using System;

using System.Diagnostics;

namespace PROJEKT

{

 class Poprawa

 {

 public static int counter = 0; //to są cztery zmienne, które będą zliczać ilość operacji

 public static int counter1 = 0;

 public static int counter2 = 0;

 public static int counter3 = 0;

 static void Main(string[] args)

 {

 int[] array = new int[268435456]; //deklaracja tablicy o 2^28 ilości elementów

 double[] results1 = new double[10]; //dodatkowa tablica dla czasów szukania

 double[] results2 = new double[10]; //dodatkowa tablica dla czasów szukania

 double[] results5 = new double[10]; //dodatkowa tablica dla czasów szukania

 double[] results6 = new double[10]; //dodatkowa tablica dla czasów szukania

 int[] results3 = new int[10]; //dodatkowa tablica dla ilości operacji

 int[] results4 = new int[10]; //dodatkowa tablica dla ilości operacji

 Random rand = new Random(); //funkcja losująca liczbę

 for (int i = 0; i < array.Length; i++) //pętla wypełniająca naszą tablicę 2^28 losowymi liczbami od 1 do 1000

 {

 int number = rand.Next(0, 1001); //ustala zakres losowych liczb na 1-1000

 array[i] = number;

 }

 bool isPresent(int[] vector, int number) //metoda szukania liniowego, z PDFa, chyba nie trzeba tłumaczyć

 {

 for (int i = 0; i < vector.Length; i++)

 {

 if (vector[i] == number)

 {

 return true;

 }

 }

 return false;

 }

 //empiryczna ocena z pomiarem czasu -średnia- ma być z 10 wyników więc pętla

 for (int i = 0; i < 10; i++)

 {

 long startingTimeAVG = Stopwatch.GetTimestamp(); //początek liczenia czasu

 isPresent(array, rand.Next(0, 1001)); //metoda szukania elementu losowego rand w tablicy array

 long endingTimeAVG = Stopwatch.GetTimestamp(); //koniec liczenia czasu

 long elapsedTimeAVG = endingTimeAVG - startingTimeAVG; //różnica czasu

 double elapsedSecondsAVG = elapsedTimeAVG \* (1.0 / Stopwatch.Frequency); //jakieś przekonwertowanie wyniku, bodajże do mikrosekund

 results1[i] = elapsedSecondsAVG; //wpisanie czasu z każdego obiegu pętli do tablicy

 }

 for (int i = 0; i < 10; i++) //tutaj zrobiłem sobie pętlę do wypisania tych wszystkich wyników

 Console.WriteLine("LINIOWA/SREDNIA | CZAS OPERACJI = "+results1[i]);

 //a tutaj żeby się nie bawić w excelu od razu zrobiłem liczenie średniej z wyników

 double result1 = (results1[0] + results1[1] + results1[2] + results1[3] + results1[4] + results1[5] + results1[6] + results1[7] + results1[8] + results1[9]) / 10;

 Console.WriteLine("LINIOWA/SREDNIA | SREDNI CZAS OPERACJI = "+result1);

 //empiryczna ocena z pomiarem czasu -negatywna- element z poza zakresu tablicy - bo wystarczy podać element, którego na pewno nie ma w tablicy

 for (int i = 0; i < 10; i++)

 {

 long startingTimeNEG = Stopwatch.GetTimestamp();

 isPresent(array, 1001); //metoda szukania elementu, którego nie ma - 1001

 long endingTimeNEG = Stopwatch.GetTimestamp();

 long elapsedTimeNEG = endingTimeNEG - startingTimeNEG;

 double result2 = elapsedTimeNEG \* (1.0 / Stopwatch.Frequency);

 results5[i] = result2;

 Console.WriteLine("LINIOWA/NEGATYWNA | CZAS OPERACJI = " + result2);

 }

 //a tutaj żeby się nie bawić w excelu od razu zrobiłem liczenie średniej z wyników

 double result9 = (results5[0] + results5[1] + results5[2] + results5[3] + results5[4] + results5[5] + results5[6] + results5[7] + results5[8] + results5[9]) / 10;

 Console.WriteLine("LINIOWA/NEGATYWNA | SREDNI CZAS OPERACJI = " + result9);

 int linearSearchOperations(int[] vector, int n) //zrobiłem dwie takie same metody - jedna dla średniej opcji, druga dla negatywnej

 //tutaj niestety sam nie wiem jak to działa, więc Ci za dużo nie wytłumaczę, ale ważne, że działa!

 //liczy się to za pomocą tego counter w jakiś sposób, już nie pamiętam, ale nie powinien się o to pytać, mnie nie pytał

 {

 for (int i = 0; i < vector.Length; i++)

 {

 counter++;

 if (vector[i] == n) return i;

 }

 return -1;

 }

 int linearSearchOperationsNEG(int[] vector, int n) //dla opcji negatywnej

 {

 for (int i = 0; i < vector.Length; i++)

 {

 counter1++;

 if (vector[i] == n) return i;

 }

 return -1;

 }

 for (int i = 0; i < 10; i++) //znów ma być dla 10 losowych liczb, więc taka pętelka

 {

 counter = 0;

 linearSearchOperations(array, rand.Next(0, 1001));

 results3[i] = counter;

 Console.WriteLine("LINIOWA/SREDNIA | LICZBA OPERACJI = "+results3[i]);

