

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

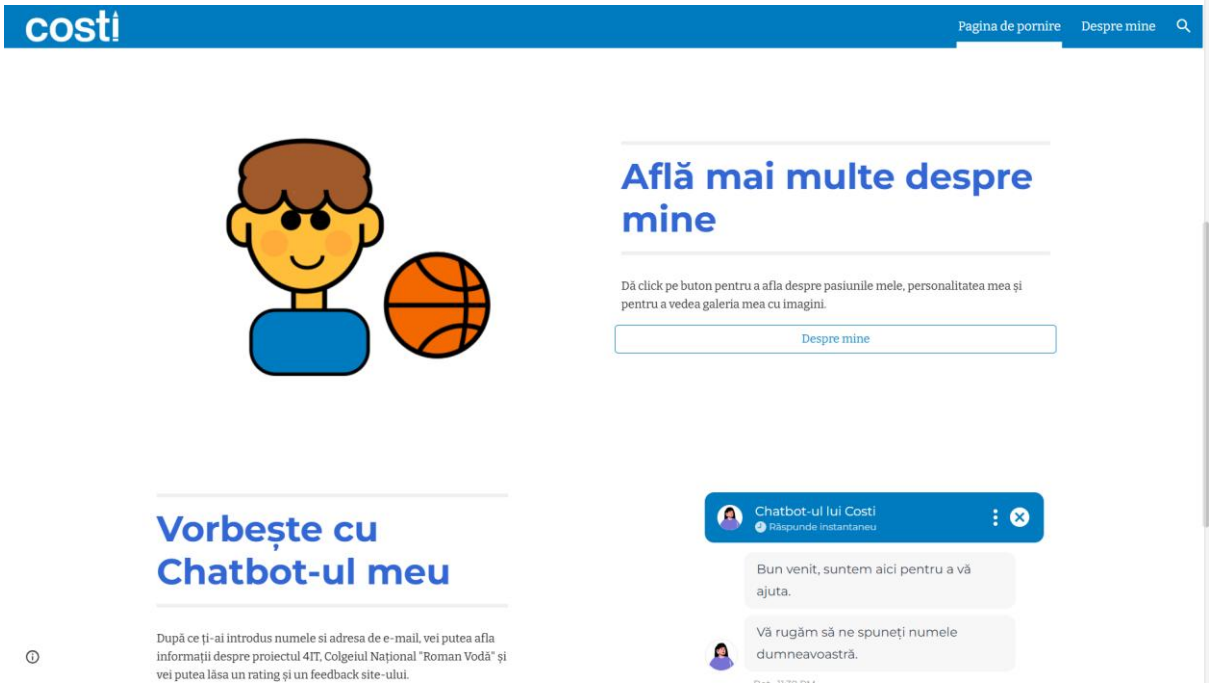
Practice tutor: Lazaros Lazarou

Participant: Buzdugan Constantin

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.

Costi - Personal website



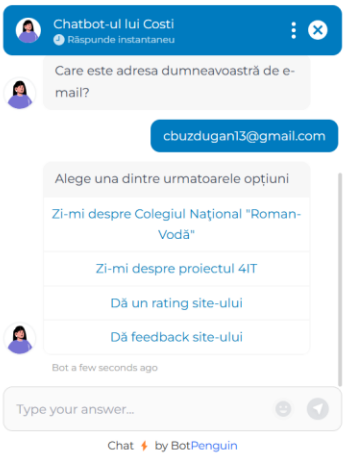


costi

Pagina de pornire | Despre mine

Vorbește cu Chatbot-ul meu

După ce ți-ai introdus numele si adresa de e-mail, vei putea afla informații despre proiectul 4IT, Colgeiul Național "Roman Vodă" și vei putea lăsa un rating și un feedback site-ului.



Colegiul Național "Roman Vodă", Roman, Neamț

costi

Pagina de pornire | Despre mine

Despre mine

Mă numesc Buzdugan Constantin, am 17 ani și sunt elev la Colegiul Național "Roman Vodă". Sunt o persoană deschisă, calmă, ambițioasă și mereu dornică să explorez noi oportunități. Printre pasiunile mele se numără:



Programarea

Programarea mă ajută să găsesc soluții practice și



Șahul

Șahul mă ajută să îmi dezvolt gândirea strategică și



Baschetul

Baschetul mă menține activ și îmi oferă bucuria

Programarea mă ajută să găsesc soluții practice și este distractivă.

Șahul mă ajută să îmi dezvolt gândirea strategică și să rămân concentrat.

Baschetul mă menține activ și îmi oferă bucuria jocului în echipă.



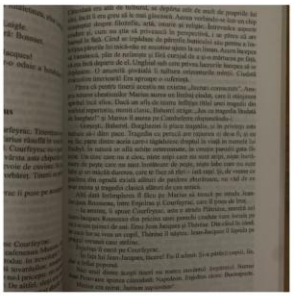
Sala

Antrenamentele la sală îmi îmbunătățesc disciplina și forma fizică.



Filme și seriale

Șahul mă ajută să îmi dezvolt gândirea strategică și să rămân concentrat.



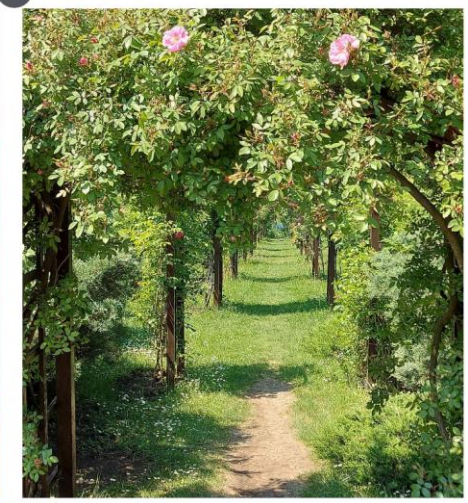
Cititul

Cititul este modul meu de a învăța și de a explora noi perspective.

