTR-O NAȚIUNE ÎN CARE NU VOR MAI FI JUDECAȚI DUPĂ CULOAREA PIELII LOR, CI DUPĂ CARACTERUL LOR. MARTIN LUTHER KING

INCLUZIUNEA SOCIALĂ - FUNDAMENTUL EGALITĂȚII ÎN SOCIETATE

CE ESTE INCLUZIUNEA?

Incluziunea socială este un proces multidimensional prin care indivizii și grupurile marginalizate sunt integrate activ în viața economică, socială, culturală și politică a unei societăți. Conform cercetărilor din domeniul sociologiei și studiilor privind drepturile omului, incluziunea socială nu înseamnă doar absența discriminării, ci și crearea unor structuri și politici care asigură participarea echitabilă a tuturor cetățenilor.



FACTORII INCLUZIUNII SOCIALE

Incluziunea socială depinde de mai mulți factori esențiali, printre care:

- **Accesul la educație și formare profesională** – Oferă oportunități egale pentru dezvoltare și mobilitate socială.
- **Participarea economică** – Asigură integrarea pe piața muncii și reducerea decalajelor salariale.
- **Accesul la servicii publice și sănătate** – Contribuie la reducerea inegalităților și îmbunătățirea calității vieții.
- **Reprezentarea și participarea politică** – Garantează că toate grupurile au o voce în procesul decizional.

BENEFICIILE INCLUZIUNII SOCIALE

- **Coeziune socială** – Diminuează conflictele sociale și promovează solidaritatea.
- **Dezvoltare economică sustenabilă** – Studiile arată că societățile incluzive sunt mai prospere.
- **Creșterea capitalului social** – Relațiile de încredere și colaborare între diferite grupuri duc la o comunitate mai stabilă.

DIVERSITATEA ESTE UN FAPT. INCLUZIUNEA ESTE O ALEGERE PE CARE O FACEM ZILNIC. EGALITATEA ESTE SCOPUL.

JOHNETTA BETSCH COLE

PREVENIREA DISCRIMINARII

Nimeni nu se naşte urând o altă persoană din cauza culorii pielii, a trecutului sau sau a religiei sale. Oamenii învaţă să urască, atunci pot fi învăţaţi să iubească.-Nelson Mandela

Prevenirea discriminării este un aspect esențial al unei societăți juste și echitabile. Aceasta presupune un set de acțiuni și strategii care vizează eliminarea inegalităților și asigurarea unui tratament egal pentru toți oamenii.



CÂTEVA MĂSURI PRIN CARE SE POATE PREVENI DISCRIMINAREA:

- **Educarea și sensibilizarea publicului**: Promovarea unor atitudini corecte și respectuoase față de diversitate prin campanii educaționale care explică conceptele de discriminare și diversitate.
- **Legea și reglementările anti-discriminare:** Crearea și implementarea unor legi clare care interzic discriminarea în toate formele sale, precum și sancționarea celor care încalcă aceste legi.
- **Promovarea incluziunii în locuri de muncă și educație**: Companiile și instituțiile educaționale pot implementa politici care garantează oportunități egale pentru toți

EDUCATIA PENTRU EGALITATE

Educația pentru egalitate joacă un rol esențial în promovarea diversității și incluziunii, învățând elevii să recunoască și să combată stereotipurile și prejudecățile care există în societate. Prin educație, tinerii învață să aprecieze diferențele de rasă, religie, gen, orientare sexuală, dizabilitate sau orice altă caracteristică personală, dezvoltându-și astfel o înțelegere corectă a principiilor de egalitate și echitate.

„Educația este cheia pentru închiderea diferențelor de șanse și pentru crearea unei lumi mai egale.” – Malala Yousafzai

Acest proces educativ este crucial pentru crearea unui mediu social, economic și cultural în care toți indivizii, indiferent de background-ul lor, au șanse egale de a se dezvolta și de a-și atinge potențialul. Prin învățarea acestor valori, educația contribuie la formarea unor relații mai armonioase între grupuri sociale diferite, reducând discriminarea și creând o societate mai echitabilă.



ACTIVITATI CARE INCURAJEAZA EDUCATIA PENTRU EGALITATE

Un exemplu practic al educației pentru egalitate ar putea fi implementarea unui program de sensibilizare asupra diversității în școli, cum ar fi "Lecții de Diversitate". Acesta poate include jocuri de rol, ateliere de discuție și întâlniri cu persoane din grupuri marginalizate, pentru a dezvolta empatia și respectul față de diferențele culturale, de gen sau religioase. În plus, bursele Erasmus pentru mobilitate studențească oferă o oportunitate reală de a învăța despre diversitate într-un context internațional, încurajând schimbul cultural și înțelegerea între tineri din diferite colțuri ale lumii. Aceste activități contribuie la formarea unei mentalități incluzive.

STEREOTIPURI SI PREJUDECATI

În societatea modernă, diversitatea este o realitate de necontestat. Cu toate acestea, modul în care ne raportăm la cei diferiți de noi este adesea influențat de stereotipuri și prejudecăți. Aceste concepte, deși par inofensive la prima vedere, pot genera discriminare, excludere socială și conflicte.

CE ESTE UN STEREOTIP?

Un stereotip este o idee fixă despre un grup de oameni, fără să îi cunoaștem cu adevărat. Acestea se bazează pe trăsături cum ar fi genul, etnia, religia sau naționalitatea.

