

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

Practice tutor: Lazaros Lazarou

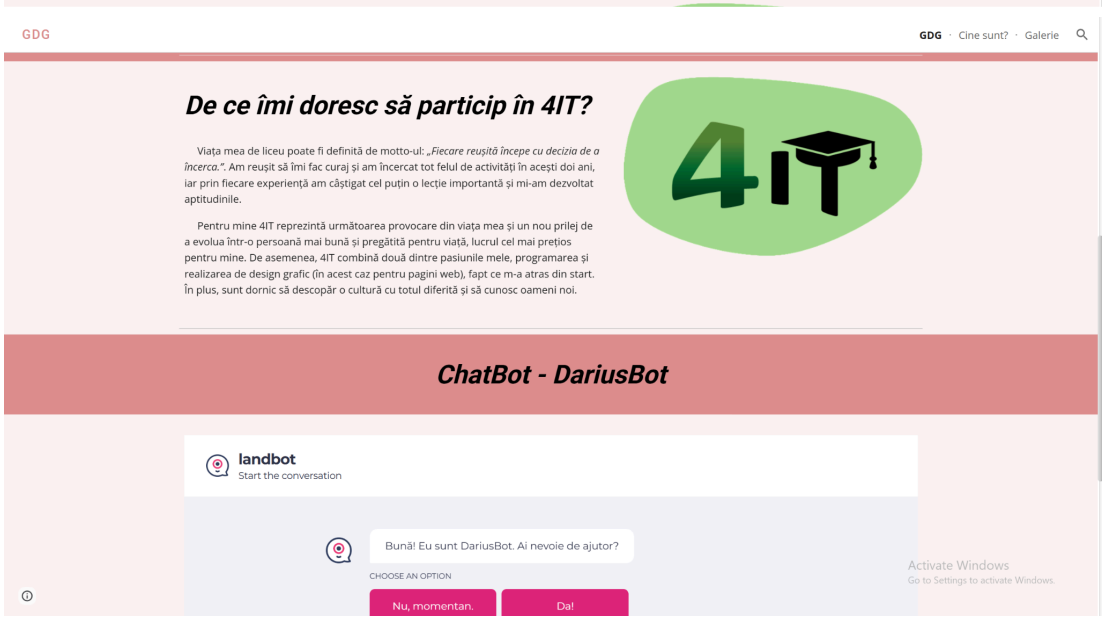
Participant: Grigoraș Darius-George

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.

GDG - Site of Selection







landbot
Start the conversation

Bună! Eu sunt DariusBot. Ai nevoie de ajutor?

CHOOSE AN OPTION

Nu, momentan. | Da!

Made with Landbot

Email

dariusgeorge2007dg@gmail.com



Colegiul Național „Roman-Vodă” localitatea Roman, județul Neamț, România.

Activate Windows
Go to Settings to activate Windows.

„Fiecare reușită începe cu decizia de a încerca.”

Cine sunt?

Bună! Eu mă numesc Grigoraș Darius-George, am 17 ani și sunt elev al Colegiului Național „Roman-Vodă”. Mă aflu în clasa a 11-a A, profil Matematică-Informatică, Intensiv Informatică, iar programarea este una dintre cele mai mari pasiuni ale mele.

Povestea mea

Povestea mea cu calculatoarele a început de la vârsta de 4 ani, când mergeam în weekend-uri la unchiul meu. Acesta obișnuia să scrie multe documente, dar din când în când se juca jocuri video, iar eu eram „mâna lui dreaptă”. Încă din acea perioadă dobândisem o pasiune pentru calculatoare, iar aceasta s-a dezvoltat și mai mult în timp, datorită maturizării mele și a dorinței din ce în ce mai mare de a rezolva probleme întâmpinate în viața mea

GDG

GDG · **Cine sunt?** · Galerie

iar aceasta s-a dezvoltat și mai mult în timp, datorită maturizării mele și a dorinței din ce în ce mai mare de a rezolva probleme întâmpinate în viața mea și în viețile celorlalți. Astfel, la sfârșitul clasei a 8-a, am luat decizia de a-mi dezvolta această pasiune într-o posibilă ocupație și mod de a-mi câștiga existența.

Primul an de liceu a avut un impact major asupra caracterului meu. Toată copilăria mea am fost un copil introvertit, timid, izolat, cu un grup restrâns de prieteni și cu stimă de sine inexistentă. Totuși, în doar 3 luni de liceu, am reușit să devin extrovertit, să îmi fac mulți prieteni noi și să am încredere în aparențele și în capacitățile mele. Cum s-a întâmplat acest lucru?

Încă din primele zile de liceu am fost „bombardat” de informații noi și de oportunități. Fiecare asociație recruta membri, iar eu simțeam nevoia să cunosc persoane noi. Astfel, am urmat pentru prima dată motto-ul „Fiecare reușită începe cu decizia de a încerca.” și am luat decizia de a mă înscrie într-o asociație de voluntariat, People To People International - Roman Chapter (PTPI), alături de un vechi amic. Această decizie mi-a schimbat radical viața.

