MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

LIBRĂRIE DE FUNCȚII PENTRU MĂSURAREA TIMPULUI DE EXECUȚIE ȘI A NUMĂRULUI DE EXECUȚII PENTRU DIFERITE COMPONENTE

-Java-

Realizat de: Izabella Bartalus

Prof. coordonator: Gheorghe Sebestyen Pal

CUPRINS

1. Introducere

Obiectivul principal al acestui proiect a fost de a crea o biblioteca de functii/proceduri atat pentru masurarea timpului de executie, cat si pentru masurarea numarului de executii a diferitor componente ale programului.

Avem nevoie de un proiect de acest gen deoarece acest pachet de functii poate fi inserat in cadrul oricarei aplicatii care a fost scrisa folosind limbajul de programare Java. Daca proiectantul doreste sa vada unde se pierde mai mult timp in executia programului, sau cat timp a durat executia unei parti de cod, sau de cate ori s-a reapelat recursiv o anumita metoda, prin inserarea acestor functii intr-un program sursa, proiectantul poate vedea cu usurinta aceste lucruri, care, uneori pot fi utile deoarece stiind aceste rezultate, se poate lucra la optimizarea programelor.

2. Descrierea problemei

Pentru a putea realiza aceasta biblioteca de functii, am creat in Java un pachet de functii pe care l-am numit ‘myLibrary’ si care contine trei clase:

- MyTimer si StackEntry – care sunt folosite pentru masurarea timpului de executie a diferitelor componente;
- RecursionConter – folosita pentru a masura numarul de executii pentru diferite componente utilizate in program.

Pentru a putea oferi exemple de utilizare a functiilor implementat in cadrul pachetului ‘myLibrary’, in cadrul programului sursa au fost implementate totodata si anumite exemple de cod care sa permita utilizatorului sa observe rezultatele aduse de aceste functii.

Componentele inserate in program pentru care s-a calculate timpul de executie sunt:

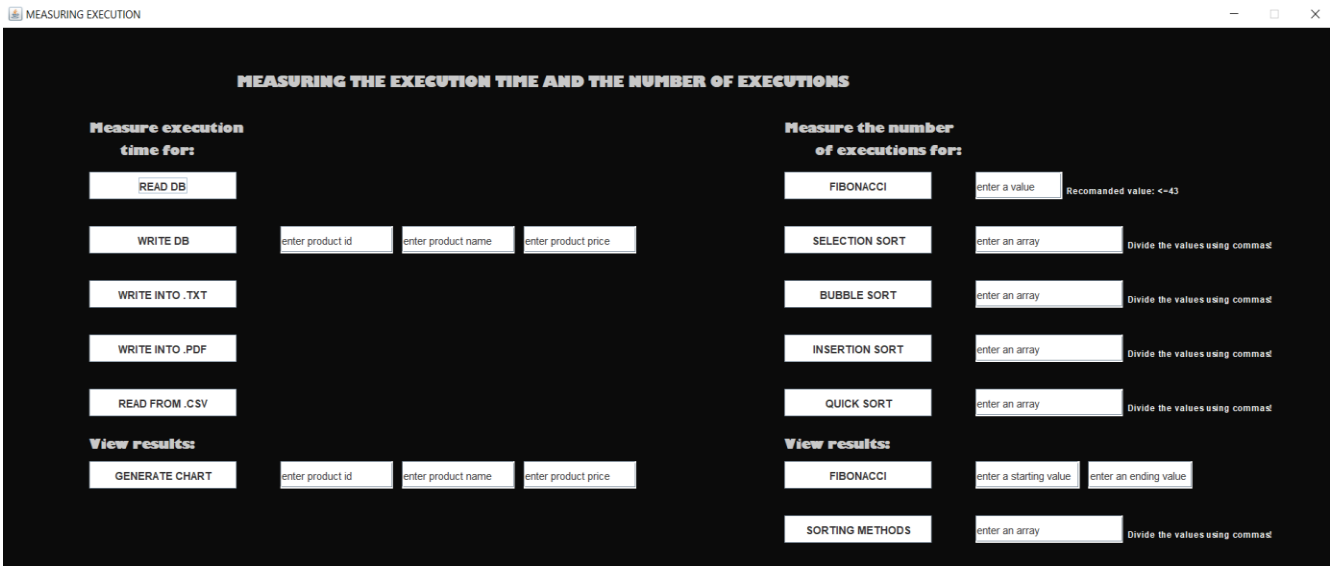
- Citirea dintr-o baza de date;
- Scrierea intr-o baza de date;
- Generarea unui fisier .txt;
- Generarea unui fisier .pdf;

- Citirea dintr-un fisier .csv.

Componentele inserate in program pentru care s-a calculat numarul de executii (numarul de apeluri recursive) sunt:

- Seria Fibonacci;
- Sortarea unui array de intregi folosind metoda de sortare Selection Sort;
- Sortarea unui array de intregi folosind metoda de sortare Bubble Sort;
- Sortarea unui array de intregi folosind metoda de sortare Insertion Sort;
- Sortarea unui array de intregi folosind metoda de sortare Quick Sort.