Galerie

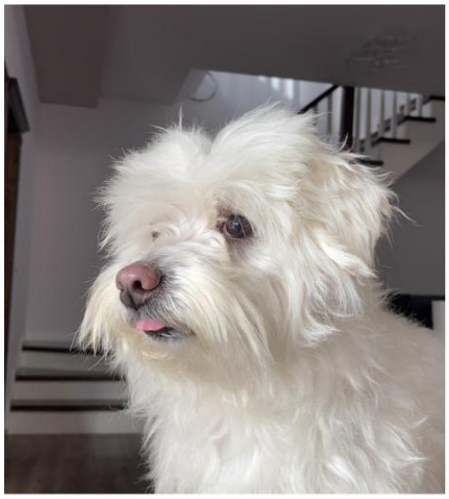
costi
Pagina de pornire Despre mine

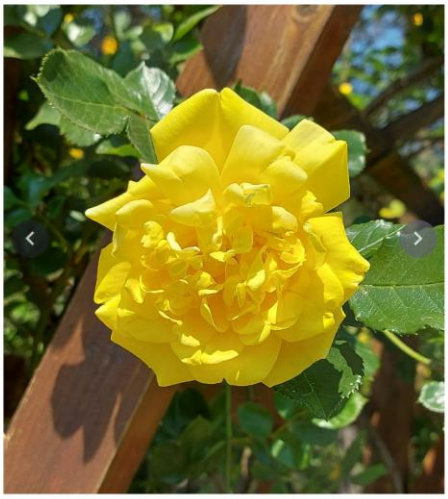




costi

Pagina de pornire Despre mine

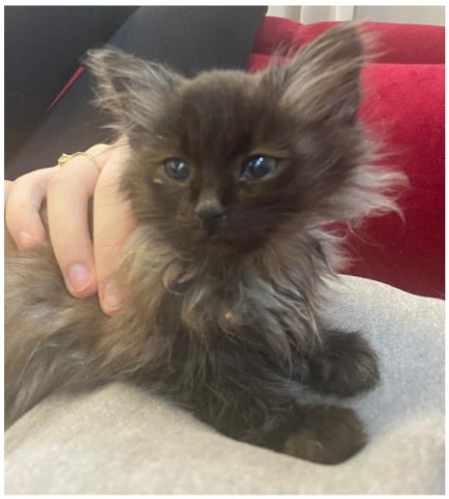


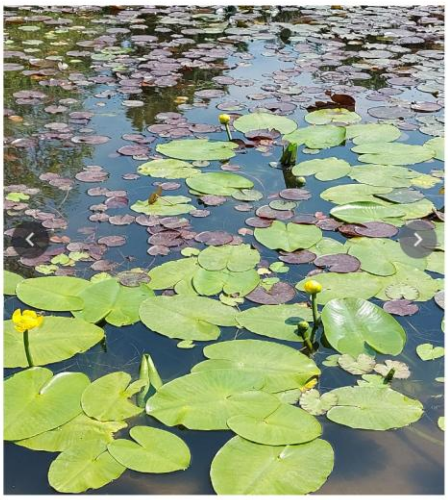


Colegiul Național "Roman Vodă", Roman, Neamț

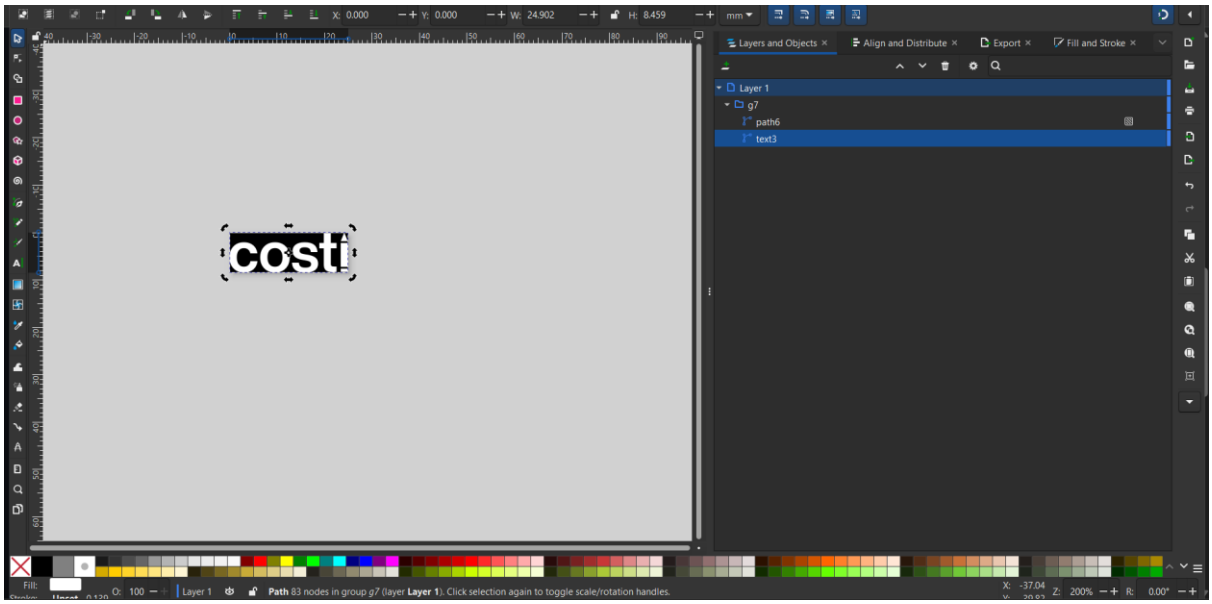
costi

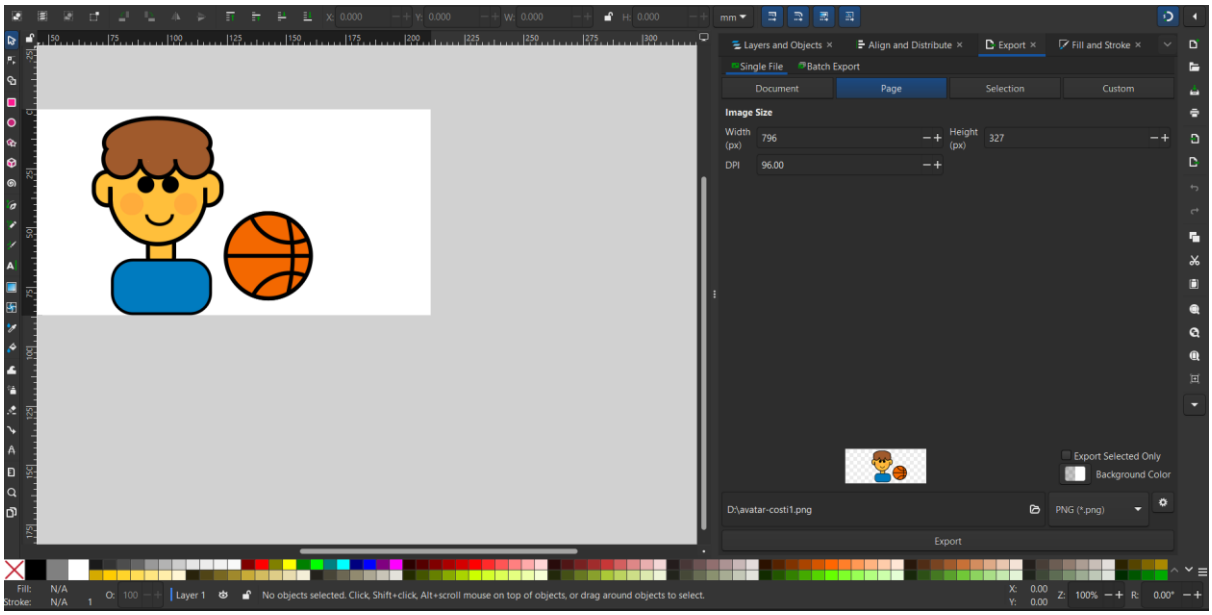
Pagina de pornire Despre mine

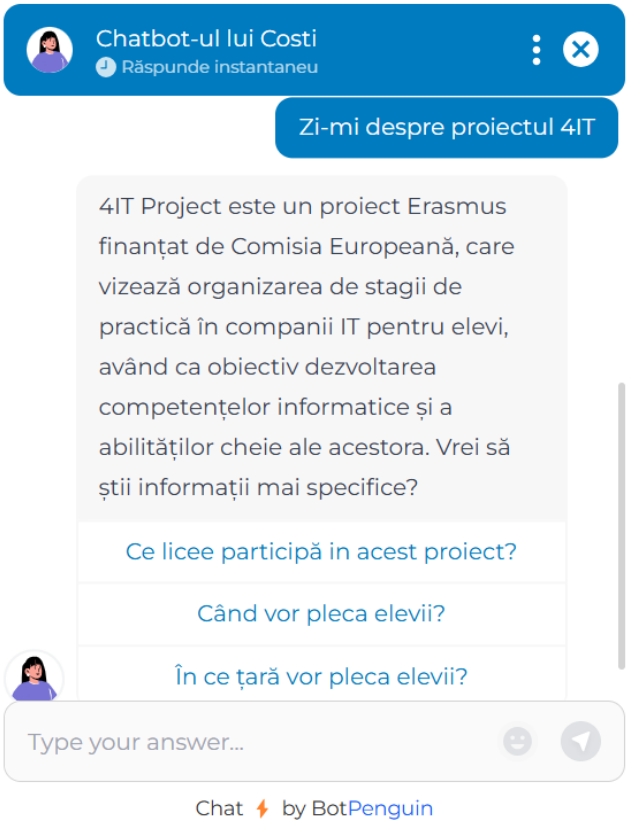


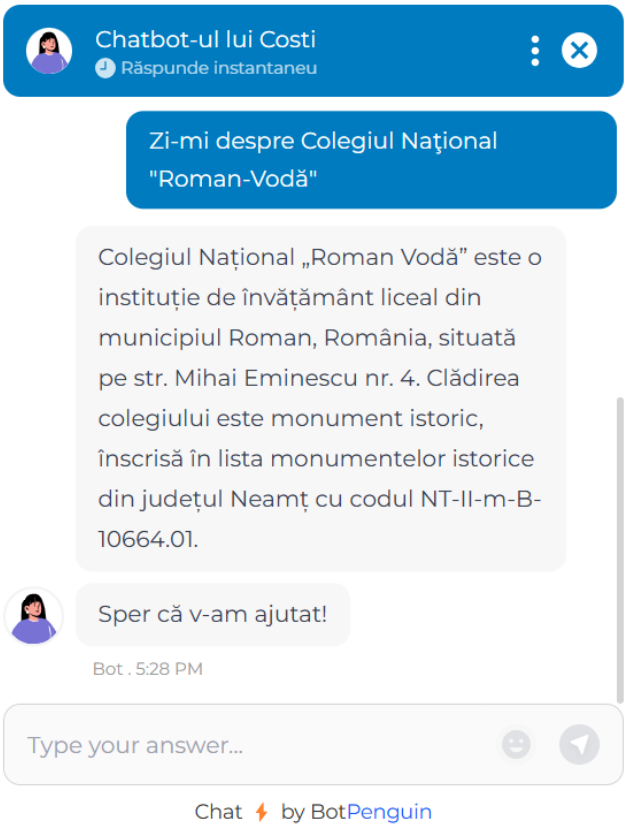


Colegiul Național "Roman Vodă", Roman, Neamț

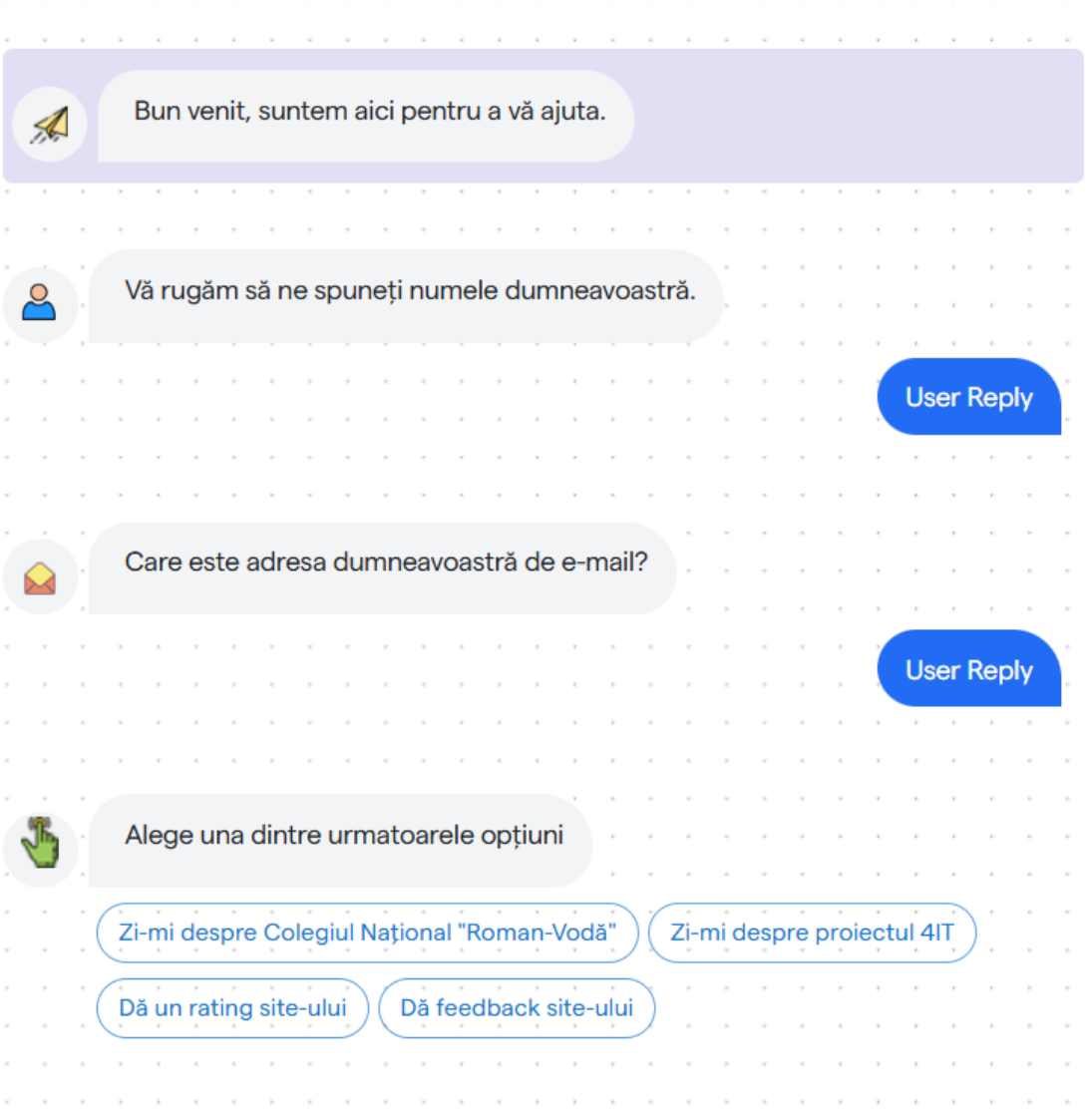

costi
Layers and Objects
Align and Distribute
Export
Fill and Stroke
Layer 1
g7
path6
text3


Layers and Objects
Align and Distribute
Export
Fill and Stroke
Single File
Batch Export
Document
Page
Selection
Custom
Image Size
Width (px)
796
Height (px)
327
DPI
96.00
Export Selected Only
Background Color
D:\avatar-costi1.png
PNG (*.png)
Export


Chatbot-ul lui Costi