Activități extrașcolare din întreaga perioadă a liceului:



Voluntar și Coordonator PR



Vicepreședinte



Membru și Fotograf

Activate Windows
Go to Settings to activate Windows.

Voltar și Coordonator PR
în asociația „PTPI - Roman Chapter”

Vicepreședinte
al Consiliului Școlar al Elevilor

Membru și Fotograf
în proiectul „Eco-Școala”

Vezi mai multe imagini

PTPI

Primul meu proiect de voluntariat la care am participat s-a numit „Caravana Bucurie în Mișcare”. Nu era desfășurat de PTPI, însă organizatorii au făcut apel la voluntarii din asociația noastră pentru a-i ajuta. A fost o experiență deosebit de importantă în dezvoltarea mea, deoarece am cunoscut câteva persoane care au avut, mai apoi, un rol special în viața mea și pentru că a fost pentru prima dată când am simțit un sentiment de bucurie unic fără să fac un lucru benefic pentru propria persoană, ci stăteam de ore în șir în picioare și puneam un zâmbet pe chipul unor copii sărmani. De atunci, dragostea mea pentru voluntariat a început cu adevărat.

Această iubire rămâne intactă și în ziua de azi, însă în acea perioadă s-a concretizat cu adevărat, prin participarea la aproape toate proiectele desfășurate. Mai apoi, am vrut să ofer și eu înapoi voluntariatului și acestei asociații, astfel încât am candidat pentru funcția de Coordonator PR și am fost votat de ceilalți voluntari. Din aventura din al doilea an al meu, ocupând noua funcție, am reușit să-mi satisfac dorința de a oferi înapoi comunității și mi-am dezvoltat creativitatea și abilitatea de a concepe grafici, texte, fotografii și de a edita video. Însă, cel mai important, am învățat cum să preiau inițiativa și să fiu un lider pentru departamentele cu care am colaborat, dar și pentru ceilalți patru membri Board, care ocupau, de asemenea, funcții cu rol decizional în asociație.

Consiliul Școlar al Elevilor

După primul meu proiect de voluntariat, am devenit din ce în ce mai activ la activități și m-am decis să încerc și alte experiențe noi, deoarece ieșitul din zona de confort funcționase foarte bine până în acel moment. Astfel, am candidat pentru postul de Vicepreședinte al Consiliului Școlar al Elevilor și am fost votat pentru a ocupa funcția. Această experiență a fost una benefică pentru mine, din care am deprins calitatea de a comunica eficient și gestionându-mi emoțiile, am învățat cum se organizează un proiect și ce înseamnă cu adevărat să fii responsabil.

Activate Windows
Go to Settings to activate Windows.

GDG

GDG · **Cine sunt?** · Galerie

Eco-Școala CNRV

Eco-Școala reprezintă pentru mine organizația care mi-a deschis orizontul spre o altă pasiune importantă din viața mea, design-ul, deși are ca scop ecologizarea și transmiterea unui mesaj care să producă o schimbare majoră în comportamentul irresponsabil al unor oameni față de mediu. Poate părea amuzant, însă era nevoie de o persoană care să capteze momentele din timpul proiectelor de ecologizare și care să ia mesajele din spatele acțiunilor voluntarilor și să îl afișeze în așa fel încât să fie captivant pentru cel ce îl privește și de asemenea să îi stârnească un gând de reflecție asupra propriului său comportament. Fiind deja membru implicat în activități, mi-a fost propus acest post de fotograf și am acceptat. Pe lângă fotografii, am realizat și grafici sau texte pentru promovarea valorilor grupului și a valorilor cu care mă identificam, iar această oportunitate mi-a dezvoltat creativitatea și probabil are un rol în candidatura mea pe postul de Coordonator PR în asociația PTPI, la sfârșitul clasei a 9-a.

Aspirațiile mele

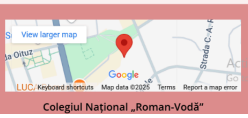
Pentru viitorul apropiat, îmi doresc să dau tot ce pot pentru participarea în mobilitatea 4IT, iar în momentul în care redactez mesajul acesta simt că mi-am îndeplinit acest scop, deși mai am câteva etape de parcurs. Deja am învățat lucruri noi din pregătirea pentru participarea într-un astfel de proiect și sunt convins că voi mai acumula și alte noțiuni în următoarele zile sau poate luni.

Pentru viitorul pe termen mediu, aspir să studiez la universitatea dorită, să mă îmbunătățesc din punct de vedere fizic, mental și emoțional, să râd de mine din anii de liceu, dar să mă uit la acea versiune a mea cu drag și cu multă recunoștință.

