Tema 1

NICOLETA RADU, OOP C++

# Problema 1

// Programare OOP.cpp : This file contains the 'main' function. Program execution begins and ends there.

//

#include <iostream>

#include <fstream>

#include <vector>

#include<iomanip>

using namespace std;

struct fly

{

int num;

int palindrom;

int divizor;

};

template <typename T>

int count\_elements(istream& is) // dimensiune vector

{

istream\_iterator<T> start(is), end;

return distance(start, end);

}

//void findSize(int& count)

//{

// ifstream inFile("numere.txt");

// if (!inFile)

// {

// cout << "ERROR: couldn't open file " << endl;

// }

// int aux{};

// while (inFile >> aux)

// {

// inFile >> aux;

// cout << "AUX = " << aux << endl;

// count = count + 1;

// }

//

// cout << "Count = " << count << endl;

// inFile.close();

//}

void populateArray(fly yourArray[], int arraySize) // citire vector

{

ifstream inFile("numere.txt");

if (!inFile)

{

cout << "ERROR: couldn't open file " << endl;

}

fly m;

while (!inFile.eof())

{

for (size\_t i = 0; i < arraySize; i++)

{

inFile >> m.num;

yourArray[i].num = m.num;

}

}

inFile.close();

}

void printArray1(fly yourArray[], int size) // afiseaza vectorul

{

cout << "Your array: " << endl;

cout << "[ ";

for (size\_t i = 0; i < size; i++)

{

cout << yourArray[i].num << " ";

}

cout << "]" << endl;

}

void printArray2(fly yourArray[], int size) // afiseaza divizorii proprii

{

cout << "Your array: " << endl;

cout << "[ ";

for (size\_t i = 0; i < size; i++)

{

cout << yourArray[i].divizor << " ";

}

cout << "]" << endl;

}

int reverseNum(int num) // aflam palindrom

{

int copyNum = num;

int reverseNum{0};

while (copyNum != 0)

{

reverseNum = reverseNum \* 10 + (copyNum % 10);

copyNum = copyNum / 10;

}

return reverseNum;

}

int lowestPalindrome(int num) // aflam cel mai mic palindrom mai mare decat numarul

{

bool notFound = true;

int i{ 1 };

int palindrome{};

while (notFound)

{

palindrome = num + i;

if (reverseNum(palindrome) == palindrome)

{

notFound = false;

return palindrome;

}

i++;

}

}

bool isPrim(int num) // aflam numerele prime

{

bool prim = true;

if (num == 2 || num == 0)

{

return prim;

}

for (size\_t i = 2; i < sqrt(num); i++)

{

if (num % i != 0)

{

prim = true;

}

else

{

prim = false;

return prim;

}

}

return prim;

}

int divizorPropriu(int num) // aflam divizorul propriu

{

int cmmdpc{};

if (isPrim(num) == false)

{

for (size\_t i = 2; i < sqrt(num); i++)

{

if (num % i == 0)

{

cmmdpc = num / i;

return cmmdpc;

}

}

}

else

{

return 1;

}

}

void populateArray2(fly yourArray[], int size) // popularea campurilor divizor si palindrome

{

ifstream inFile("numere.txt");

if (!inFile)

{

cout << "ERROR: couldn't open file " << endl;

}

for (size\_t i = 0; i < size; i++)

{

yourArray[i].divizor = divizorPropriu(yourArray[i].num);

yourArray[i].palindrom = lowestPalindrome(yourArray[i].num);

}

inFile.close();

}

void sortArray(fly yourAray[],int size) // sortare prin selectie

{

int aux{};

for (size\_t i = 0; i < size - 1; i++)

{

for (size\_t j = i + 1; j < size; j++)

{

if (yourAray[j].divizor < yourAray[i].divizor)

{

aux = yourAray[i].divizor;

yourAray[i].divizor = yourAray[j].divizor;

yourAray[j].divizor = aux;

}

}

}

}

int main()

{

ifstream inFile;

inFile.open("numere.txt");

if (!inFile)

