

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: Pascal Andrei

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.

Meet Andrei - Site for selection





Meet Andrei

- Pagina principală
  - Pasiuni
  - Aspirații

Valori personale

Aici sunt câteva valori personale care consider că mă definesc și cred că sunt esențiale pentru o viață plină de succes.



Dedicația



Colaborarea



Organizarea timpului

Colegiul Național "Roman-Vodă" - Strada Mihai Eminescu nr. 4, orașul Roman, județul Neamț



Meet Andrei

- Pagina principală
  - Pasiuni
  - Aspirații

Pasiuni

Călătoria

Călătoresc cu ajutorul părinților mei, cu care împărtășesc această pasiune. Pe parcursul vieții, am călătorit în diverse țări, precum Franța, Austria, Italia și Olanda. Îmi place să vizitez alte țări pentru a afla mai multe despre cultura și obiceiurile din cadrul lor și pentru a vizita obiective renumite. Aici sunt câteva fotografii pe care le-am făcut în timpul călătoriilor.



Meet Andrei

- Pagina principală
  - Pasiuni
  - Aspirații

Explorarea naturii

Aceasta este, de asemenea, o pasiune pe care o împărtășesc cu părinții mei. Încă de mic, am mers pe diferite trasee, urcând munți, explorând Delta Dunării sau lacuri naturale. Consider că este relaxant să mă plimb prin natură, admirând peisajele unice. Aici sunt câteva fotografii pe care le-am făcut în locurile în care am mers cu părinții mei.



Web Design-ul

Am descoperit pasiunea pentru Web Design acum câțiva ani, prin intermediul internetului. Consider că este ceva special să creez ceva de la zero, după bunul meu plac. Mai jos sunt câteva fotografii cu site-uri create de mine, fie pentru adăugarea lor la portofoliul personal, fie pentru utilizarea la concursuri. De asemenea, sunt ilustrate și câteva poze cu mediul în care îmi place să codez.

Meet Andrei

Pagina principală

Pasiuni

Aspirații

Web Design-ul

Am descoperit pasiunea pentru Web Design acum câțiva ani, prin intermediul internetului. Consider că este ceva special să creez ceva de la zero, după bunul meu plac. Mai jos sunt câteva fotografii cu site-uri create de mine, fie pentru adăugarea lor la portofoliul personal, fie pentru utilizarea la concursuri. De asemenea, sunt ilustrate și câteva poze cu mediul în care îmi place să codez.



Rețelistica

Am descoperit această pasiune acum câteva luni, prin intermediul unui curs oferit de Cisco pentru elevi. În octombrie, m-am înscris la un curs dedicat rețelisticii, care mă face pe zi ce trece să devin cât mai interesat. Aceasta este o pasiune pe care doresc să o urmez în viitor. Mai jos sunt câteva fotografii care ilustrează câteva activități legate de rețelistică.

Meet Andrei

Pagina principală

Pasiuni

Aspirații

Rețelistica

Am descoperit această pasiune acum câteva luni, prin intermediul unui curs oferit de Cisco pentru elevi. În octombrie, m-am înscris la un curs dedicat rețelisticii, care mă face pe zi ce trece să devin cât mai interesat. Aceasta este o pasiune pe care doresc să o urmez în viitor. Mai jos sunt câteva fotografii care ilustrează câteva activități legate de rețelistică.



Arduino

Am descoperit pasiunea pentru Arduino în clasa a 9-a. Deși nu am fost foarte impresionat atunci când mi-a fost prezentat de alte persoane, am devenit captivat atunci când am avut ocazia să folosesc eu, pe cont propriu, componentele. Această pasiune m-a îndrumat să particip la un concurs de Arduino împreună cu alți 2 colegi, mai jos fiind prezentate câteva poze de pe parcursul realizării proiectului.



Meet Andrei

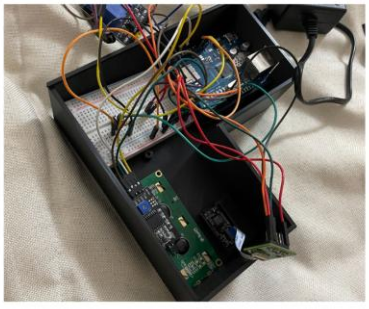
Pagina principală

Pasiuni

Aspirații

Arduino

Am descoperit pasiunea pentru Arduino în clasa a 9-a. Deși nu am fost foarte impresionat atunci când mi-a fost prezentat de alte persoane, am devenit captivat atunci când am avut ocazia să folosesc eu, pe cont propriu, componentele. Această pasiune m-a îndrumat să particip la un concurs de Arduino împreună cu alți 2 colegi, mai jos fiind prezentate câteva poze de pe parcursul realizării proiectului.



Colegiul Național "Roman-Vodă" - Strada Mihai Eminescu nr. 4, orașul Roman, județul Neamț



Meet Andrei

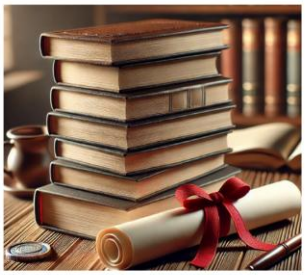
Pagina principală

Pasiuni

Aspirații

Aspirații

Până la sfârșitul liceului, plănuiesc să mă dedic pregătirii pentru examenul de bacalaureat și pentru admiterea la facultate. De asemenea, vreau să îmi dezvolt abilitățile legate de pasiunile pe care vreau să le urmez în viitor.



Meet Andrei

Pagina principală

Pasiuni

Aspirații



Doresc să intru la Facultatea de Automatică și Calculatoare din București și să obțin licența în domeniul Ingineria Sistemelor. După aceea, aspir să urmez un Master în rețelistică, fie în România, fie în Olanda.

După ce îmi obțin diploma de Master, visez să mă stabilesc în Olanda, deoarece mi-a plăcut foarte mult în acea țară atunci când am vizitat-o și este una dintre cele mai bune țări din Europa pentru a munci în domeniul IT. Doresc să mă angajez la o companie la care să muncesc cu plăcere și să întemeiez o familie.



Meet Andrei

Pagina principală

Pasiuni

Aspirații

După ce îmi obțin diploma de Master, visez să mă stabilesc în Olanda, deoarece mi-a plăcut foarte mult în acea țară atunci când am vizitat-o și este una dintre cele mai bune țări din Europa pentru a munci în domeniul IT. Doresc să mă angajez la o companie la care să muncesc cu plăcere și să întemeiez o familie.



Consider că aspirațiile mele pentru viitor sunt ridicate, dar sunt motivat să le îndeplinesc. Sunt convins că prin intermediul dedicației, muncii și asocierii cu persoanele potrivite, pot face tot ce îmi propun.