Pe termen lung, am ca scop să fiu așezat la casa mea, să am o familie, să am un program de muncă flexibil, să fiu mai spiritual, să continui să lucrez și să trăiesc cu pasiune în viața de zi cu zi, iar nu în ultimul rând, să îmi mențin sănătatea.

Email

dariusgeorge2007dg@gmail.com



Colegiul Național „Roman-Vodă”

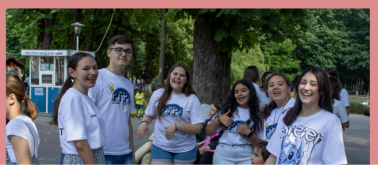
GDG

GDG · Cine sunt? · **Galerie**

„Fiecare reușită începe cu decizia de a încerca.”

Galerie - Fotografii

Fotografii PTPI



Fotografii Eco-Școala







Fotografii Consiliul Școlar al Elevilor



Fotografii Raliul Romanului 2023



Activate Windows
Go to Settings to activate Windows.

GDG

GDG · Cine sunt? · **Galerie**

Galerie - Grafică

Grafică PTPI realizată de mine



Grafică Eco-Școala realizată de mine



Activate Windows
Go to Settings to activate Windows.

GDG

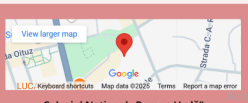
GDG · Cine sunt? · **Galerie**





Email

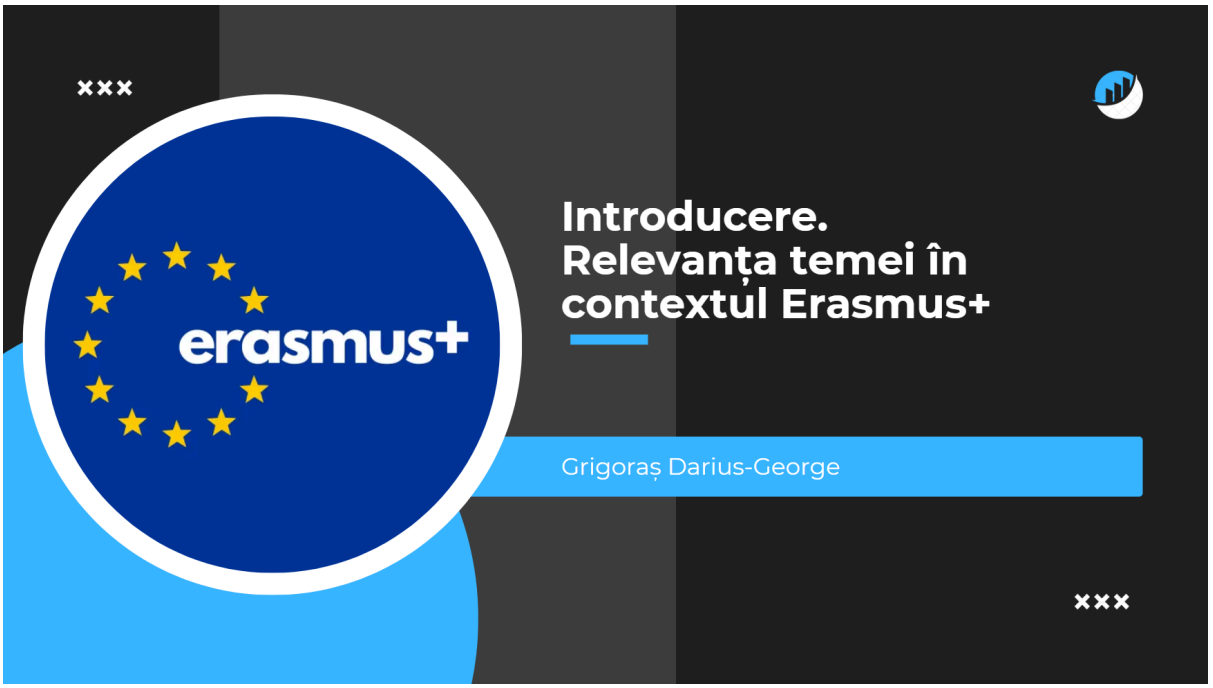
dariusgeorge2007dg@gmail.com



Colegiul Național „Roman-Vodă”
localitatea Roman, județul Neamț, România.

Project - “Cum Să-ți Găsești Un Loc De Muncă”





Introducere

- Procesul de căutare a unui loc de muncă poate părea copleșitor, dar este esențial să înțelegem pașii corecți pentru a avea succes.
- **De ce este important să înțelegem acest proces?**
- **Ce vom învăța?**



Relevanța temei în contextul Erasmus+

Ce înseamnă Erasmus+ pentru cariera ta?

- ***Experiență internațională:*** Oportunitatea de a explora diverse culturi și piețe de muncă în Europa.
- ***Experiență practică:*** Participarea la proiecte, stagii și activități de voluntariat care îți oferă abilități utile pentru cariera ta viitoare.
- ***Rețea de contacte internaționale:*** Oportunitatea de a te conecta cu angajatori și colegi din întreaga Europă.