Răspunde instantaneu
Zi-mi despre proiectul 4IT
4IT Project este un proiect Erasmus finanțat de Comisia Europeană, care vizează organizarea de stagii de practică în companii IT pentru elevi, având ca obiectiv dezvoltarea competențelor informatice și a abilităților cheie ale acestora. Vrei să știi informații mai specifice?
Ce licee participă in acest proiect?
Când vor pleca elevii?
În ce țară vor pleca elevii?
Type your answer...
Chat by BotPenguin


Chatbot-ul lui Costi
Răspunde instantaneu
Zi-mi despre Colegiul Naţional "Roman-Vodă"
Colegiul Național „Roman Vodă” este o instituție de învățământ liceal din municipiul Roman, România, situată pe str. Mihai Eminescu nr. 4. Clădirea colegiului este monument istoric, înscrisă în lista monumentelor istorice din județul Neamț cu codul NT-II-m-B-10664.01.
Sper că v-am ajutat!
Bot . 5:28 PM
Type your answer...
Chat by BotPenguin

Create/Reorder Chat Flow



Create/Reorder Chat Flow

Colegiul Național „Roman Vodă” este o instituție de învățământ liceal din municipiul Roman, România, situată pe str. Mihai Eminescu nr. 4. Clădirea colegiului este monument istoric, înscrisă în lista monumentelor istorice din județul Neamț cu codul NT-II-m-B-10664.01.

Sper că v-am ajutat!

4IT Project este un proiect Erasmus finanțat de Comisia Europeană, care vizează organizarea de stagii de practică în companii IT pentru elevi, având ca obiectiv dezvoltarea competențelor informatice și a abilităților cheie ale acestora.
Vrei să știi informații mai specifice?

Ce licee participă in acest proiect?
Când vor pleca elevii?
În ce țară vor pleca elevii?

- Colegiul National “Gheorghe Vrănceanu”

Create/Reorder Chat Flow

- Colegiul Național “Gheorghe. Vrănceanu”
- Colegiul Național „Costache Negri”
- Colegiul Național “Roman Vodă” Roman
- Colegiul Național „Ferdinand I”, Bacău

Elevii vor pleca pe 13 aprilie și se vor întoarce pe 27 aprilie.

Elevii vor pleca în Cipru.