Colegiul Național "Roman-Vodă" - Strada Mihai Eminescu nr. 4, orașul Roman, județul Neamț



Programs created during the mobility

14.04.2024 - Dynamic Arrays

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

/*
Write a program where you will use a dynamic array to save characters. The user will give you numbers (integers) that range from 65 to 90.
As soon as you read the number you will convert it to character and save it into a dynamic character array. Your program will be able to make the array bigger if it is full. At the end you must display the result that you have read from the user.
*/

int main()
{
   int ARRAYSIZE = 0;  //The intial size of the array
   char *a;   //The array in which we will store the inputted ASCII codes
   char *temp;  //A temporary array, used to grow the intial array
   int n=0, tempN=0;  //n=counter, tempN = temporary number used to increment the array size
   int number; //integer (between 65 and 90) which represents the ASCII code inputted by the user
   cout<<"How many ASCII codes would you like to enter?";  //Ask the user to input the initial length of the
                                                           //array
   cin>>ARRAYSIZE;   //User input
   a = new char[ARRAYSIZE]; //We allocate space for a dynamic array which contains characters
   do{
      cout<<"Enter ASCII code number "<<n<<" -> ";  //We ask the user to give us a number
      {
         cin>>number; //User input
         if(number>=65 && number<=90){  //We check if the number given by the user is between 65 and 90
                                        //(if it is a capital letter when converted)
            a[n]=char(number); //We typecast the number from the user and store it as a character into our
                               //array
            cout<<"Character saved!"<<endl;
            n++;  //We increment the counter
         }
         else cout<<"Please enter a number between 65 and 90"<<endl; //If the number is not between 65
                                                                     //and 90, we display an error message to the user
      }

      if(n>=ARRAYSIZE){  //If the counter reaches the ARRAYSIZE, we need to grow the array or display it
         cout<<"The array is out of space! How many more characters would you like to add? (0 to
exit)"<<endl; //We ask the user hwo many more characters would they like to add to the array, or 0 if they
              //want their current array to be displayed
         cin>>tempN; //User input
         if(tempN>0){ //We check if the user wants to input more characters
            ARRAYSIZE = ARRAYSIZE + tempN; //We increase the array size
```

```
            temp = new char[ARRAYSIZE]; //We allocate memory for a bigger, temporary array
            for(int i=0; i<n; i++){ //We copy the elements from the first array to the temporary, bigger one
               temp[i] = a[i];
            }
            delete [] a; //We release the memory from the initial array
            a = temp; //We make a point to the bigger array
         }

      }
   }while(n<ARRAYSIZE);

   cout<<"Now printing the saved word:\n";
   for(int i=0; i<ARRAYSIZE;i++){
      cout<<a[i];  //We print the elements of the array
   }
   delete [] a; //We free the memory of a
   return 0;
}
```

15.04.2024 - Dynamic arrays, menu implementation

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

/*
Convert the above program in order to not only read information but also search for a name and a phone number.
*/


string getPhone(string *names,string *phone, int arraySize, string targetName){
   for(int i=0;i<arraySize;i++){
      if(names[i]==targetName) return phone[i];     //Function to get a phone number using a name
   }
   return "Name not found";
}

string getName(string *names,string *phone, int arraySize, string targetPhone){
   for(int i=0;i<arraySize;i++){
      if(phone[i]==targetPhone) return names[i];       //Function to get a name using a phone number
   }
   return "Phone number not found";
}

int main()
{
   string *names,*namesTemp;
   string *phone, *phoneTemp;
   string result, targetName, targetPhone;
   int arraySize=0;
   cout<<"Enter number of students you need to save: ";            //Initial input
   cin>>arraySize;
   names = new string[arraySize];
   phone = new string[arraySize];
   for(int i=0; i<arraySize; i++){
      cout<<"Enter Student name: ";
      cin>>names[i];
      cout<<"Enter phone number for "<<names[i]<<": ";
      cin>>phone[i];
   }
   int n=arraySize, tempN=0, oldSize;
   int option;
   do{
      cout<<endl<<endl;
      cout<<"1.Add more students\n";
      cout<<"2.Show the current students\n";
      cout<<"3.Find a students' phone number using their name\n";
      cout<<"4.Find a students' name using their phone number\n";    //Menu
      cout<<"5.Exit\n";
```

```
    cout<<"Your option: ";
    cin>>option;
    switch(option){
      case 1: {
        cout<<"How many students would you like to add?";
        cin>>tempN;
        oldSize = arraySize;
        arraySize += tempN;          ///Making the arrays bigger
        namesTemp = new string[arraySize];
        phoneTemp = new string[arraySize];
        for(int i=0; i<oldSize; i++){
          namesTemp[i] = names[i];
          phoneTemp[i] = phone[i];       //Copying the initial elements
        }
        for(int i=oldSize;i<arraySize;i++){
          cout<<"Enter Student name: ";
          cin>>namesTemp[i];                    //Inputting the additional elements from where we left off
          cout<<"Enter phone number for "<<namesTemp[i]<<": ";
          cin>>phoneTemp[i];
        }
        delete [] names;
        delete [] phone;
        names = namesTemp;
        phone = phoneTemp;
        break;
      }
      case 2:
          cout<<"You saved: \n-----------------------------------\n";
          for(int i=0;i<arraySize;i++){
            cout<<names[i]<<" | "<<phone[i]<<"\n";              //Displaying the data
          }
        break;
      case 3:
        cout<<"Input the name you want to find: ";
        cin>>targetName;
        result = getPhone(names,phone,arraySize,targetName);
        cout<<result;
        break;
      case 4:
        cout<<"Input the name you want to find: ";              //Using the functions
        cin>>targetPhone;
        result = getName(names,phone,arraySize,targetPhone);
        cout<<result;
        break;
      case 0:
        cout<<"Goodbye!";
        break;
    }
}while(option!=0);

delete [] names;
```

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

16.04.2024 - Classes examples

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

/*
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
*/


class car{
  private:
    string brand;
    string model;
    string plate;
    string owner;
  public:
    car(){
      brand="EMPTY";
      model="EMPTY";
      plate="EMPTY";
      owner="EMPTY";
    }
    void setBrand(string s){
      this->brand = s;
    }
    void setModel(string s){
      this->model = s;
    }
    void setPlate(string s){
      this->plate = s;
    }
    void setOwner(string s){
      this->owner = s;
    }
    string getBrand(){
      return this->brand;
    }
    string getModel(){
      return this->model;
    }
```

```
        string getPlate(){
            return this->plate;
        }
        string getOwner(){
            return this->owner;
        }
};

int main(){
    int arraySize = 0;
    cout<<"How many cars would you like to input? ";
    cin>>arraySize;
    cin.get();
    string tBrand, tModel, tPlate, tOwner;
    car *cars = new car[arraySize];
    for(int i=0; i<arraySize; i++){
        cout<<"Car number "<<i<<":\n";

        cout<<"Car brand: ";
        getline(cin,tBrand);
        cars[i].setBrand(tBrand);

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

        cout<<"Car plate: ";
        getline(cin, tPlate);
        cars[i].setPlate(tPlate);

        cout<<"Car owner: ";
        getline(cin, tOwner);
        cars[i].setOwner(tOwner);