Beneficii:

- **Acces la un job internațional.**
- **Dobândirea unor abilități transversale** (ex. comunicare interculturală, adaptabilitate).

C++ Problems

14.04.2025 - Lesson 1 Homework:

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

int main()
{
   int ARRAYSIZE = 0; // Initial array size to be provided by the user
   char* a;          // Pointer to dynamically allocated character array
   char* temp;       // Temporary pointer used for resizing the array
   int n = 0, tmpN = 0; // 'n' is the number of characters entered, 'tmpN' holds additional size if
needed

   // Ask user for initial number of characters to input
   cout << "How many numbers would you like to enter?: ";
   cin >> ARRAYSIZE;

   // Allocate dynamic array with given size
   a = new char[ARRAYSIZE];

   // Loop to input characters (ASCII values between 65 and 90)
   do {
      int x; // Temporary variable to store user input as integer
      cout << "Enter number " << n << " -> ";
      cin >> x;

      // Only accept values that map to uppercase ASCII letters (A-Z)
      if (x >= 65 && x <= 90) {
         a[n] = (char)(x); // Store the ASCII character in array
         cout << "Number saved!" << endl;
         n++; // Increment the counter of stored characters

         // Check if array is full
         if (n >= ARRAYSIZE) {
            // Ask user if they want to add more characters
            cout << "Out of Space!!!! would you like to enter more? Please enter how many (0 to
Exit): ";
            cin >> tmpN;

            // If user requests more space
            if (tmpN > 0) {
               ARRAYSIZE += tmpN; // Increase total array size
```

```
            temp = new char[ARRAYSIZE]; // Allocate larger array

            // Copy old data into new array
            for (int i = 0; i < n; i++) {
               temp[i] = a[i];
            }

            delete[] a; // Free memory of old array
            a = temp;   // Point 'a' to the new larger array
         }
      }
   } else {
      // Reject invalid input
      cout << "Number is not in 65-90 range. ";
   }

}while (n < ARRAYSIZE); // Continue until desired number of valid inputs

// Print stored characters
cout << "Now printing the numbers you inputted" << endl;
for (int i = 0; i < ARRAYSIZE; i++) {
   cout << a[i] << endl;
}

// Clean up dynamic memory
delete[] a;

return 0;
}
```

15.04.2025 - Lesson 2 Homework:

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

// Function to search for a string in an array of strings
int search(string *v, int arraysize, string variable)
{
   int i;
   for (i = 0; i < arraysize; i++)
   {
      if (v[i] == variable)
         return i; // Return the index if found
```

```
    }
    return 0;
}

int main()
{
    int arraysize;
    string *telephone;
    string *name;
    int i;
    string variable;

    cout << "How many students would you like to enter?: ";
    cin >> arraysize;

    // Dynamic memory allocation for name and telephone arrays
    telephone = new string[arraysize];
    name = new string[arraysize];

    // Input names and telephone numbers
    for (i = 0; i < arraysize; i++)
    {
        cout << "Enter name ->" << i << "-> ";
        cin >> name[i];
        cout << endl << "Enter number ->" << i << "-> ";
        cin >> telephone[i];
        cout << endl;
    }

    int OK = 1;
    while (OK != 0)
    {
        // Menu
        cout << "What would you like to search for?" << endl
             << "Option 1: Search for a name" << endl
             << "Option 2: Search for a phone number" << endl
             << "Option 3: Exit";
        int x;
        cin >> x;

        switch(x)
        {
            case 1:
            {
                string variable;
                cout << "Enter the telephone number" << endl;
                cin >> variable;
```

```
            int y = search(name, arraysize, variable);
            if (y != 0)
               cout << "The name you are searching for is: " << name[i] << endl;
            else
               cout << "Not found";
            break;
         }
         case 2:
         {
            string variable;
            cout << "Enter the name:" << endl;
            cin >> variable;
            int y = search(telephone, arraysize, variable);
            if (y != 0)
               cout << "The number you are searching for is: " << name[i] << endl;
            else
               cout << "Not found";
            break;
         }
         case 3:
         {
            OK = 0; // Exit loop
            break;
         }
         default:
         {
            cout << "Invalid name or phone number" << endl;
            break;
         }
      }
   }

   // Free dynamically allocated memory
   delete [] name;
   delete [] telephone;
   return 0;
}
```