Ce rating a-ți da site-ului?

User Reply

Ce feedback aveți pentru acest site?

User Reply

Teme realizate

Pointeri

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

int main() {
  int age;
  int *aPtr;
  age = 5;
  aPtr = &age; // & is the address-of operator, it returns the address of age;
               // an address holds a hexadecimal(base 16) value e.g 0xb7acdff814
  cout << "age: " << age << endl;          // result: 5
  cout << "aPtr: " << aPtr << endl;        // result: address of age, 5
  cout << "Dereferencing aPtr: " << *aPtr; // result: value of age, 5
  return 0;
}
```

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

int main() {
  int a;
  int *ptr;
  ptr = &a;
  // we do a for loop and the adresses increase by 4 bytes
each time(int is 4 bytes)
  for(int i = 0; i < 5; i++) {
    cout << ptr << endl;
    ptr++;
  }

  return 0;
}
```

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

int main() {
  int *dynamicArray; // pointer to an integer, which is
going to be the dynamic
                     // array
  dynamicArray = new int[50]; // dynamically allocate memory
for 50 integers
  for (int i = 0; i < 50;
       i++) { // use the for loop to fill the dynamic array
with 50 integers
    dynamicArray[i] = i + 1;
  }
  for (int i = 0; i < 50; i++) { // display the contents of
the dynamic array
    cout << dynamicArray[i] << " - ";
  }
  cout << endl;
  delete dynamicArray; // release the dynamically allocated
space back to the
                       // operating system
  return 0;
}
```

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

int main() {
  // Collect name and telephone number
  // metadata = store data for data
  string *names = NULL, *phones = NULL, *tempArrayNames =
NULL, *tempArrayPhones = NULL;
  int size = 0, counter = 0;
  cout << "How many students data do you want to enter?:";
  cin >> size;
  names = new string[size];
  phones = new string[size];
  do
  {
    cout << "Enter student name" << counter << " :";
    cin >> names[counter];
    cout << "Enter phone number for " << names[counter] << "
" << counter << " :";
    cin >> phones[counter];
    counter++;
    if(counter >= size) {
      cout << "Out of space! Please enter how many more
numbers would you like to enter(0 to exit)";
      int additionalSize = 0;
      cin >> additionalSize;
      if(additionalSize > 0) {
        size += additionalSize;
        tempArrayNames = new string[size];
        tempArrayPhones = new string[size];
        for (int i = 0; i < counter; i++) {
          tempArrayNames[i] = names[i];
          tempArrayPhones[i] = phones[i];
        }
        delete []names;
        delete []phones;
        names = tempArrayNames;
        phones = tempArrayPhones;
      }
    }
  } while (counter<size);
```

```
  cout<<"Now printing the numbers you inputed: "<<endl;
  for (int i = 0; i < counter; i++) {
    cout << "Name: "<< names[i] << endl;
    cout << "Phone: "<< phones[i] << endl;
  }
  delete []names;
  delete []phones;

  return 0;
}
```

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

int main() {
  /* write a program where you will use a dynamic array to
save character
  the user will give you integers that range from 65-90 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
  */

  char *dynamicArray = NULL, *tempArray = NULL;
  int arraySize = 0;
  cout << "How many numbers would you like to enter?: ";
  cin >> arraySize;
  dynamicArray = new char[arraySize];
  int counter = 0;
  do {
    cout << "Enter number " << counter << " -> ";
    int input = 0;
    cin >> input;
    if (input < 65 || input > 90 && input != 0) {
      cout << "Input needs to be between 65 and 90";
      continue;
    }
    cout << "Number " << input << " saved!" << endl;
    dynamicArray[counter] = (char)input;

    counter++;
```

```
if (counter >= arraySize) {
      cout << "Out of space! Please enter how many more
numbers would you like "
              "to enter(0 to exit)";
      int additionalSize = 0;
      cin >> additionalSize;
      if (additionalSize > 0) {
        arraySize += additionalSize;
        tempArray = new char[arraySize];
        for (int i = 0; i < counter; i++) {
          tempArray[i] = dynamicArray[i];
        }
        delete[] dynamicArray;
        dynamicArray = tempArray;
      }
    }
  } while (counter < arraySize);

  cout << "Now printing the numbers you inputed: " << endl;
  for (int i = 0; i < counter; i++) {
    cout << dynamicArray[i] << endl;
  }
  delete[] dynamicArray;

  return 0;
}
```

```cpp
// Write a program where you pass a pointer to a function , the pointer will
// point to an array and do a small search

#include <iostream>
using namespace std;

void searchNumbers(int *numbersArray, int arraySize, int searchValue) {
  for (int i = 0; i < arraySize; i++) {
    if (numbersArray[i] == searchValue) {
      cout << "Found number " << searchValue << "!" << endl;
      return;
    }
  }
  cout << "Number not found!" << endl;
}

int main() {
  int arraySize = 0;
  cout << "How many numbers would you like to enter?: ";
  cin >> arraySize;
  int *numbersArray = new int[arraySize];
  for (int i = 0; i < arraySize; i++) {
    cout << "Enter number " << i + 1 << ": ";
    cin >> numbersArray[i];
  }
  cout << "Enter a number to search: ";
  int numberToSearch = 0;
  cin >> numberToSearch;
  searchNumbers(numbersArray, arraySize, numberToSearch);

  cout << "You entered: ";
  for (int i = 0; i < arraySize; i++) {
    cout << numbersArray[i] << " ";
  }
  cout << endl;