    }
    cout<<"\n\n\n";
    cout<<"Here is the saved data:\n(Car Brand, Car Model, Car Plate Number, Car Owner)\n----------
----------------------------------------------------\n";
    for(int i=0; i<arraySize; i++){
        cout<<cars[i].getBrand()<<" | "<<cars[i].getModel()<<" | "<<cars[i].getPlate()<<" |
"<<cars[i].getOwner()<<"\n";
    }
        delete [] cars;
    return 0;
}
```

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

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


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

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

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

void getProductByName(int arraySize, product* products,string productName){
    int found=0;
    for(int i=0; i<arraySize; i++){
        if(products[i].getBrand() == productName){
            found=1;
            cout<<endl<<endl;
            cout<<"Brand: "<<products[i].getBrand()<<endl;
            cout<<"Type: "<<products[i].getType()<<endl;
            cout<<"Price: "<<products[i].getPrice()<<endl;
            cout<<"Stock: "<<products[i].getStock()<<endl;
            cout<<"Barcode: "<<products[i].getBarcode()<<endl;;
        }
    }
    if(found==0){
        cout<<endl<<endl;
        cout<<"Product not found!"<<endl;
    }
}

void getProductsByType(int arraySize, product* products, string productType){
    int found=0;
    for(int i=0; i<arraySize; i++){
        if(products[i].getType() == productType){
            cout<<endl<<endl;
            found=1;
            cout<<"Brand: "<<products[i].getBrand()<<endl;
            cout<<"Type: "<<products[i].getType()<<endl;
            cout<<"Price: "<<products[i].getPrice()<<endl;
            cout<<"Stock: "<<products[i].getStock()<<endl;
            cout<<"Barcode: "<<products[i].getBarcode()<<endl;
        }
    }
    if(found==0){
        cout<<endl<<endl;
```

```
        cout<<"No products found"<<endl;
    }
}

void decreaseStockUsingBarcode(int arraySize, product* products, string barcode){
    int found=0;
    for(int i=0; i<arraySize; i++){
        if(products[i].getBarcode() == barcode){
            found=1;
            int tStock = products[i].getStock();
            tStock--;
            products[i].setStock(tStock);
            cout<<"Stock decreased!"<<endl;
    }
    }
    if(found==0){
        cout<<endl<<endl;
        cout<<"Product not found!"<<endl;
    }
}

void displayProducts(int arraySize, product* products){
    cout << "Products:\n--------------------------------------" << endl;
    for(int i=0; i<arraySize; i++){
        cout << "Brand: " << products[i].getBrand() << endl;
        cout << "Type: " << products[i].getType() << endl;
        cout << "Price: " << products[i].getPrice() << endl;
        cout << "Stock: " << products[i].getStock() << endl;
        cout << "Barcode: " << products[i].getBarcode() << endl<<endl;
    }
}

int main(){
    int arraySize = 0;
    cout << "Enter the number of products: ";
    cin >> arraySize;
    product *products = new product[arraySize];
    string brand, type, barcode;
    float price;
    int stock;
    for(int i=0; i<arraySize; i++){
        cout<<"\nProduct number "<<i+1<<endl;
        cout<<"Brand: ";
        cin>>brand;
        products[i].setBrand(brand);
        cout<<"Type: ";
        cin>>type;
        products[i].setType(type);
        cout<<"Price: ";
        cin>>price;
        products[i].setPrice(price);
```

```
    cout<<"Stock: ";
    cin>>stock;
    products[i].setStock(stock);
    cout<<"Barcode: ";
    cin>>barcode;
    products[i].setBarcode(barcode);
}
cout<<"\n\n\n";
cout << "Products:\n--------------------------------------" << endl;
for(int i=0; i<arraySize; i++){
    cout << "Brand: " << products[i].getBrand() << endl;
    cout << "Type: " << products[i].getType() << endl;
    cout << "Price: " << products[i].getPrice() << endl;
    cout << "Stock: " << products[i].getStock() << endl;
    cout << "Barcode: " << products[i].getBarcode() << endl;
}
cout<<"\n\n\n";
int option;
string productName, productType;
do{
    cout<<endl<<endl;
    cout<<"1.Search a product using brand"<<endl;
    cout<<"2.Search a product using type"<<endl;
    cout<<"3.Decrease stock using barcode"<<endl;
    cout<<"4.Show all products"<<endl;
    cout<<"0.Exit"<<endl;
    cout<<"Enter your option: ";
    cin>>option;
    cout<<endl;
    switch(option){
        case 1:{
            cout<<"Enter the brand name: ";
            cin>>productName;
            getProductByName(arraySize, products, productName);
            break;
        }
        case 2:{
            string productType;
            cout<<"Enter the type name: ";
            cin>>productType;
            getProductsByType(arraySize, products, productType);
            break;
        }
        case 3:{
            string barcode;
            cout<<"Enter the barcode: ";
            cin>>barcode;
            decreaseStockUsingBarcode(arraySize, products, barcode);
            break;
        }
```

```
            case 4:{
                displayProducts(arraySize, products);
                break;
            }
            case 0:{
                cout<<"Goodbye!"<<endl;
                break;
            }
            default:{
                cout<<"Invalid option!"<<endl;
            }
        }
    }while(option!=0);
    delete [] products;
    return 0;
}
```

17.04.2024 - Car dealership project

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

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

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;
        }
};

void initialization(car *&carsArray, int& MAXOBJS){
    cout<<"How many cars would you like to enter? ";
    cin>>MAXOBJS;
    carsArray = new car[MAXOBJS];
    string userInput="";
    for(int i=0;i<MAXOBJS;i++){
        cout<<endl<<endl;
        cout<<"---------- Car number "<<i+1<<" ----------"<<endl;
        cout<<"Car Brand: ";
        cin>>userInput;
        carsArray[i].setBrand(userInput);
        cout<<"Car Model: ";
        cin>>userInput;
        carsArray[i].setModel(userInput);
        cout<<"Car Plate Number: ";
        cin.ignore();
        getline(cin, userInput);
        userInput = userInput;
        carsArray[i].setPlateNumber(userInput);
        cout<<"Car Owner: ";
        cin>>userInput;
        carsArray[i].setOwner(userInput);
    }
}

void displayData(car *carsArray, int MAXOBJS){
    cout<<endl<<endl;
    cout<<"---------- Cars ----------"<<endl;
    for(int i=0;i<MAXOBJS;i++){
        cout<<endl;;
        cout<<"---------- Car number "<<i+1<<" ----------"<<endl;
        cout<<"Car Brand: "<<carsArray[i].getBrand()<<endl;
        cout<<"Car Model: "<<carsArray[i].getModel()<<endl;
        cout<<"Car Plate Number: "<<carsArray[i].getPlateNumber()<<endl;
        cout<<"Car Owner: "<<carsArray[i].getOwner()<<endl;
    }
}

void searchByBrand(car *carsArray, int MAXOBJS){
    string userInput="";
```

```
    int ok=0;
    cout<<"Which make are you looking for? ";
    cin>>userInput;
    for(int i=0;i<MAXOBJS;i++){
        if(carsArray[i].getBrand() == userInput){
            ok=1;
            cout<<endl<<endl;
            cout<<"Car Brand: "<<carsArray[i].getBrand()<<endl;
            cout<<"Car Model: "<<carsArray[i].getModel()<<endl;
            cout<<"Car Plate Number: "<<carsArray[i].getPlateNumber()<<endl;
            cout<<"Car Owner: "<<carsArray[i].getOwner()<<endl;
        }
    }
    if(ok==0){
        cout<<"No car found with that brand."<<endl;
    }
}