16.04.2025 - Lesson 3 Homework 1:

```
/*-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*/

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

class car{
   private:
      string brand;
      string model;
      string number;
      string owner_name;
   public:
      car(){
         brand="EMPTY";
         model="EMPTY";
         number="EMPTY";
         owner_name="EMPTY";
      }
   void setBrand(string b){  brand=b; }
   void setModel(string m){ model=m; }
   void setNumber(string n){ number=n;}
   void setOwner_name(string o){ owner_name=o;}
   string getBrand(){ return brand;}
   string getModel(){ return model;}
   string getNumber(){ return number;}
   string getOwner_name(){ return owner_name;}
};

int main(){
   int Number_of_Cars;
   cout << "Please insert the number of cars you want to include: ";
   cin >> Number_of_Cars;
   cout << "\n";

   car myclass[Number_of_Cars];
   string temp_brand="", temp_model="", temp_number="", temp_owner_name="";

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

      cout<<"Car Brand "<<i<<": ";
      cin>>temp_brand;
      myclass[i].setBrand(temp_brand);
```

```
        cout<<"Car Model "<<i<<": ";
        cin>>temp_model;
        myclass[i].setModel(temp_model);

        cout<<"Car Number "<<i<<": ";
        cin >> temp_number;
        myclass[i].setNumber(temp_number);


        cout<<"Owner Name "<<i<<": ";
        cin>>temp_owner_name;
        myclass[i].setOwner_name(temp_owner_name);

    }
    cout<<endl<<endl<<endl;
    cout<<"---------YOU HAVE SAVED THE FOLLOWING INFO ----------"<<endl;
    for (int i = 0; i < Number_of_Cars; i++)
    {
        cout << myclass[i].getBrand() << " " << myclass[i].getModel() << " " << myclass[i].getNumber()
<< " " << myclass[i].getOwner_name() << endl;
    }
    return 0;
}
```

16.04.2025 - Lesson 3 Homework 2:

```
/*-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
Your program must be able to search for products by name and by Type
*/

#include <iostream>
#include <string>
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 = "EMPTY";
    }

    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; }

    void subtractStock()
    {
        if (stock > 0)
        {
            stock--;
            cout << "Purchase successful. Remaining stock: " << stock << endl;
        }
        else
            cout << "Out of stock!" << endl;
    }

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

```
int main()
{
    int productCount;
    cout << "Enter number of products: ";
    cin >> productCount;

    Product products[productCount];
    string temp_brand, temp_type, temp_barcode;
    float temp_price;
    int temp_stock;

    for (int i = 0; i < productCount; i++)
    {
        cout << "\nProduct " << i + 1 << ":\n";

        cout << "  Brand: ";
        cin >> temp_brand;
        products[i].setBrand(temp_brand);

        cout << "  Type: ";
        cin >> temp_type;
        products[i].setType(temp_type);

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

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

        cout << "  Barcode: ";
        cin >> temp_barcode;
        products[i].setBarcode(temp_barcode);
    }

    int choice;
    do {
        cout << "\n----- MENU -----\n";
        cout << "1. Display All Products\n";
        cout << "2. Search by Brand\n";
        cout << "3. Search by Type\n";
        cout << "4. Buy Product (By Barcode)\n";
        cout << "0. Exit\n";
        cout << "Choice: ";
        cin >> choice;
```

```
if (choice == 1)
{
   for (int i = 0; i < productCount; i++)
      products[i].display();

}
else if (choice == 2)
{
   string search_brand;
   cout << "Enter brand to search: ";
   cin >> search_brand;
   bool found = false;

   for (int i = 0; i < productCount; i++)
   {
      if (products[i].getBrand() == search_brand)
      {
         products[i].display();
         found = true;
      }
   }
   if(!found)
      cout << "No products found with brand: " << search_brand << endl;

}
else if(choice == 3)
{
   string search_type;
   cout << "Enter type to search: ";
   cin >> search_type;
   bool found = false;

   for(int i = 0; i < productCount; i++)
   {
      if(products[i].getType() == search_type)
      {
         products[i].display();
         found = true;
      }
   }
   if(!found)
      cout << "No products found with type: " << search_type << endl;
}
else if(choice == 4)
{
   string search_barcode;
   cout << "Enter barcode to buy: ";
```

```
            cin >> search_barcode;
            bool found = false;

            for (int i = 0; i < productCount; i++)
            {
                if (products[i].getBarcode() == search_barcode) {
                    products[i].subtractStock();
                    found = true;
                    break;
                }
            }
            if (!found)
                cout << "Product with barcode " << search_barcode << " not found.\n";
        }
    }while(choice != 0);
    cout << "Goodbye!\n";
    return 0;
}
```

17.04.2025 - Lesson 4 Homework:

```
/*Break the above program into functions:

-Create a menu of options with the following info

1. enter data
2. display data
3. search by brand
4. search by model
5. search by owner
6. search by plate number
7. to exit ( 0)

After you do that you will convert the program to use dynamic (growing arrays)
*/

#include <iostream>
using namespace std;

const int MAXOBJ = 4;

class Car {
private:
    string brand;
    string model;
```

```
    string plate;
    string owner;
public:
    void setBrand(string br) { brand = br; }
    void setModel(string mod) { model = mod; }
    void setPlate(string pla) { plate = pla; }
    void setOwner(string ow) { owner = ow; }