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

Argumente în linia de comandă

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

int main(int argc, char *argv[]) { // argc -> argument count
  // first argument is the exe file
  cout << "You have entered" << argc << "arguments: " <<
endl;
  int i = 1;
  // iterate through arguments
  while (i < argc) {
    cout << "Argument " << i + 1 << ": " << argv[i] << endl;
    if (strcmp(argv[i], "/h") == 0) {
      // write code to show help
    }
    if (strcmp(argv[i], "/s") == 0) {
      // write code to implement /s option
    }
    if (strcmp(argv[i], "/t") == 0) {
      // write code to show /t option
    }

    i++;
  }
  // /h /s /t
  return 0;
}
```

Programarea orientată obiect

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

class Car {
 private:
  string brand;
  string model;
  string plateNumber;
  string ownerName;

 public:
  Car() {
    brand = "EMPTY";
    model = "EMPTY";
    plateNumber = "EMPTY";
    ownerName = "EMPTY";
  }
  void setBrand(string brandValue) { brand = brandValue; }
  void setModel(string modelValue) { model = modelValue; }
  void setPlateNumber(string plateNumberValue) { plateNumber = plateNumberValue; }
  void setOwnerName(string ownerNameValue) { ownerName = ownerNameValue; }
  string getBrand() { return brand; }
  string getModel() { return model; }
  string getPlateNumber() { return plateNumber; }
  string getOwnerName() { return ownerName; }
};

void resizeArray(Car *&carsArray, int &index) {
  Car *tempArray;
  tempArray = new Car[index + 1];
  for (int i = 0; i < index; i++) tempArray[i] = carsArray[i];
  delete[] carsArray;
  carsArray = tempArray;
}
```

```
void readCarData(Car *&carsArray, int &index) {
  resizeArray(carsArray, index);
  string tempBrand = "", tempModel = "", tempPlateNumber = "", tempOwnerName = "";
  cout << "Input brand for car " << index + 1 << " :";
  cin >> tempBrand;
  carsArray[index].setBrand(tempBrand);

  cout << "Input model for car " << index + 1 << " :";
  cin >> tempModel;
  carsArray[index].setModel(tempModel);

  cout << "Input plate number for car " << index + 1 << " :";
  cin.ignore();
  getline(cin, tempPlateNumber);
  carsArray[index].setPlateNumber(tempPlateNumber);

  cout << "Input owner name for car " << index + 1 << " :";
  getline(cin, tempOwnerName);
  carsArray[index].setOwnerName(tempOwnerName);
  index++;
}

void displayCar(Car carsArray[], int index) {
  cout << "Brand for car " << index + 1 << " :" << carsArray[index].getBrand() << '\n';
  cout << "Model for car " << index + 1 << " :" << carsArray[index].getModel() << '\n';
  cout << "Plate number for car " << index + 1 << " :" << carsArray[index].getPlateNumber() << '\n';
  cout << "Owner name for car " << index + 1 << " :" << carsArray[index].getOwnerName() << '\n';
  cout << endl;
}

void displayAllCars(Car carsArray[], int carCount) {
  if (carCount == 0) {
    cout << "No cars to display.\n";
    return;
  }
  cout << "Displaying all cars:\n";
  for (int i = 0; i < carCount; i++) {
    displayCar(carsArray, i);
  }
}

void searchByBrand(Car carsArray[], int carsCount, string brand) {
  int found = 0;
  for (int i = 0; i < carsCount; i++) {
    if (brand == carsArray[i].getBrand()) {
      displayCar(carsArray, i);
      found = 1;
    }
  }
  if (!found) {
    cout << "No car found with brand: " << brand << '\n';
  }
}
```

```
void searchByModel(Car carsArray[], int carsCount, string model) {
  int found = 0;
  for (int i = 0; i < carsCount; i++) {
    if (model == carsArray[i].getModel()) {
      displayCar(carsArray, i);
      found = 1;
    }
  }
  if (!found) {
    cout << "No car found with model: " << model << '\n';
  }
}

void searchByPlateNumber(Car carsArray[], int carsCount, string plateNumber) {
  int found = 0;
  for (int i = 0; i < carsCount; i++) {
    if (plateNumber == carsArray[i].getPlateNumber()) {
      displayCar(carsArray, i);
      found = 1;
    }
  }
  if (!found) {
    cout << "No car found with plate number: " << plateNumber << '\n';
  }
}

void searchByOwnerName(Car carsArray[], int carsCount, string ownerName) {
  int found = 0;
  for (int i = 0; i < carsCount; i++) {
    if (ownerName == carsArray[i].getOwnerName()) {
      displayCar(carsArray, i);
      found = 1;
    }
  }
  if (!found) {
    cout << "No car found with owner name: " << ownerName << '\n';
  }
}

int findOwnerName(Car carsArray[], int carsCount, string ownerName) {
  for (int i = 0; i < carsCount; i++) {
    if (ownerName == carsArray[i].getOwnerName()) {
      return i;
    }
  }
  return -1;
}
```

```
void editData(Car carsArray[], int carsCount, int index) {
  string newBrand, newModel, newPlateNumber, newOwnerName;
  cout << "Current data:\n";
  displayCar(carsArray, index);
  cout << "Enter new brand: ";
  cin >> newBrand;
  carsArray[index].setBrand(newBrand);

  cout << "Enter new model: ";
  cin >> newModel;
  carsArray[index].setModel(newModel);

  cout << "Enter new plate number: ";
  cin.ignore();
  getline(cin, newPlateNumber);
  carsArray[index].setPlateNumber(newPlateNumber);

  cout << "Enter new owner name: ";
  getline(cin, newOwnerName);
  carsArray[index].setOwnerName(newOwnerName);
}

int main() {
  /* 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
  o
  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
  */
  /* Break the above program into functions create a menu of optoins with the
   * following info: 1. enter data 2.display data 3.search by brand 4. search by
   * model 5. search by plate number 6. search by owner 0. to exit after you do
   * that you will convert the program to use dynamic(growing) arrays*/
  Car *carsArray = nullptr;
  int carsCount = 0;
  carsArray = new Car[carsCount + 1];
  int input = 0;
  do {
    cout << "======================================== Car Management "
            "System ========================================\n"
         << "1. Enter Data\n"
         << "2. Display Data\n"
         << "3. Search by Brand\n"
         << "4. Search by Model\n"
         << "5. Search by Plate Number\n"
         << "6. Search by Owner Name\n"
         << "0. Exit\n"
         << "Enter your choice:\n";
    cin >> input;
```

```
switch (input) {
    case 0:
      cout << "Goodbye";
      break;
    case 1:
      readCarData(carsArray, carsCount);
      break;
    case 2:
      displayAllCars(carsArray, carsCount);
      break;
    case 3: {
      string brand;
      cout << "Enter brand to search: ";
      cin >> brand;
      searchByBrand(carsArray, carsCount, brand);
      break;
    }
    case 4: {
      string model;
      cout << "Enter model to search: ";
      cin >> model;
      searchByModel(carsArray, carsCount, model);
      break;
    }
    case 5: {
      string plateNumber;
      cout << "Enter plate number to search: ";
      cin.ignore();
      getline(cin, plateNumber);
      searchByPlateNumber(carsArray, carsCount, plateNumber);
      break;
    }
    case 6: {
      string ownerName;
      cout << "Enter owner name to search: ";
      cin.ignore();
      getline(cin, ownerName);
      searchByOwnerName(carsArray, carsCount, ownerName);
      break;
    }
    case 7: {
      string ownerNameToReplace;
      cout << "Enter owner name to replace data";
      cin.ignore();
      getline(cin, ownerNameToReplace);
      int index = findOwnerName(carsArray, carsCount, ownerNameToReplace);
      if (index == -1) {
        cout << "Owner name not found.\n";
        break;
      }
      cout << "Owner name found at index: " << index << '\n';
      editData(carsArray, carsCount, index);
    }
    default:
      cout << "Invalid option! Please try again.\n";
      break;
    }
  } while (input != 0);
  delete[] carsArray;
  return 0;
}
```

```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 brandValue) { brand = brandValue; }
  void setType(string typeValue) { type = typeValue; }
  void setPrice(float priceValue) { price = priceValue; }
  void setStock(int stockValue) { stock = stockValue; }