void searchByModel(car *carsArray, int MAXOBJS){
    string userInput="";
    int ok=0;
    cout<<"Which model are you looking for? ";
    cin>>userInput;
    for(int i=0;i<MAXOBJS;i++){
        if(carsArray[i].getModel() == userInput){
            ok=1;
            cout<<endl<<endl;
            cout<<"Car Brand: "<<carsArray[i].getBrand()<<endl;
            cout<<"Car Model: "<<carsArray[i].getModel()<<endl;
            cout<<"Car Plate Number: "<<carsArray[i].getPlateNumber()<<endl;
            cout<<"Car Owner: "<<carsArray[i].getOwner()<<endl;
        }
    }
    if(ok==0){
        cout<<"No car found with that model."<<endl;
    }
}

void searchByOwner(car *carsArray, int MAXOBJS){
    string userInput="";
    int ok=0;
    cout<<"Which owner are you looking for? ";
    cin>>userInput;
    for(int i=0;i<MAXOBJS;i++){
        if(carsArray[i].getOwner() == userInput){
            ok=1;
            cout<<endl<<endl;
            cout<<"Car Brand: "<<carsArray[i].getBrand()<<endl;
            cout<<"Car Model: "<<carsArray[i].getModel()<<endl;
            cout<<"Car Plate Number: "<<carsArray[i].getPlateNumber()<<endl;
            cout<<"Car Owner: "<<carsArray[i].getOwner()<<endl;
```

```
        }
    }
    if(ok==0){
        cout<<"No car found with that owner."<<endl;
    }
}

void searchByPlateNumber(car *carsArray, int MAXOBJS){
    string userInput="";
    int ok=0;
    cout<<"Which plate number are you looking for? ";
    cin>>userInput;
    for(int i=0;i<MAXOBJS;i++){
        if(carsArray[i].getPlateNumber() == userInput){
            ok=1;
            cout<<endl<<endl;
            cout<<"Car Brand: "<<carsArray[i].getBrand()<<endl;
            cout<<"Car Model: "<<carsArray[i].getModel()<<endl;
            cout<<"Car Plate Number: "<<carsArray[i].getPlateNumber()<<endl;
            cout<<"Car Owner: "<<carsArray[i].getOwner()<<endl;
        }
    }
    if(ok==0){
        cout<<"No car found with that plate number."<<endl;
    }
}

void growArray(car *&carsArray, int& MAXOBJS){
    cout<<"How many more cars would you like to add? "<<endl;
    int tmpN;
    string userInput="";
    cin>>tmpN;
    car *temp = new car[MAXOBJS + tmpN];
    for(int i=0;i<MAXOBJS;i++){
        temp[i] = carsArray[i];
    }
    for(int i=MAXOBJS;i<MAXOBJS+tmpN;i++){
        cout<<endl<<endl;
        cout<<"---------- Car number "<<i+1<<" ----------"<<endl;
        cout<<"Car Brand: ";
        cin>>userInput;
        temp[i].setBrand(userInput);
        cout<<"Car Model: ";
        cin>>userInput;
        temp[i].setModel(userInput);
        cout<<"Car Plate Number: ";
        cin.ignore();
        getline(cin, userInput);
        userInput = userInput;
        temp[i].setPlateNumber(userInput);
        cout<<"Car Owner: ";
```

```
        cin>>userInput;
        temp[i].setOwner(userInput);
    }
    delete[] carsArray;
    carsArray = temp;
    MAXOBJS += tmpN;
}

int main(){
    int MAXOBJS=0;
    car *carsArray;
    int option=0;
    do{
        cout<<endl<<endl;
        cout<<"---------- Menu ----------"<<endl;
        cout<<"1. Enter data"<<endl;
        cout<<"2. Display data"<<endl;
        cout<<"3. Search by brand"<<endl;
        cout<<"4. Search by model"<<endl;
        cout<<"5. Search by owner"<<endl;
        cout<<"6. Search by plate number"<<endl;
        cout<<"7. Add more cars"<<endl;
        cout<<"0. Exit"<<endl;
        cout<<"Select an option: ";
        cin>>option;
        switch(option){
            case 1:
                if(MAXOBJS!=0){
                    cout<<"Data already entered. Please use option 7 to add more cars."<<endl;
                    break;
                }
                initialization(carsArray, MAXOBJS);
                break;
            case 2:
                if(MAXOBJS==0){
                    cout<<"No data available. Please enter data first."<<endl;
                    break;
                }
                displayData(carsArray, MAXOBJS);
                break;
            case 3:
                if(MAXOBJS==0){
                    cout<<"No data available. Please enter data first."<<endl;
                    break;
                }
                searchByBrand(carsArray, MAXOBJS);
                break;
            case 4:
                if(MAXOBJS==0){
                    cout<<"No data available. Please enter data first."<<endl;
                    break;
```