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

    Car() {
        brand = "EMPTY";
        model = "EMPTY";
        plate = "EMPTY";
        owner = "EMPTY";
    }

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

void enterData(Car cars[]) {
    string input1, input2;
    for (int i = 0; i < MAXOBJ; i++) {
        cout << "---------CAR " << i + 1 << "----------" << endl;
        cout << "Car brand: ";
        cin >> input1;
        cars[i].setBrand(input1);

        cout << "Car model (2 words): ";
        cin >> input1 >> input2;
        cars[i].setModel(input1 + " " + input2);

        cout << "Plate number: ";
        cin >> input1;
        cars[i].setPlate(input1);

        cout << "Owner name: ";
        cin >> input1;
        cars[i].setOwner(input1);
```

```
    }
}

void displayAll(const Car cars[]) {
    cout << "------------- SAVED INFO -------------" << endl;
    for (int i = 0; i < MAXOBJ; i++) {
        cout << "------------ CAR " << i + 1 << " ------------" << endl;
        cars[i].display();
    }
}

void searchByField(const Car cars[], const string& field) {
    string query;
    cout << "Enter " << field << " to search: ";
    cin >> ws;  // clean whitespace
    getline(cin, query);

    bool found = false;
    for (int i = 0; i < MAXOBJ; i++) {
        if ((field == "brand" && cars[i].getBrand() == query) ||
            (field == "model" && cars[i].getModel() == query) ||
            (field == "plate" && cars[i].getPlate() == query) ||
            (field == "owner" && cars[i].getOwner() == query)) {
            cout << "---- Match Found at Car " << i + 1 << " ----" << endl;
            cars[i].display();
            found = true;
        }
    }
    if (!found) {
        cout << "No match found for " << field << "." << endl;
    }
}

int main() {
    Car cars[MAXOBJ];
    int choice;

    do {
        cout << "\n====== MENU ======\n";
        cout << "1. Enter Data\n";
        cout << "2. Display All\n";
        cout << "3. Search by Brand\n";
        cout << "4. Search by Model\n";
        cout << "5. Search by Plate Number\n";
        cout << "6. Search by Owner\n";
        cout << "0. Exit\n";
        cout << "Enter choice: ";
```

```
        cin >> choice;

        switch (choice) {
            case 1: enterData(cars); break;
            case 2: displayAll(cars); break;
            case 3: searchByField(cars, "brand"); break;
            case 4: searchByField(cars, "model"); break;
            case 5: searchByField(cars, "plate"); break;
            case 6: searchByField(cars, "owner"); break;
            case 0: cout << "Exiting...\n"; break;
            default: cout << "Invalid choice. Try again.\n"; break;
        }
    } while (choice != 0);

    return 0;
}
```

18.04.2025 - Lesson 5 Homework 1:

```
class A
{
public:
    int a, b;
    // constructor
    A(int _a, int _b)
    {
        a = _a; b = _b;
    }
    int Suma()
    {
        return a + b;
    }
};
```

18.04.2025 - Lesson 5 Homework 2:

```
class Table
{
public:
    int *t;
    int n;
    Table(int dim, int a, int b, int c, int d)
    {
```

```
    n = dim;
    t = new int[n];
    t[0] = a;
    for (int i = 1; i < n; i++)
      t[i] = 1 + (t[i - 1] * b + c) % d;
  }
  Table()
  {
    n = 0;
    delete[] t;
  }
  int Max()
  {
    int maxVal = t[0];
    for (int i = 1; i < n; i++) {
      if (t[i] > maxVal)
        maxVal = t[i];
    }
    return maxVal;
  }
  int Cauta(int val)
  {
    for (int i = 0; i < n; i++) {
      if (t[i] == val)
        return i;
    }
    return -1;
  }
    };
```

18.04.2025 - Lesson 5 Homework 3:

```
class Stiva
{
private:
   struct s {
    int val, val_max;
  };
  s *v;
  int n, capacity = 1;

public:
  Stiva() {
    n = 0;
    v = new s[capacity];
```

```
    }
    void Push(int x)
    {
      if(n == capacity) {
          capacity *= 2;
          s *u = v;
          v = new s[capacity];
          for(int i = 0; i < n; i++) {
              v[i] = u[i];
          }
      }
      if(n == 0) {
          v[n] = {x, x};
          n += 1;
      } else {
          v[n] = {x, max(x, v[n-1].val_max)};
          n += 1;
      }
    }
    void Pop()
    {
       if(!this->Empty()) {
          n -= 1;
      }
    }
    int Top()
    {
      if(this->Empty()) {
          return -1;
      }else {
          return v[n-1].val;
      }
    }
    int Max()
    {
      if(this->Empty()) {
          return -1;
      }
      return v[n-1].val_max;
    }
    int Empty()
    {
      return n == 0;
    }
};
```

21.04.2025 - Lesson 6 Homework:

```cpp
#include <iostream>
#include <string>
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 = "EMPTY";
    }

    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; }

    void subtractStock() {
        if (stock > 0) {
            stock--;
            cout << "Purchase successful. Remaining stock: " << stock << endl;
        } else {
            cout << "Out of stock!" << endl;
        }
    }

    void display() {
        cout << "Brand: " << brand
```

```
            << " | Type: " << type
            << " | Price: $" << price
            << " | Stock: " << stock
            << " | Barcode: " << barcode << endl;
    }

    string getProductInfo() {
        return "Brand: " + brand + " | Type: " + type + " | Barcode: " + barcode;
    }
};

void searchByBrand(Product** products, int productCount, string search_brand) {
    bool found = false;
    for (int i = 0; i < productCount; i++) {
        if (products[i]->getBrand() == search_brand) {
            cout << products[i]->getProductInfo() << endl;
            found = true;
        }
    }
    if (!found) {
        cout << "No products found with brand: " << search_brand << endl;
    }
}

void searchByType(Product** products, int productCount, string search_type) {
    bool found = false;
    for (int i = 0; i < productCount; i++) {
        if (products[i]->getType() == search_type) {
            cout << products[i]->getProductInfo() << endl;
            found = true;
        }
    }
    if (!found) {
        cout << "No products found with type: " << search_type << endl;
    }
}

void searchByBarcode(Product** products, int productCount, string search_barcode) {
    bool found = false;
    for (int i = 0; i < productCount; i++) {
        if (products[i]->getBarcode() == search_barcode) {
            cout << products[i]->getProductInfo() << endl;
            found = true;
        }
    }
    if (!found) {
        cout << "No products found with barcode: " << search_barcode << endl;
```

```
    }
}

void addMoreProducts(Product**& products, int& productCount) {
    int newProductCount;
    cout << "Enter the number of new products to add: ";
    cin >> newProductCount;

    Product** newArray = new Product*[productCount + newProductCount];

    for (int i = 0; i < productCount; i++) {
        newArray[i] = products[i];
    }

    for (int i = productCount; i < productCount + newProductCount; i++) {
        newArray[i] = new Product();

        string temp_brand, temp_type, temp_barcode;
        float temp_price;
        int temp_stock;

        cout << "\nProduct " << i + 1 << ":\n";
        cout << "  Brand: "; cin >> temp_brand;
        cout << "  Type: "; cin >> temp_type;
        cout << "  Price: "; cin >> temp_price;
        cout << "  Stock: "; cin >> temp_stock;
        cout << "  Barcode: "; cin >> temp_barcode;

        newArray[i]->setBrand(temp_brand);
        newArray[i]->setType(temp_type);
        newArray[i]->setPrice(temp_price);
        newArray[i]->setStock(temp_stock);
        newArray[i]->setBarcode(temp_barcode);
    }

    delete[] products;
    products = newArray;
    productCount += newProductCount;
}

int main() {
    int productCount;
    cout << "Enter number of products: ";
    cin >> productCount;

    Product** products = new Product*[productCount];
```

```
for (int i = 0; i < productCount; i++) {
    products[i] = new Product();

    string temp_brand, temp_type, temp_barcode;
    float temp_price;
    int temp_stock;

    cout << "\nProduct " << i + 1 << ":\n";
    cout << "  Brand: "; cin >> temp_brand;
    cout << "  Type: "; cin >> temp_type;
    cout << "  Price: "; cin >> temp_price;
    cout << "  Stock: "; cin >> temp_stock;
    cout << "  Barcode: "; cin >> temp_barcode;

    products[i]->setBrand(temp_brand);
    products[i]->setType(temp_type);
    products[i]->setPrice(temp_price);
    products[i]->setStock(temp_stock);
    products[i]->setBarcode(temp_barcode);
}

int choice;
do {
    cout << "\n----- MENU -----\n";
    cout << "1. Display All Products\n";
    cout << "2. Search by Brand\n";
    cout << "3. Search by Type\n";
    cout << "4. Search by Barcode\n";
    cout << "5. Buy Product (By Barcode)\n";
    cout << "6. Add More Products\n";
    cout << "0. Exit\n";
    cout << "Choice: ";
    cin >> choice;

    if (choice == 1) {
        for (int i = 0; i < productCount; i++) {
            products[i]->display();
        }
    }
    else if (choice == 2) {
        string search_brand;
        cout << "Enter brand to search: ";
        cin >> search_brand;
        searchByBrand(products, productCount, search_brand);
    }
    else if (choice == 3) {
        string search_type;
```