{

cout << "Unable to open file";

}

fly\* myVector;

int myVectorSize{};

myVectorSize = count\_elements<int>(inFile);

myVector = new fly[myVectorSize];

populateArray(myVector, myVectorSize);

cout << "Print Array: " << endl;

printArray1(myVector, myVectorSize);

populateArray2(myVector, myVectorSize);

sortArray(myVector, myVectorSize);

cout << "Print Array divizor: " << endl;

printArray2(myVector, myVectorSize);

cout << endl;

return 0;

}

# Problema 3

// Programare OOP.cpp : This file contains the 'main' function. Program execution begins and ends there.

//

#include <iostream>

#include <fstream>

#include<iomanip>

using namespace std;

struct terminator

{

int row;

int column;

int\*\* address;

};

template <typename T>

int count\_elements(istream& is) // dimensiune vector

{

istream\_iterator<T> start(is), end;

return distance(start, end);

}

void populateMatrix(int\*\* yourMatrix, int rows, int cols)

{

ifstream inFile;

inFile.open("numere.txt");

if (!inFile)

{

cout << "ERROR: Could not open file." << endl;

}

inFile >> rows;

inFile >> cols;

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

yourMatrix[i] = new int[cols];

}

}

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

inFile >> yourMatrix[i][j];

}

}

}

void printMatrix(int\*\* yourMatrix, int rows, int cols)

{

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

cout << yourMatrix[i][j] << " ";

}

cout << endl;

}

cout << endl;

}

void sumaMatrici(int\*\* yourMatrix\_1, int\*\* yourMatrix\_2, int\*\* yourMatrix\_3, int rows, int cols)

{

if (rows == cols)

{

for (size\_t i = 0; i < rows; i++)

{

for (size\_t j = 0; j < cols; j++)

{

yourMatrix\_3[i][j] = yourMatrix\_1[i][j] + yourMatrix\_2[i][j];

}

}

}

else

{

cout << "ERROR: couldn't sum matrices. Rows and columns are not equal." << endl;

}

}

void produsMatrici(int\*\* yourMatrix\_1, int\*\* yourMatrix\_2, int\*\* yourMatrix\_3, int rows1, int cols1, int rows2, int cols2)

{

if (rows2 != cols1)

{

cout << "The number of columns in Matrix-1 must be equal to the number of rows in Matrix-2" << endl;

}

else

{

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

yourMatrix\_3[i][j] = 0;

for (int k = 0; k < rows2; k++) {

yourMatrix\_3[i][j] += yourMatrix\_1[i][k] \* yourMatrix\_2[k][j];

}

cout << yourMatrix\_3[i][j] << "\t";

}

cout << endl;

}

}

}

int main()

{

terminator myMatrix\_1;

terminator myMatrix\_2;

terminator myMatrix\_3;

terminator myMatrix\_3\_size;

terminator myMatrix\_2\_size;

terminator myMatrix\_1\_size;

ifstream inFile;

inFile.open("numere.txt");

if (!inFile)

{

cout << "ERROR: Could not open file." << endl;

}

inFile >> myMatrix\_1\_size.row;

inFile >> myMatrix\_1\_size.column;

myMatrix\_1.address = new int\* [myMatrix\_1\_size.row];

for (size\_t i = 0; i < myMatrix\_1\_size.row; i++)

{

myMatrix\_1.address[i] = new int[myMatrix\_1\_size.column];

}

for (size\_t i = 0; i < myMatrix\_1\_size.row; i++)

{

for (size\_t j = 0; j < myMatrix\_1\_size.column; j++)

{

inFile >> myMatrix\_1.address[i][j];

}

}

inFile >> myMatrix\_2\_size.row;

inFile >> myMatrix\_2\_size.column;

myMatrix\_2.address = new int\* [myMatrix\_2\_size.row];

for (size\_t i = 0; i < myMatrix\_2\_size.row; i++)

{

myMatrix\_2.address[i] = new int[myMatrix\_2\_size.column];