```
            }
            searchByModel(carsArray, MAXOBJS);
            break;
        case 5:
            if(MAXOBJS==0){
                cout<<"No data available. Please enter data first."<<endl;
                break;
            }
            searchByOwner(carsArray, MAXOBJS);
            break;
        case 6:
            if(MAXOBJS==0){
                cout<<"No data available. Please enter data first."<<endl;
                break;
            }
            searchByPlateNumber(carsArray, MAXOBJS);
            break;
        case 7:
            if(MAXOBJS==0){
                cout<<"No data available. Please enter data first."<<endl;
                break;
            }
            growArray(carsArray, MAXOBJS);
            break;
        case 0:
            cout<<"Exiting..."<<endl;
            break;
        default:
            cout<<"Invalid option. Please try again."<<endl;
    }
  }while(option!=0);
  delete [] carsArray;
  return 0;
}
```

18.04.2024 - Supermarket project

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

class Product {
  private:
    string brand;
    string type;
    float price;
    int stockRemaining;
    string barcode;
  public:
    Product() {
      brand = "EMPTY";
      type = "EMPTY";
      price = 0.0;
      stockRemaining = 0;
      barcode = "EMPTY";
    }
    void setBrand(string B) {
      brand = B;
    }
    void setType(string T) {
      type = T;
    }
    void setPrice(float P) {
      price = P;
    }
    void setStockRemaining(int S) {
      stockRemaining = S;
    }
    void setBarcode(string B) {
      barcode = B;
    }
    string getBrand() {
      return brand;
    }
    string getType() {
      return type;
    }
    float getPrice() {
      return price;
    }
    int getStockRemaining() {
      return stockRemaining;
```

```
    }
    string getBarcode() {
      return barcode;
    }
};

void initialization(Product *&productsArray, int &MAX_PRODUCTS) {
  cout << "How many products would you like to enter? ";
  cin >> MAX_PRODUCTS;
  productsArray = new Product[MAX_PRODUCTS];
  string userInput;
  float tmpPrice;
  int tmpStock;

  for (int i = 0; i < MAX_PRODUCTS; i++) {
    cout << endl << endl;
    cout << "---------- Product number " << i + 1 << " ----------" << endl;
    cout << "Product Brand: ";
    cin >> userInput;
    productsArray[i].setBrand(userInput);
    cout << "Product Type: ";
    cin >> userInput;
    productsArray[i].setType(userInput);
    cout << "Product Price: ";
    cin >> tmpPrice;
    productsArray[i].setPrice(tmpPrice);
    cout << "Stock Remaining: ";
    cin >> tmpStock;
    productsArray[i].setStockRemaining(tmpStock);
    cout << "Product Barcode: ";
    cin >> userInput;
    productsArray[i].setBarcode(userInput);
  }
}

void displayData(Product *productsArray, int MAX_PRODUCTS) {
  cout << endl << endl;
  cout << "---------- Products ----------" << endl;
  for (int i = 0; i < MAX_PRODUCTS; i++) {
    cout << endl;
    cout << "---------- Product number " << i + 1 << " ----------" << endl;
    cout << "Product Brand: " << productsArray[i].getBrand() << endl;
    cout << "Product Type: " << productsArray[i].getType() << endl;
    cout << "Product Price: " << productsArray[i].getPrice() << endl;
    cout << "Stock Remaining: " << productsArray[i].getStockRemaining() << endl;
    cout << "Product Barcode: " << productsArray[i].getBarcode() << endl;
  }
}

void searchByBrand(Product *productsArray, int MAX_PRODUCTS) {
  string userInput;
```

```
    int found = 0;
    cout << "Which product brand are you looking for? ";
    cin >> userInput;
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (productsArray[i].getBrand() == userInput) {
            found = 1;
            cout << endl;
            cout << "Product Brand: " << productsArray[i].getBrand() << endl;
            cout << "Product Type: " << productsArray[i].getType() << endl;
            cout << "Product Price: " << productsArray[i].getPrice() << endl;
            cout << "Stock Remaining: " << productsArray[i].getStockRemaining() << endl;
            cout << "Product Barcode: " << productsArray[i].getBarcode() << endl;
        }
    }
    if (!found) {
        cout << "No product found with that brand." << endl;
    }
}

void searchByType(Product *productsArray, int MAX_PRODUCTS) {
    string userInput;
    int found = 0;
    cout << "Which product type are you looking for? ";
    cin >> userInput;
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (productsArray[i].getType() == userInput) {
            found = 1;
            cout << endl;
            cout << "Product Brand: " << productsArray[i].getBrand() << endl;
            cout << "Product Type: " << productsArray[i].getType() << endl;
            cout << "Product Price: " << productsArray[i].getPrice() << endl;
            cout << "Stock Remaining: " << productsArray[i].getStockRemaining() << endl;
            cout << "Product Barcode: " << productsArray[i].getBarcode() << endl;
        }
    }
    if (!found) {
        cout << "No product found with that type." << endl;
    }
}

void decreaseStock(Product *productsArray, int MAX_PRODUCTS) {
    string barcodeInput;
    int quantity;
    int found = 0;
    cout << "Enter the barcode of the product to decrease stock: ";
    cin >> barcodeInput;
    cout << "Enter the quantity to decrease: ";
    cin >> quantity;

    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (productsArray[i].getBarcode() == barcodeInput) {
```

```
            found = 1;
            if (productsArray[i].getStockRemaining() >= quantity) {
                productsArray[i].setStockRemaining(productsArray[i].getStockRemaining() - quantity);
                cout << "Stock updated successfully." << endl;
            } else {
                cout << "Not enough stock available." << endl;
            }
            break;
        }
    }
    if (!found) {
        cout << "No product found with that barcode." << endl;
    }
}

void growArray(Product *&productsArray, int &MAX_PRODUCTS) {
    cout << "How many more products would you like to add? ";
    int additionalProducts;
    cin >> additionalProducts;

    Product *temp = new Product[MAX_PRODUCTS + additionalProducts];
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        temp[i] = productsArray[i];
    }

    string userInput;
    float tmpPrice;
    int tmpStock;

    for (int i = MAX_PRODUCTS; i < MAX_PRODUCTS + additionalProducts; i++) {
        cout << endl << endl;
        cout << "---------- Product number " << i + 1 << " ----------" << endl;
        cout << "Product Brand: ";
        cin >> userInput;
        temp[i].setBrand(userInput);
        cout << "Product Type: ";
        cin >> userInput;
        temp[i].setType(userInput);
        cout << "Product Price: ";
        cin >> tmpPrice;
        temp[i].setPrice(tmpPrice);
        cout << "Stock Remaining: ";
        cin >> tmpStock;
        temp[i].setStockRemaining(tmpStock);
        cout << "Product Barcode: ";
        cin >> userInput;
        temp[i].setBarcode(userInput);
    }

    delete[] productsArray;
    productsArray = temp;
```

```
    MAX_PRODUCTS += additionalProducts;
}

int main() {
    int MAX_PRODUCTS = 0;
    Product *productsArray = nullptr;
    int option = 0;

    do {
        cout << endl << endl;
        cout << "---------- Menu ----------" << endl;
        cout << "1. Enter data" << endl;
        cout << "2. Display data" << endl;
        cout << "3. Search by brand" << endl;
        cout << "4. Search by type" << endl;
        cout << "5. Decrease stock by barcode" << endl;
        cout << "6. Add more products" << endl;
        cout << "0. Exit" << endl;
        cout << "Select an option: ";
        cin >> option;

        switch (option) {
            case 1:
                if (MAX_PRODUCTS != 0) {
                    cout << "Data already entered. Please use option 6 to add more products." << endl;
                    break;
                }
                initialization(productsArray, MAX_PRODUCTS);
                break;
            case 2:
                if (MAX_PRODUCTS == 0) {
                    cout << "No data available. Please enter data first." << endl;
                    break;
                }
                displayData(productsArray, MAX_PRODUCTS);
                break;
            case 3:
                if (MAX_PRODUCTS == 0) {
                    cout << "No data available. Please enter data first." << endl;
                    break;
                }
                searchByBrand(productsArray, MAX_PRODUCTS);
                break;
            case 4:
                if (MAX_PRODUCTS == 0) {
                    cout << "No data available. Please enter data first." << endl;
                    break;
                }
                searchByType(productsArray, MAX_PRODUCTS);
                break;
            case 5:
```

```
            if (MAX_PRODUCTS == 0) {
                cout << "No data available. Please enter data first." << endl;
                break;
            }
            decreaseStock(productsArray, MAX_PRODUCTS);
            break;
        case 6:
            if (MAX_PRODUCTS == 0) {
                cout << "No data available. Please enter data first." << endl;
                break;
            }
            growArray(productsArray, MAX_PRODUCTS);
            break;
        case 0:
            cout << "Exiting..." << endl;
            break;
        default:
            cout << "Invalid option. Please try again." << endl;
    }
  } while (option != 0);

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

21.04.2024 - Supermarket project, optimized function for entering data

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

class Product {
  private:
    string brand;
    string type;
    float price;
    int stockRemaining;
    string barcode;
  public:
    Product() {
      brand = "EMPTY";
      type = "EMPTY";
      price = 0.0;
      stockRemaining = 0;
      barcode = "EMPTY";
    }
    void setBrand(string B) {
      brand = B;
    }
    void setType(string T) {
      type = T;
    }
    void setPrice(float P) {
      price = P;
    }
    void setStockRemaining(int S) {
      stockRemaining = S;
    }
    void setBarcode(string B) {
      barcode = B;
    }
    string getBrand() {
      return brand;
    }
    string getType() {
      return type;
    }
    float getPrice() {
      return price;
    }
    int getStockRemaining() {
      return stockRemaining;
```

```
    }
    string getBarcode() {
      return barcode;
    }
};

Product* enterData(Product *productsArray, int &MAX_PRODUCTS, int n) {
  Product* tempArray = new Product[MAX_PRODUCTS+n];
  for(int i=0;i<MAX_PRODUCTS;i++){
    tempArray[i] = productsArray[i];
  }

  string userInput;
  float tmpPrice;
  int tmpStock;

  for (int i = MAX_PRODUCTS; i < MAX_PRODUCTS+n; i++) {
    cout << endl << endl;
    cout << "---------- Product number " << i + 1 << " ----------" << endl;
    cout << "Product Brand: ";
    cin >> userInput;
    tempArray[i].setBrand(userInput);
    cout << "Product Type: ";
    cin >> userInput;
    tempArray[i].setType(userInput);
    cout << "Product Price: ";
    cin >> tmpPrice;
    tempArray[i].setPrice(tmpPrice);
    cout << "Stock Remaining: ";
    cin >> tmpStock;
    tempArray[i].setStockRemaining(tmpStock);
    cout << "Product Barcode: ";
    cin >> userInput;
    tempArray[i].setBarcode(userInput);
  }
  MAX_PRODUCTS += n;
  delete [] productsArray;
  return tempArray;
}

void displayData(Product *productsArray, int MAX_PRODUCTS) {
  cout << endl << endl;
  cout << "---------- Products ----------" << endl;
  for (int i = 0; i < MAX_PRODUCTS; i++) {
    cout << endl;
    cout << "---------- Product number " << i + 1 << " ----------" << endl;
    cout << "Product Brand: " << productsArray[i].getBrand() << endl;
    cout << "Product Type: " << productsArray[i].getType() << endl;
    cout << "Product Price: " << productsArray[i].getPrice() << endl;
    cout << "Stock Remaining: " << productsArray[i].getStockRemaining() << endl;
    cout << "Product Barcode: " << productsArray[i].getBarcode() << endl;
```

```
    }
}

void searchByBrand(Product *productsArray, int MAX_PRODUCTS) {
    string userInput;
    int found = 0;
    cout << "Which product brand are you looking for? ";
    cin >> userInput;
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (productsArray[i].getBrand() == userInput) {
            found = 1;
            cout << endl;
            cout << "Product Brand: " << productsArray[i].getBrand() << endl;
            cout << "Product Type: " << productsArray[i].getType() << endl;
            cout << "Product Price: " << productsArray[i].getPrice() << endl;
            cout << "Stock Remaining: " << productsArray[i].getStockRemaining() << endl;
            cout << "Product Barcode: " << productsArray[i].getBarcode() << endl;
        }
    }
    if (!found) {
        cout << "No product found with that brand." << endl;
    }
}

void searchByType(Product *productsArray, int MAX_PRODUCTS) {
    string userInput;
    int found = 0;
    cout << "Which product type are you looking for? ";
    cin >> userInput;
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (productsArray[i].getType() == userInput) {
            found = 1;
            cout << endl;
            cout << "Product Brand: " << productsArray[i].getBrand() << endl;
            cout << "Product Type: " << productsArray[i].getType() << endl;
            cout << "Product Price: " << productsArray[i].getPrice() << endl;
            cout << "Stock Remaining: " << productsArray[i].getStockRemaining() << endl;
            cout << "Product Barcode: " << productsArray[i].getBarcode() << endl;
        }
    }
    if (!found) {
        cout << "No product found with that type." << endl;
    }
}

void decreaseStock(Product *productsArray, int MAX_PRODUCTS) {
    string barcodeInput;
    int quantity;
    int found = 0;
    cout << "Enter the barcode of the product to decrease stock: ";
    cin >> barcodeInput;
```

```
    cout << "Enter the quantity to decrease: ";
    cin >> quantity;

    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (productsArray[i].getBarcode() == barcodeInput) {
            found = 1;
            if (productsArray[i].getStockRemaining() >= quantity) {
                productsArray[i].setStockRemaining(productsArray[i].getStockRemaining() - quantity);
                cout << "Stock updated successfully." << endl;
            } else {
                cout << "Not enough stock available." << endl;
            }
            break;
        }
    }
    if (!found) {
        cout << "No product found with that barcode." << endl;
    }
}

void growArray(Product *&productsArray, int &MAX_PRODUCTS) {
    cout << "How many more products would you like to add? ";
    int additionalProducts;
    cin >> additionalProducts;

    Product *temp = new Product[MAX_PRODUCTS + additionalProducts];
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        temp[i] = productsArray[i];
    }

    string userInput;
    float tmpPrice;
    int tmpStock;

    for (int i = MAX_PRODUCTS; i < MAX_PRODUCTS + additionalProducts; i++) {
        cout << endl << endl;
        cout << "---------- Product number " << i + 1 << " ----------" << endl;
        cout << "Product Brand: ";
        cin >> userInput;
        temp[i].setBrand(userInput);
        cout << "Product Type: ";
        cin >> userInput;
        temp[i].setType(userInput);
        cout << "Product Price: ";
        cin >> tmpPrice;
        temp[i].setPrice(tmpPrice);
        cout << "Stock Remaining: ";
        cin >> tmpStock;
        temp[i].setStockRemaining(tmpStock);
        cout << "Product Barcode: ";
        cin >> userInput;
```

```
        temp[i].setBarcode(userInput);
    }

    delete[] productsArray;
    productsArray = temp;
    MAX_PRODUCTS += additionalProducts;
}

int main() {
    int MAX_PRODUCTS = 0, n=0;
    Product *productsArray = nullptr;
    int option = 0;

    do {
        cout << endl << endl;
        cout << "---------- Menu ----------" << endl;
        cout << "1. Enter data" << endl;
        cout << "2. Display data" << endl;
        cout << "3. Search by brand" << endl;
        cout << "4. Search by type" << endl;
        cout << "5. Decrease stock by barcode" << endl;
        cout << "0. Exit" << endl;
        cout << "Select an option: ";
        cin >> option;

        switch (option) {
            case 1:
                cout << "How many products would you like to enter? ";
                cin >> n;
                productsArray = enterData(productsArray, MAX_PRODUCTS, n);
                break;
            case 2:
                if (MAX_PRODUCTS == 0) {
                    cout << "No data available. Please enter data first." << endl;
                    break;
                }
                displayData(productsArray, MAX_PRODUCTS);
                break;
            case 3:
                if (MAX_PRODUCTS == 0) {
                    cout << "No data available. Please enter data first." << endl;
                    break;
                }
                searchByBrand(productsArray, MAX_PRODUCTS);
                break;
            case 4:
                if (MAX_PRODUCTS == 0) {
                    cout << "No data available. Please enter data first." << endl;
                    break;
                }
                searchByType(productsArray, MAX_PRODUCTS);
```

```
            break;
        case 5:
            if (MAX_PRODUCTS == 0) {
                cout << "No data available. Please enter data first." << endl;
                break;
            }
            decreaseStock(productsArray, MAX_PRODUCTS);
            break;
        case 0:
            cout << "Exiting..." << endl;
            break;
        default:
            cout << "Invalid option. Please try again." << endl;
    }
  } while (option != 0);
  delete[] productsArray;
  return 0;
}
```