```
            cout << "Enter type to search: ";
            cin >> search_type;
            searchByType(products, productCount, search_type);
        }
        else if (choice == 4) {
            string search_barcode;
            cout << "Enter barcode to search: ";
            cin >> search_barcode;
            searchByBarcode(products, productCount, search_barcode);
        }
        else if (choice == 5) {
            string search_barcode;
            cout << "Enter barcode to buy: ";
            cin >> search_barcode;
            bool found = false;

            for (int i = 0; i < productCount; i++) {
                if (products[i]->getBarcode() == search_barcode) {
                    products[i]->subtractStock();
                    found = true;
                    break;
                }
            }
            if (!found)
                cout << "Product with barcode " << search_barcode << " not found.\n";
        }
        else if (choice == 6) {
            addMoreProducts(products, productCount);
        }
    } while (choice != 0);

    for (int i = 0; i < productCount; i++) {
        delete products[i];
    }
    delete[] products;

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

22.04.2025 - Lesson 7 Homework:

```
#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);
void searchByBrand(car *, int);

int main(){

   int ARRAYSIZE = 4;
   int nextAvailPos = 0;
   car *carsArray;
   carsArray = new car[ARRAYSIZE];
   int choice = -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:
               displayData(carsArray, nextAvailPos);
               break;
            case 3:
               searchByBrand(carsArray, nextAvailPos);
               break;
            case 0:
               break;
         }

      }while(choice != 0);
      delete[] carsArray;
      return 0;
}

//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 returns back a pointer to a 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
   newSize = 0; //the new size of the array which is calculated by adding array size + 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 new size 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;
}

// function to display all cars in the array
// accepts a pointer to the array and the number of elements in it
void displayData(car *arr, int size){
    if(size == 0){
        // if there is no data entered, notify the user
        cout << "No data to display." << endl;
        return;
    }
    // loop through the array and display brand and model for each car
    for(int i = 0; i < size; i++){
        cout << "Car [" << i << "] - Brand: " << arr[i].getBrand()
             << ", Model: " << arr[i].getModel() << endl;
    }
}

// function to search for cars by brand
// accepts a pointer to the array and the number of elements in it
void searchByBrand(car *arr, int size){
    if(size == 0){
        // if there is no data, notify the user that there's nothing to search
        cout << "No data to search." << endl;
        return;
    }
    string searchBrand;
    cout << "Enter the brand to search for: ";
    cin >> searchBrand; // read the brand to be searched from the user
    bool found = false;

    // loop through the array to find matches
    for(int i = 0; i < size; i++){
        if(arr[i].getBrand() == searchBrand){
            // if brand matches, display the car info
            cout << "Found - Car [" << i << "] - Brand: " << arr[i].getBrand()
                 << ", Model: " << arr[i].getModel() << endl;
            found = true;
        }
    }
    // if no match was found, notify the user
    if(!found){
        cout << "No car found with brand: " << searchBrand << endl;
    }
}
```

23.04.2025 - Lesson 8 Homework:

```
#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);
void editData(car *, int);

int main(){

   int ARRAYSIZE = 4;
   int nextAvailPos = 0;
   car *carsArray;
   carsArray = new car[ARRAYSIZE];
   int choice = -1;
   int searchIndex = -1;
   do{
      cout << "1. Enter Data" << endl;
      cout << "2. Display Data" << endl;
```

```
        cout << "3. Search by brand" << endl;
        cout << "4. Edit Data" << 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,tbe 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:
          {
            searchIndex = searchByBrand(carsArray, nextAvailPos);
            if (searchIndex != -1)
              editData(carsArray, searchIndex);
            else
              cout << "Car not found. Cannot edit." << endl;
            break;
          }
          case 0:
            cout << "Exiting the program. Goodbye!" << endl;
            break;
        }
    }while(choice != 0);
    return 0;
}

//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 returns back a pointer to a 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
    newSize = 0; //the new size of the array which is calculated by adding array size + 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 new size 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(car *, int) will print all data that we saved in our object array at any given
time.
//function receives as arguments a car object array and the size of the 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;
    }
}


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

    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; // Stop at the first match
        }
    }

    if(index == -1)
    {
        cout << "Nothing found!" << endl;
    }

    return index;
}

void editData(car *arr, int index)
{
    string tstr;
    cout << "Current BRAND at index " << index << ": " << arr[index].getBrand() << endl;
    cout << "Current MODEL at index " << index << ": " << arr[index].getModel() << endl;

    cout << "Enter new BRAND: ";
    cin >> tstr;
    arr[index].setBrand(tstr);

    cout << "Enter new MODEL: ";
    cin >> tstr;
    arr[index].setModel(tstr);

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

Arduino

1.

```
/*  Blink
  Turns an LED on for one second, then off for one second, repeatedly.
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

2.

```
/* 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.
*/

// 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
}
```

3.

```
/*  DigitalReadSerial
  Reads a digital input on pin 2, prints the result to the Serial Monitor
*/

// 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
}
```

4.

```
/* Fade
  This example shows how to fade an LED on pin 9 using the analogWrite()
  function.
*/

int led = 9;         // the PWM pin the LED is attached to
int brightness = 0;  // how bright the LED is
int fadeAmount = 5;  // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);
```

```
  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```