 }

 int result3 = results3[0] + (results3[1] + results3[2] + results3[3] + results3[4] + results3[5] + results3[6] + results3[7] + results3[8] + results3[9]) / 10; //średnia liczba operacji

 Console.WriteLine("LINIOWA/SREDINA | SREDNIA LICZBA OPERACJI = " + result3);

 linearSearchOperationsNEG(array, 1001); //tutaj dla negatywnego

 Console.WriteLine("LINIOWA/NEGATYWNA | LICZBA OPERACJI = " + counter1);

 Array.Sort(array); //sortowanie tablicy, bo jest to potrzebne dla szukania binarnego

 Console.WriteLine("\n \*\*\*\*\* \n");

 bool isPresent2(int[] Vector, int Number) //metoda szukania binarnego, to też z PDFa, nie trzeba tłumaczyć (mam nadzieję)

 {

 int Left = 0, Right = Vector.Length - 1, Middle;

 while (Left < Right)

 {

 Middle = (Left + Right) / 2;

 if (Vector[Middle] == Number)

 return true;

 else if (Vector[Middle] > Number)

 Right = Middle - 1;

 else

 Left = Middle + 1;

 }

 return false;

 }

 //empiryczna ocena z pomiarem czasu -średnia- znów 10 wyników więc znów pętla do 10

 for (int i = 0; i < 10; i++)

 {

 long startingTimeBINAVG = Stopwatch.GetTimestamp();

 isPresent2(array, rand.Next(0, 1001));

 long endingTimeBINAVG = Stopwatch.GetTimestamp();

 long elapsedTimeBINAVG = endingTimeBINAVG - startingTimeBINAVG;

 double elapsedSecondsBINAVG = elapsedTimeBINAVG \* (1.0 / Stopwatch.Frequency);

 results2[i] = elapsedSecondsBINAVG;

 }

 for (int i = 0; i < 10; i++) //znów pętla do wypisania wyników

 Console.WriteLine("BINARNA/SREDNIA | CZAS OPERACJI = "+results2[i]);

 //znów liczenie średniej od razu

 double result4 = (results2[0] + results2[1] + results2[2] + results2[3] + results2[4] + results2[5] + results2[6] + results2[7] + results2[8] + results2[9]) / 10;

 Console.WriteLine("BINARNA/SREDNIA | SREDNI CZAS OPERACJI = " + result4);

 //empiryczna ocena z pomiarem czasu -negatywna- element z poza zakresu tablicy - pętelka

 for (int i = 0; i < 10; i++)

 {

 long startingTimeBINNEG = Stopwatch.GetTimestamp();

 isPresent2(array, 1001); //znów 1001

 long endingTimeBINNEG = Stopwatch.GetTimestamp();

 long elapsedTimeBINNEG = endingTimeBINNEG - startingTimeBINNEG;

 double result5 = elapsedTimeBINNEG \* (1.0 / Stopwatch.Frequency);

 results6[i] = result5;

 Console.WriteLine("BINARNA/NEGATYWNA | CZAS OPERACJI = " + result5);

 }

 double result8 = (results6[0] + results6[1] + results6[2] + results6[3] + results6[4] + results6[5] + results6[6] + results6[7] + results6[8] + results6[9]) / 10;

 Console.WriteLine("BINARNA/NEGATYWNA | SREDNI CZAS OPERACJI = " + result8);

 int binarySearchOperations(int[] tab, int n) //funkcja do liczenia operacji szukania binarnego, tutaj też nie pamiętam dlaczego tak :c

 //tutaj też dwie, dla średniej i negatywnej

 {

 int left = 0;

 int right = tab.Length - 1;

 while (left <= right)

 {

 counter2++;

 int middle = (left + right) / 2;

 if (n == tab[middle])

 {

 return n;

 }

 else if (n < tab[middle])

 {

 right = middle - 1;

 }

 else

 {

 left = middle + 1;

 }

 }

 return -1;

 }

 int binarySearchOperationsNEG(int[] tab, int n) //dla opcji negatywnej

 {

 int left = 0;

 int right = tab.Length - 1;

 while (left <= right)

 {

 counter3++;

 int middle = (left + right) / 2;

 if (n == tab[middle])

 {

 return n;

 }

 else if (n < tab[middle])

 {

 right = middle - 1;

 }

 else

 {

 left = middle + 1;

 }

 }

 return -1;

 }

 for (int i = 0; i < 10; i++) //opcja średnia = znów ma być dla 10 losowych liczb, więc taka pętelka

 {

 counter2 = 0;

 binarySearchOperations(array, rand.Next(0, 1001));

 results4[i] = counter2;

 Console.WriteLine("BINARNA/SREDNIA | LICZBA OPERACJI = "+results4[i]);

 }

 int result6 = (results4[0] + results4[1] + results4[2] + results4[3] + results4[4] + results4[5] + results4[6] + results4[7] + results4[8] + results4[9]) / 10; //średnia liczba operacji

 Console.WriteLine("BINARNA/SREDNIA | SREDNIA LICZBA OPERACJI= " + result6);

 binarySearchOperationsNEG(array, 1001); //negatywna = poza zakresem

 Console.WriteLine("BINARNA/NEGATYWNA | LICZBA OPERACJI = " + counter3);

 Console.ReadLine();

 }

 }

}