Spre exemplu:

- Fetele sunt mai sensibile și plâng ușor.
- Oamenii cu tatuaje sunt delincvenți sau rebeli.
- Fetele se pricep mai bine la gătit și la curățenie.
- Toți asiaticii sunt foarte inteligenți și buni la matematică.
- Oamenii bogați sunt aroganți și egoiști.

CE ESTE O PREJUDECATĂ?

O prejudecată este o opinie negativă formată înainte să cunoaștem o persoană, bazată doar pe aparențe sau apartenență. Acestea pot fi alimentate de familie, mass-media sau experiențe limitate.

Spre exemplu:

- Nu am încredere într-o femeie manager, sigur nu se va impune
- Dacă are accent străin, sigur nu știe limba bine.
- Dacă e sărac, sigur nu are bun-simț.
- E prea bătrân ca să mai învețe ceva nou.
- Are tatuaje, sigur e delincvent.

STEREOTIPURI SI PREJUDECATI

CUM NE AFECTEAZĂ STEREOTIPURILE ȘI PREJUDECĂȚILE?

Efect	Exemplu concret
Excludere socială	Elevii care vin din altă țară sunt ignorați sau marginalizați.
Auto-limitare	Fetele evită specializări tehnice pentru că „nu sunt pentru ele”.
Discriminare	Candidații cu nume străine sunt respinși la angajare.
Creșterea tensiunilor sociale	Prejudecățile etnice pot duce la conflicte sau ură colectivă.

CUM COMBATEM STEREOTIPURILE ȘI PREJUDECĂȚILE?

Soluție	Exemplu practic
Educație pentru diversitate	Proiecte în școală despre culturi și comunități diverse.
Contact direct	Activități comune cu grupuri diferite (schimburi de experiență).
Promovarea modelelor pozitive	Personalități din grupuri diverse care au reușit.
Gândire critică	Analizăm informațiile și ne punem întrebări înainte să judecăm.
Implicare activă	Să luăm poziție când vedem discriminare în jurul nostru.

ORGANIZATII CARE PROTEJEAZA MINORITATI

FRA (Agency for Fundamental Rights)

-este o organizație unică si independentă care se ocupă cu promovarea si protejarea drepturilor omului in UE

ONU(Organizația Națiunilor Unite)

-are printre obiective menținerea păcii mondiale și dezvoltarea unor relații de prietenie între țări

OSCE(Organization for Security and Co-operation in Europe)

-această organizație ajută FRA în a combata și dezvălui cazuri de violență din motive discriminatorii

Printre drepturile protejate se regăsesc:
-dreptul de a nu fi discriminat pe baza vârstei, dizabilităților sau religie
-dreptul de a avea datele personale protejate
-dreptul de a avea acces la justiție





IMPORTANTA ORGANIZATIILOR



Protejarea și egalitatea minorităților

-indiferent de persoană ea trebuie să aibă drepturi și responsabilități egale, indiferent de loc

Asigurarea incluziunii minorităților

-doar pentru că ești diferit nu înseamnă că nu meriți să fi inclus

Menținerea identității sociale ale individului

-orice persoană are dreptul să își exprime liber orientările și mentalitățile

Minimalizarea discriminării între persoane

-o lume în care fiecare persoană este acceptată așa cum este ea este idealul spre care ar trebui să îl atingem



IMPORTANTA TEMEI IN CADRUL PROIECTULUI

Creșterea conștientizării asupra drepturilor fundamentale

Această conștientizare stimulează respectarea drepturilor fiecărei persoane, indiferent de context.

Promovarea unei mentalități incluzive

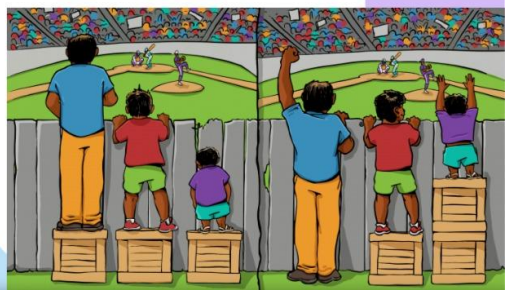
Cunoașterea acestei teme ne ajută să identificam situațiile de nedreptate și să susținem incluziunea și egalitatea.

Dezvoltarea empatiei și toleranței

Aceste calități sunt esențiale pentru relații armonioase și colaborare internațională.

Înțelegerea și adaptarea într-un mediu cultural diferit

Ne ajută să fim mai deschiși și mai respectuoși față de diversitatea culturală și socială, facilitând integrarea într-un mediu nou.



BIBLIOGRAFIE

- *https://fra.europa.eu/ro*
- *https://epochtimes-romania.com/news/toleranta-o-valoare-necesara-convietuirii-pasnice---320667*
- *https://www.britishcouncil.ro/despre-noi/sanse-egale-diversitate#:~:text=Egalitatea%20de%20%C8%99anse%20%C3%AEnseamn%C4%83%20a,de%20lucru%20%C8%99i%20%C3%AEn%20societate.*
- *https://gep.unibuc.ro/wp-content/uploads/2023/01/Modul-Diversitate-egalitate-si-incluziune-sociala%CC%86.pdf*
- *https://eige.europa.eu/gender-based-violence*
- *https://www.europarl.europa.eu/about-parliament/ro/democracy-and-human-rights/fundamental-rights-in-the-eu/promoting-equal-opportunities*

MULTUMIM PENTRU ATENTIE!