22.04.2024 - Car dealership project, optimized and organized

```
#include <iostream>
#include <string>
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, string);

int main(){
   int ARRAYSIZE=0;
   int nextAvailPos=0;
   car *carsArray;
   carsArray=new car[ARRAYSIZE];
   int choice=-1;
   string tString;
   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;
        cout<<"Your choice: ";
        cin>>choice;

        switch(choice){
           case 1:
              carsArray=readData(carsArray,ARRAYSIZE, nextAvailPos);
              break;
           case 2:
              displayData(carsArray, ARRAYSIZE);
              break;
           case 3:
              cout<<"Enter the brand you want to search for: ";
              cin>>tString;
              searchByBrand(carsArray, ARRAYSIZE, tString);
              break;
           case 0:
              cout<<"Exiting...";
              break;
           default:
              cout<<"Please enter a valid choice"<<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 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 nad 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++;

    }
    return arr;
}

//function void displayData(car *, int)
//the function displays the data from the array of cars
//the function accepts a pointer to a car object(the array)
//and the size of the array
void displayData(car *arr, int arrSize){
    cout<<endl;
    for (int i=0;i<arrSize;i++){
        cout<<"Car["<<i<<"]"<<endl;
        cout<<"Brand: "<<arr[i].getBrand()<<endl;
        cout<<"Model: "<<arr[i].getModel()<<endl<<endl;
    }
}
//function void searchByBrand(car *, int, string)
//the function searches for a car by brand
//the function accepts a pointer to a car object(the array)
//the size of the array and the brand to search for (string)
//the function displays the car if it was found
//or a message if it was not found
void searchByBrand(car *arr, int arrSize, string brand){
    cout<<endl;
    int found=0; //variable to check if the car was found or not
    for (int i=0;i<arrSize;i++){
        if(arr[i].getBrand()==brand){
            found=1;//
            cout<<"Car["<<i<<"]"<<endl;
            cout<<"Brand: "<<arr[i].getBrand()<<endl;
            cout<<"Model: "<<arr[i].getModel()<<endl<<endl;
        }
    }
    if(found==0){ //if the car was not found
        cout<<"No car was found"<<endl;
    }
}
```