  void setBarcode(string barcodeValue) { barcode = barcodeValue;
} string getBrand() { return brand; }
  string getType() { return type; }
  float getPrice() { return price; }
  int getStock() { return stock; }
  string getBarcode() { return barcode; }
};

Product *resizeArray(Product *productsArray, int &capacity) {
  int newCapacity = capacity * 2;
  Product *tempArray = new Product[newCapacity];
  for (int i = 0; i < capacity; i++) {
    tempArray[i] = productsArray[i];
  }
  delete[] productsArray;
  capacity = newCapacity;
  return tempArray;
}
```

```
void searchByBrand(Product productsArray[], int productCount, string brand) {
  int found = 0;
  for (int i = 0; i < productCount; i++) {
    if (brand == productsArray[i].getBrand()) {
      displayProduct(productsArray, i);
      found = 1;
    }
  }
  if (!found) {
    cout << "No product found with brand: " << brand << '\n';
  }
}

void searchByType(Product productsArray[], int productCount, string type) {
  int found = 0;
  for (int i = 0; i < productCount; i++) {
    if (type == productsArray[i].getType()) {
      displayProduct(productsArray, i);
      found = 1;
    }
  }
  if (!found) {
    cout << "No product found with type: " << type << '\n';
  }
}

void searchByBarcode(Product productsArray[], int productCount,
                     string barcode) {
  int found = 0;
  for (int i = 0; i < productCount; i++) {
    if (barcode == productsArray[i].getBarcode()) {
      displayProduct(productsArray, i);
      found = 1;
      productsArray[i].setStock(productsArray[i].getStock() - 1);
      cout << "Stock updated. Remaining stock: " << productsArray[i].getStock()
           << '\n';
    }
  }
  if (!found) {
    cout << "No product found with barcode: " << barcode << '\n';
  }
}
```

```
int main() {
  /*
  -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
  Your program must be able to search for products by name
and by Type
  */
  int capacity = 5;
  int productsCount = 0;
  Product *productsArray = new Product[productsCount];
  int input = 0;
  do {
    cout << "1. Add Product\n"
         << "2. Display All Products\n"
         << "3. Search by Brand\n"
         << "4. Search by Type\n"
         << "5. Search by Barcode\n"
         << "6. Exit\n";
    cin >> input;
    switch (input) {
    case 1:
      productsArray = addProduct(productsArray,
productsCount, capacity);
      break;
    case 2:
      displayAllProducts(productsArray, productsCount);
      break;
```

```
 case 3: {
      string brand;
      cout << "Enter brand to search: ";
      cin >> brand;
      searchByBrand(productsArray, productsCount, brand);
      break;
    }
    case 4: {
      string type;
      cout << "Enter type to search: ";
      cin >> type;
      searchByType(productsArray, productsCount, type);
      break;
    }
    case 5: {
      string barcode;
      cout << "Enter barcode to search: ";
      cin >> barcode;
      searchByBarcode(productsArray, productsCount,
barcode);
      break;
    }
    case 6:
      cout << "Goodbye!";
      break;
    default:
      cout << "Invalid option! Please try again.\n";
      break;
    }
  } while (input != 6);

  return 0;
}
```

```cpp
#include <iostream>
using namespace std;
/*
This example is showing you how to dynamically grow an array during run time.
It uses pointers to create the array.
*/

int main() {
  int ARRAYSIZE = 0;    // the ARRAYSIZE is how many numbers the user wants to input. This
                        // might change if the user wishes to change it!
  int *a;               // a is a pointer to an integer. This is going to be our main array to
                        // save integers
  int *temp;            // temp is a pointer to an integer and is going to be a temporary
                        // array. We need it in order to be able to GROW our array.
  int n = 0, tmpN = 0;  // n is the counter to count the iterations. tmpN is a
                        // number we read from the user in order to see if they
                        // need to input more numbers than the initial decision.

  //--- Read into the array
  cout << "How many numbers would you like to enter?:";  // ask the user and
                                                         // create an array
                                                         // according to the
                                                         // user's demand.
  cin >> ARRAYSIZE;                                      // read the size of the array from the user
  a = new int[ARRAYSIZE];  // dynamically create an array with the number the
                           // user gave to us.

  do {
    cout << "Enter number " << n << " ->";  // prompt the user to enter the numbers
    cin >> a[n];                            // save the number we just read into our dynamic array
    cout << "Number saved!" << endl;
    n++;                  // after saving the number we have to increase the counter
    if (n >= ARRAYSIZE) {  // if the new counter is equal to the ARRAYSIZE we
                           // need to GROW THE ARRAY
      cout << "Out of Space!!!! would you like to enter more? Please enter how "
              "many(0 to Exit): ";      // LET THE USER KNOW THAT THE ARRAY IS OUT OF
                                        // SPACE AND ASK HOW MANY MORE NUMBERS THEY
                                        // NEED TO INPUT
      cin >> tmpN;                      // read how many more number the user needs to input
      if (tmpN > 0) {                   // ERROR CHECKING: If the number the user gave us is not
                                        // zero OR below zero we grow the array
        ARRAYSIZE = ARRAYSIZE + tmpN;   // Add to the previous size the added size the user gave us
        temp = new int[ARRAYSIZE];      // create new temporary bigger array. We need
                                        // it to copy the contents of the previous
                                        // array so we don't lose them
        for (int i = 0; i < n; i++) {   // We use the for loop to copy the contents from the
                                        // previous array into the temporary array(which has a
                                        // bigger size
          temp[i] = a[i];               // copy values to new array.
        }
        delete[] a;  // free old array memory. ALSO SEE LAST COMMENT UNDER MAIN
        a = temp;    // now a points to new array.
      }
    }

  } while (n < ARRAYSIZE);
```

```
 cout << "Now printing the numbers you inputed" << endl;

  for (int i = 0; i < ARRAYSIZE; i++) {
    cout << a[i] << endl;
  }
  delete[] a;  // free the memory of a

  return 0;
}

/*Here you must not delete the temp because we set 'a' to equal temp
        Remember that pointers are variables which actually save memory
   addresses. in this program, at the part where we grow our array we set
   a=temp. Therefore a and temp point to the same part of memory. Because of
   that you need to delete only pointer a, and the memory will be deleted for
   temp as well.
    *
```

```
class A {
public:
  int a, b;
  // constructor
  A(int _a, int _b) {
    a = _a;
    b = _b;
  }
  // functie membra: returneaza suma dintre a si b
  int Suma() { return a + b; }
};
```

```cpp
#include <iostream>

class Stiva {
private:
  int *st;
  int *maxSt;
  int n;
  int capacity;

public:
  Stiva() {
    capacity = 1000;
    st = new int[capacity];
    maxSt = new int[capacity];
    n = 0;
  }

  ~Stiva() {
    delete[] st;
    delete[] maxSt;
  }

  void Push(int x) {
    if (n == capacity) {
      // redimensionare pentru ambele stive
      int *temp = new int[capacity * 2];
      int *tempMax = new int[capacity * 2];
      for (int i = 0; i < n; i++) {
        temp[i] = st[i];
        tempMax[i] = maxSt[i];
      }
      delete[] st;
      delete[] maxSt;
      st = temp;
      maxSt = tempMax;
      capacity *= 2;
    }

    st[n] = x;
    if (n == 0)
      maxSt[n] = x;
    else
      maxSt[n] = std::max(x, maxSt[n - 1]);

    n++;
  }
```