}

for (size\_t i = 0; i < myMatrix\_2\_size.row; i++)

{

for (size\_t j = 0; j < myMatrix\_2\_size.column; j++)

{

inFile >> myMatrix\_2.address[i][j];

}

}

myMatrix\_3\_size.row = myMatrix\_2\_size.row;

myMatrix\_3\_size.column = myMatrix\_1\_size.column;

myMatrix\_3.address = new int\*[myMatrix\_3\_size.row];

for (size\_t i = 0; i < myMatrix\_3\_size.row; i++)

{

myMatrix\_3.address[i] = new int[myMatrix\_3\_size.column];

}

printMatrix(myMatrix\_1.address, myMatrix\_1\_size.row, myMatrix\_1\_size.column);

printMatrix(myMatrix\_2.address, myMatrix\_2\_size.row, myMatrix\_2\_size.column);

produsMatrici(myMatrix\_1.address, myMatrix\_2.address, myMatrix\_3.address,myMatrix\_1\_size.row, myMatrix\_1\_size.column, myMatrix\_2\_size.row, myMatrix\_2\_size.column);

printMatrix(myMatrix\_3.address, myMatrix\_3\_size.row, myMatrix\_3\_size.column);

return 0;

}

# Problema 4

// Programare OOP.cpp : This file contains the 'main' function. Program execution begins and ends there.

//

#include <iostream>

#include <fstream>

#include<iomanip>

using namespace std;

struct terminator

{

long long num;

int sumaCifre;

};

template <typename T>

int count\_elements(istream& is)

{

istream\_iterator<T> start(is), end;

return distance(start, end);

} // afla dimensiune fisier

void populateArrayNums(struct terminator yourArray[], int size)

{

ifstream inFile;

inFile.open("numere.txt"); // deschidere fisier

if (!inFile)

{

cout << "ERROR: couldn't open file." << endl;

}

inFile >> size;

cout << "Size: " << size << endl;

for (size\_t i = 0; i < size; i++)

{

yourArray[i].num = 0;

}

for (size\_t i = 0; i < size; i++)

{

inFile >> yourArray[i].num;

}

inFile.close();

}

void printArrayNums(struct terminator yourArray[], int size)

{

cout << endl;

cout << "[ ";

for (size\_t i = 0; i < size; i++)

{

cout << yourArray[i].num << " ";

}

cout << "]";

cout << endl;

}

void printArraySum(struct terminator yourArray[], int size)

{

cout << endl;

cout << "[ ";

for (size\_t i = 0; i < size; i++)

{

cout << yourArray[i].sumaCifre << " ";

}

cout << "]";

cout << endl;

}

int sumOfDigits(long long num)

{

int sum{};

while (num != 0)

{

sum = sum + num % 10;

num = num / 10;

}

return sum;

}

void populateArraySum(struct terminator yourArray[], int size)

{

for (size\_t i = 0; i < size; i++)

{

yourArray[i].sumaCifre = sumOfDigits(yourArray[i].num);

}

}

void selectionSort(struct terminator yourArray[], int size)

{

int aux;

for (size\_t i = 0; i < size; i++)

{

for (size\_t j = i + 1; j < size; j++)

{

if (yourArray[j].sumaCifre < yourArray[i].sumaCifre)

{

aux = yourArray[i].sumaCifre;

yourArray[i].sumaCifre = yourArray[j].sumaCifre;

yourArray[j].sumaCifre = aux;

}

}

}

}

int main()

{

ifstream inFile;

inFile.open("numere.txt"); // deschidere fisier

if (!inFile)

{

cout << "ERROR: couldn't open file." << endl;

}

struct terminator\* numere;

int dim\_numere{};

inFile >> dim\_numere;

numere = new struct terminator[dim\_numere];

populateArrayNums(numere, dim\_numere);

printArrayNums(numere, dim\_numere);

populateArraySum(numere, dim\_numere);

printArraySum(numere, dim\_numere);

selectionSort(numere, dim\_numere);

printArraySum(numere, dim\_numere);

return 0;

}