23.04.2024 - Car dealership project, function for editing data

```
#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);

int main(){

   int ARRAYSIZE=4;
   int nextAvailPos=0;
   car *carsArray;
   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 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,
              //the variable ARRAYSIZE is the TOTAL SIZE OF THE ARRAY,
              //whidh 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 editData function
                 //the edit data function will firts, show you what is
                 //saved at the current location and then give you the option
                 //to input new data.
                 searchIndex = searchByBrand(carsArray, nextAvailPos);
                 editData(carsArray, nextAvailPos, searchIndex);
              break;
           case 0:
              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 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
   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 nad 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++;

    }
    return arr;
}

//void displayData(car*, int)
//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<<endl;
    }
}

//int searchByBrand(car*, int)
//the function receives as arguments
//a car object array and the size of the array
//the function will search for the index of the car with the specific brand
//and will return the index of the car
//or -1 if no car was found
int searchByBrand(car* arr, int arrSize){
    string tstr=""; //temporary string to read data from keyboard
    int index=-1;
    cout<<endl;
    cout<<"Enter the brand you are looking for: ";
```

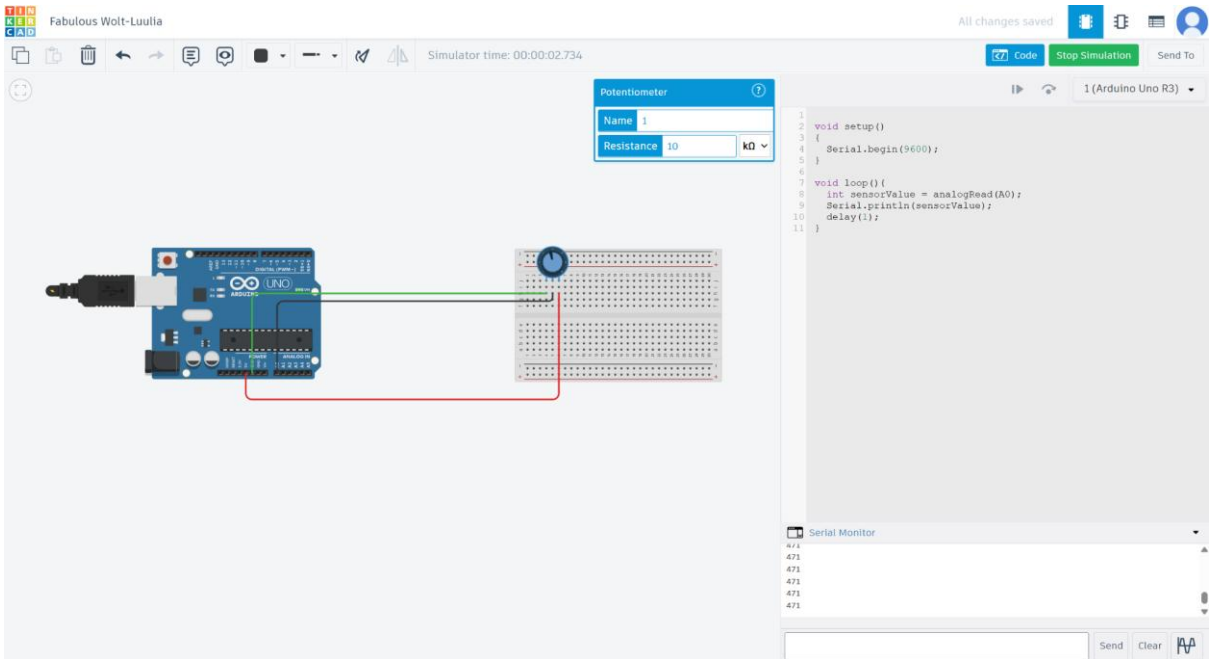
```
    cin>>tstr;
    for(int i=0;i<arrSize;i++){
        if(arr[i].getBrand()==tstr){
            cout<<"Index found at position "<<i<<endl<<endl;
            index = i;
        }
    }
    if(index==-1) cout<<"Nothing found!"<<endl<<endl;
    return index;
}

//void editData(car*, int)
//the function receives as arguments
//a car object array, the size of the array and the index of the car that needs editing
//the function will display the current data of the car
//and then will ask the user if they want to change the brand, the model or both
//the function will then ask the user to input the new data
//and will edit the data inside the array
void editData(car* arr, int arrSize, int index){
    if(index!=-1){
        string tstr="";
        int option = -1;
        cout<<"Current data for car["<<index<<"]"<<endl;
        cout<<"Brand: "<<arr[index].getBrand()<<endl;
        cout<<"Model: "<<arr[index].getModel()<<endl;
        cout<<"What do you want to change?"<<endl;
        cout<<"1. Brand"<<endl;
        cout<<"2. Model"<<endl;
        cout<<"3. Both"<<endl;
        cout<<"0. Exit"<<endl;
        cout<<"Your choice: ";
        cin>>option;
        switch(option){ //Menu for editing data
            case 1:
                cout<<"Enter new brand: ";
                cin>>tstr;
                arr[index].setBrand(tstr);
                break;
            case 2:
                cout<<"Enter new model: ";
                cin>>tstr;
                arr[index].setModel(tstr);
                break;
            case 3:
                cout<<"Enter new brand: ";
                cin>>tstr;
                arr[index].setBrand(tstr);
                cout<<"Enter new model: ";
                cin>>tstr;
                arr[index].setModel(tstr);
                break;
```

```
            case 0:
                break;
        }
    }
}
```