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

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

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

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

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

class student {
  // these are private by defualt
  string fullName;
  string lastName;
  int age;

public:
  void setAge(int a) { age = a; }

private:
};
int main() {
  student oneStudent;
  oneStudent.setAge(18);
  return 0;
}
```

```
class Tablou {
public:
  int *t; // tabloul unidimensional
  int n;  // dimensiunea fizica si logica tabloului
  // constructorul
  Tablou(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;
  }
  int Max() {
    int max = t[0];
    for (int i = 1; i < n; i++) {
      if (t[i] > max)
        max = t[i];
    }
    return max;
  }
  int Cauta(int val) {
    for (int i = 0; i < n; i++) {
      if (val == t[i])
        return i;
    }
    return -1;
  }
};
```

Arduino

```
/*
  AnalogReadSerial

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

  This example code is in the public domain.


 https://www.arduino.cc/en/Tutorial/BuiltInExamples/AnalogRe
adSerial
*/

// 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
}
```

```
/*
  DigitalReadSerial

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

  This example code is in the public domain.


 https://www.arduino.cc/en/Tutorial/BuiltInExamples/DigitalR
eadSerial
*/

// 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
}
```

```
/*
  Blink

  Turns an LED on for one second, then off for one second,
repeatedly.

  Most Arduinos have an on-board LED you can control. On the
UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6.
LED_BUILTIN is set to
  the correct LED pin independent of which board is used.
  If you want to know what pin the on-board LED is connected
to on your Arduino
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

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

// 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
}
```

Dezvoltare durabilă și antreprenoriat

Crearea unor produse sustenabile din materiale reciclate

Avantajele utilizării materialelor reciclate:

1. **Reducerea poluării** – reduce cantitatea de deșeuri care ajunge la gropile de gunoi și în oceane.
2. **Economisirea resurselor naturale** – scade necesitatea extragerii de materii prime.
3. **Reducerea consumului de energie** – Ex: Producerea aluminiului reciclat consumă cu 95% mai puțină energie decât extragerea și rafinarea aluminiului nou.

Simboluri ECO ce se regăsesc pe ambalaje

Bucla Mobius

Simbolul internațional al reciclării - indica un produs reciclabil - Dacă are un număr în interior, indică procentul de material reciclat folosit



Simbolul „Punctul verde”

Simbolul internațional care indica faptul că producătorul a avut o contribuție financiara la o schema de recuperare și reciclare a ambalajelor.



Simbol pentru reciclare separată

Indică faptul că acest tip de deșeu se colectează separat, și este interzis aruncarea acestuia cu alte tipuri de deșeuri.



Exemple de produse sustenabile din materiale reciclate:

- **Îmbrăcăminte și încălțăminte ecologică** - este folosit poliesterul reciclat și plasticul reciclat din oceane
- **Ambalaje ecologice** - sunt folosite ambalajele din plastic cu carton reciclat sau bio-plastic



