**Proiect APD**

**Dominte Mihai**

**CR3.1B**

**Introducere**

In cadrul acestui proiect am realizat un algoritm de sortare a unui sir de numere folosind metoda Bubble Sort si am studiat performanta de rulare a acestuia pe diferite esantioane de numere.

Algoritmul foloseste o functie de generare a numerelor pentru a constitui vectorul ce trebuie sortat. A fost folosita functia rand() pentru generarea de numere aleatoare.

Functia de sortare prin metoda bubble sort este cronometrata folosind libraria <chrono> rezultatul obtinut find afisat in microsecunde.

Functia de afisare este optionala si afiseaza numerele sortate din vector.

**Algoritm, Limbaj, Masina:**

-Ca metoda de sortare a fost folosit Bubble Sort, algoritmul complet fiind prezentat mai jos;

-Limbajul folosit a fost C++, codul a fost scris folosind Visual Studio 2019

-Masina pe care a fost rulat foloseste Windows 10, 32GB RAM, procesor AMD Ryzen 2700X cu 8 nuclee

#include "stdio.h"

#include <chrono>

#include <iostream>

#include <stdlib.h>

#include <time.h>

using namespace std;

using namespace std::chrono;

void numGenerator(int sir[], int dimensiune);

void bubbleSort(int sir[], int dimensiune);

void afisare(int sir[], int dimensiune);

int main() {

int dimensiune, optiune;

printf("Introduceti dimensiunea:");

scanf\_s("%d", &dimensiune);

int\* sir = (int\*)malloc(dimensiune \* sizeof(int));

numGenerator(sir, dimensiune);

printf("Generare finalizata \n");

auto start = high\_resolution\_clock::now();

bubbleSort(sir, dimensiune);

auto stop = high\_resolution\_clock::now();

auto duration = duration\_cast<microseconds>(stop - start);

cout << "Timpul de sortare: " << duration.count() << " microsecunde" << endl;

printf("Afisati numerele sortate 1/0:");

scanf\_s("%d", &optiune);

if (optiune == 1) {

afisare(sir, dimensiune);

}

return 0;

}

void numGenerator(int sir[], int dimensiune) {

srand(time(NULL));

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

sir[i] = rand() \* rand();

}

}

void bubbleSort(int sir[], int dimensiune) {

for (int i = 0; i < dimensiune - 1; i++) {

for (int j = 0; j < dimensiune - i - 1; j++) {

if (sir[j] > sir[j + 1]) {

int temp = sir[j];

sir[j] = sir[j + 1];

sir[j + 1] = temp;

}

}

}

}

void afisare(int sir[], int dimensiune) {

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

printf("%d ", sir[i]);

}

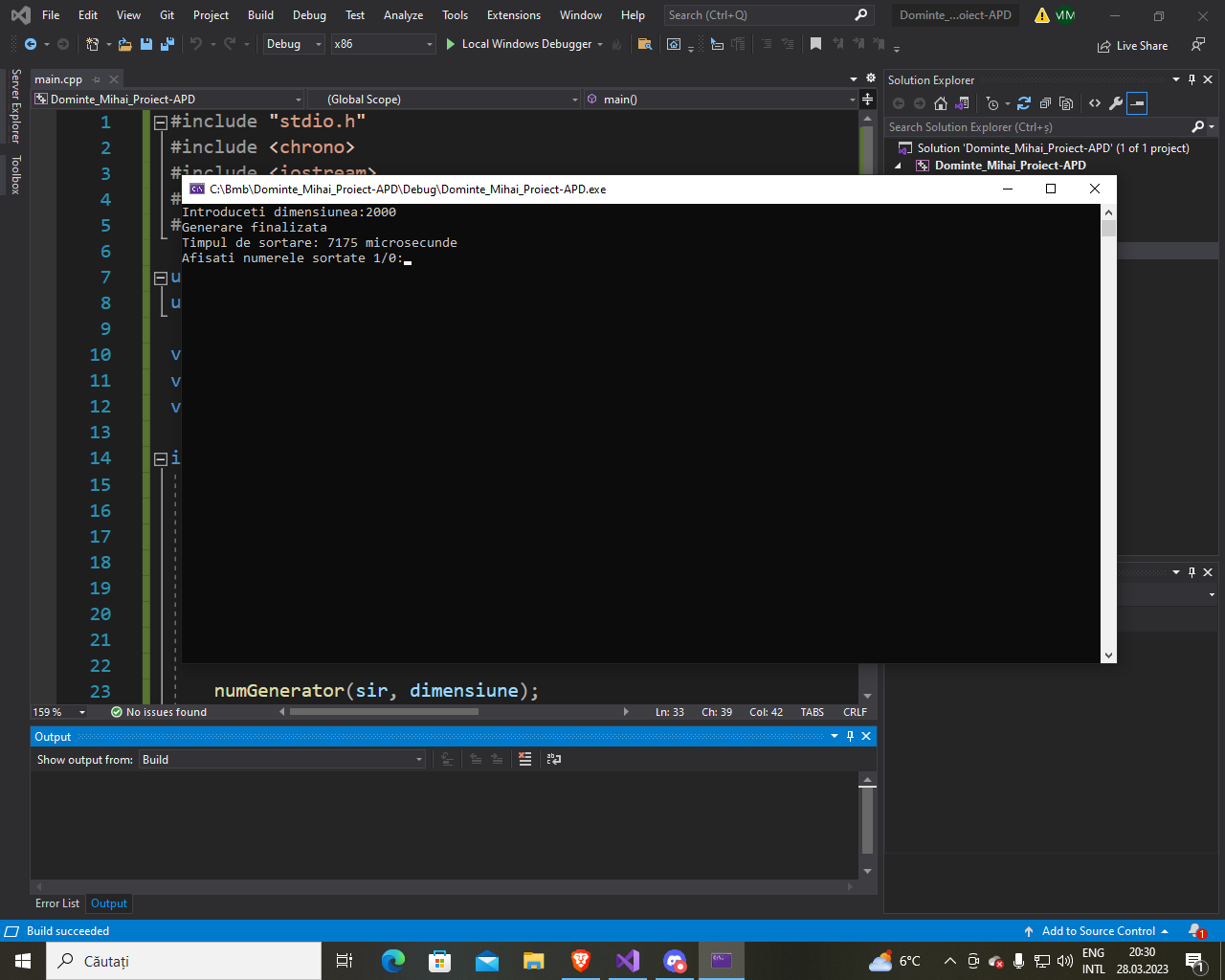
}

**Teste:**

Au fost efectuate 10 teste de sortare cu urmatoarele esantioane de numere: 500, 1000, 2000, 5000, 10000, 20000, 50000, 100000, 200000, 500000 de numere cu rezultatele obtinute prezentate in graficul de mai jos:

O imagine care conține text

Descriere generată automat



**Concluzie:**

Algoritmul Bubble Sort are un timp de rulare O(n^2) lucru care poate fi observat si prin testele de mai sus unde timpul de rulare creste de aproximativ 4 ori la fiecare dublare a dimensiunii esantionului.