Arduino projects

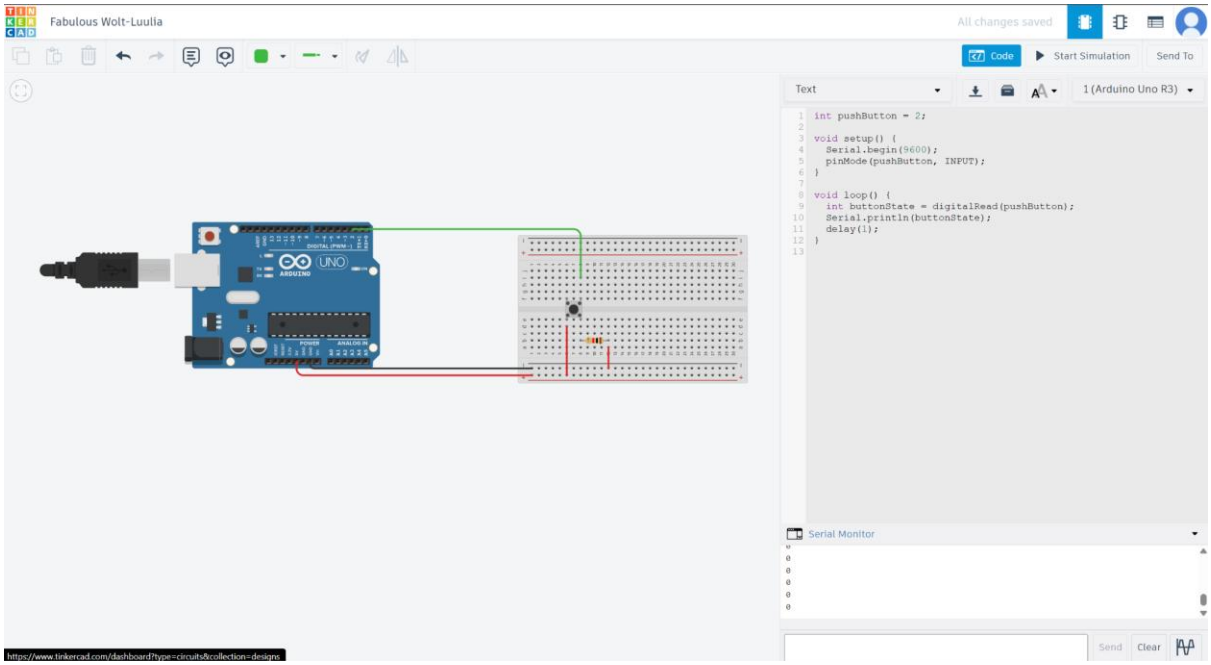
Reading analog value from a potentiometer




```
void setup()
{
  Serial.begin(9600);
}

void loop(){
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(1);
}
```

Reading and displaying the state of a button




```
int pushButton = 2;

void setup() {
  Serial.begin(9600);
  pinMode(pushButton, INPUT);
}

void loop() {
  int buttonState = digitalRead(pushButton);
  Serial.println(buttonState);
  delay(1);
}
```

Finding a workplace - Tutorial

CUM SĂ-ȚI GĂSEȘTI UN LOC DE MUNCĂ

Ciuștea Ioana-Alexandra
Enea Alexandru-Petronel
Grigoraș Darius-George
Laic Andrei-Ștefan
Murariu Cosmin-Andrei
Pascal Andrei





Ce este un CV și cum să redactezi unul

Pascal Andrei

Ce este un CV?

***CV**-ul (Curriculum Vitae) reprezintă prezentarea cursului vieții în fața potențialului angajator. Acesta oferă o imagine de ansamblu asupra **abilităților, experienței de muncă și realizărilor**.*



Ce trebuie să conțină un CV?

Educație
Experiență profesională
Proiecte
Activități extracuriculare
Premii
Certificări
Abilități

Cum îți poți crea un CV?

Europass
Overleaf
Google Templates

Ce trebuie să ai în vedere atunci când creezi un CV?

- Evită modelele cu multe culori - **maxim o culoare**
- Alege un model care îți oferă destul **spațiu**
- Găsește un model care oferă toate **elementele** de care ai nevoie

Sfaturi pentru fiecare secțiune

- **Date personale**
  - Folosește un **email destinat** uzului profesional
  - Adaugă **prefixul internațional** pentru numărul de telefon (ex. +40)
  - Adaugă informații despre contul de **LinkedIn, GitHub,** etc.
- **Educație**
  - Pune secțiunea de educație **prima**
  - Menționează **cursurile relevante**
  - Adaugă **media generală** dacă este avantajoasă (peste 9)
  - Studiile liceale **nu** sunt foarte relevante

Sfaturi pentru fiecare secțiune

- **Experiență profesională**
  - Folosește **puncte de marcaj**
  - Include **informațiile esențiale** (numele funcției, numele companiei, perioada de angajare, locația)
  - Descrie clar **responsabilitățile**
  - Pune accent pe **realizări**
  - Folosește verbe precum ***am implementat***, ***am îmbunătățit***, ***am gestionat***
  - Scoate în evidență **experiențele relevante** pentru postul pentru care aplici

Sfaturi pentru fiecare secțiune

- **Proiecte**
  - Folosește **puncte de marcaj**
  - Evidențiază **cuvintele cheie**
  - Adaugă **datele de realizare** a fiecărui proiect
  - Folosește verbe precum ***am implementat***, ***am îmbunătățit***, ***am gestionat***
  - Menționează **impactul** adus de proiecte (folosește numere și procentaje)
- **Abilități**
  - Menționează **nivelul** pentru fiecare (**Fii sincer!**)
  - Este recomandat să incluzi cel puțin o abilitate de nivel **intermediar/avansat**
  - **Evită** reprezentările **grafice** (grafice, desene, etc.)

Sfaturi pentru fiecare secțiune

- **Activități extracuriculare / Voluntariat**
  - Enumerează-le în ordine **invers cronologică**
  - Evidențiază **impactul** adus de activități

- **Certificări / Premii**
  - Adaugă doar certificările/premiile **relevante**
  - Menționează **data** și alte informații legate de **context** (abilități dobândite, mijloace folosite, etc.)