De asemenea, pentru a permite utilizatorului sa interactioneze cat mai mult cu programul, s-a creat o interfata grafica care ii permite acestuia sa aleaga dintre componentele enumerate mai sus si sa vizualizeze rezultatele in consola aplicatiei sau in mod grafic.



Daca un utilizator isi importa in proiectul sau acest pachet de functi si doreste:

- Sa calculeze timpul de executie pentru o componenta din programul, va proceda astfel:
  MyTimer.timeIt("Numele componentei");
  -----Cod componenta-----
  MyTimer.timeUp();
- Sa calculeze numarul de executii pentru o metoda recursiva:
  RecursionCounter counter = new RecursionCounter();
  numeMetoda(parametrii, counter);
  System.out.println("Number of executions for numeMetoda " + counter.usagetimes("numeMetoda"));

3. Perspective tehnice

Atat biblioteca de functii cat si componentele pe care se realizeaza testarea lor, au fost implementate folosind limbajul de programare Java. Pentru lucrul cu baza de date, s-a folosit MySQL.

4. Testare si rezultate obtinute

Testarea componentelor de pe coloana din stanga (a timpului de executie):

- Daca utilizatorul doreste sa vada timpul de executie pentru o citire din baza de date, tot ce trebuie sa faca este sa apese butonul ‘READ DB’ iar in consola va vedea rezultatul:

```
Reading from db takes: : 109 milliseconds
```

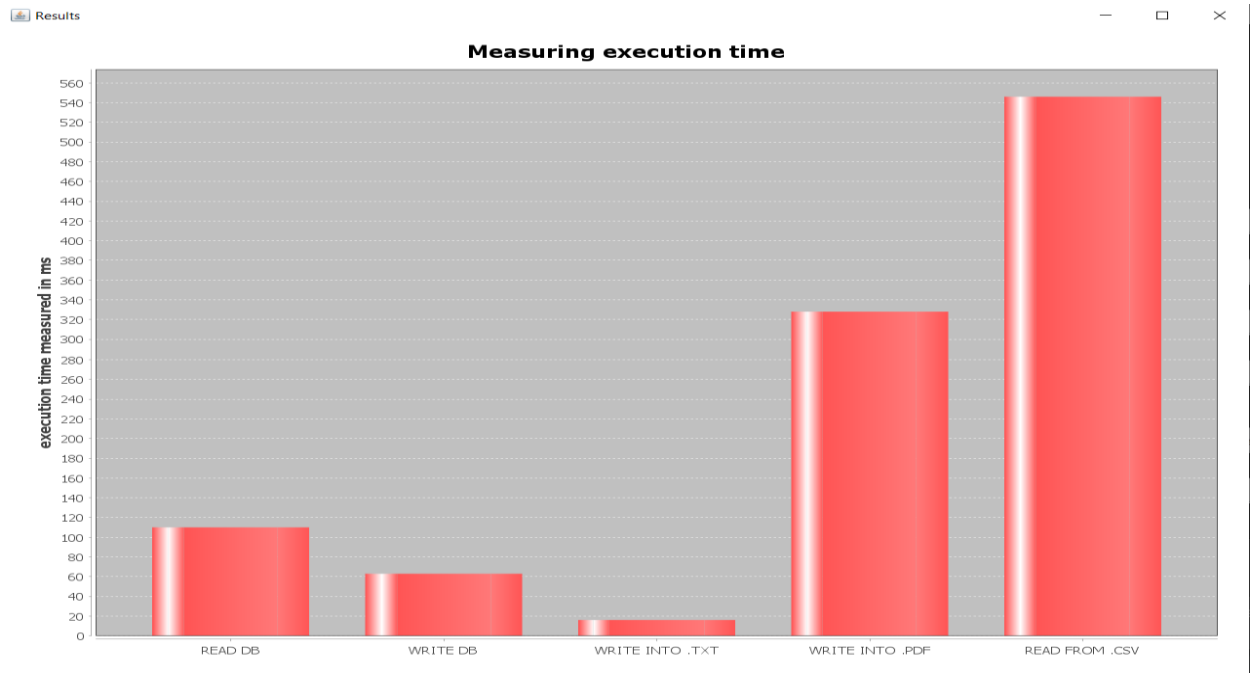
- Daca se doreste sa se vizualizeze timpul de executie pentru scrierea in baza de date, trebuie sa se introduca anumite date, precum: id-ul unui produs (sa fie unic in baza de date), numele unui produs si pretul acestuia, iar apoi sa apese butonul ‘WRITE DB’.

Apasand butonul ‘WRITE DB’ in consola apare rezultatul:

```
Writing into db takes: : 109 milliseconds
```

- Daca utilizatorul doreste sa vada in mod grafic timpul de executie pentru toate cele 5 componente implementate pe prima coloana, acesta trebuie sa completeze ce anume vrea sa insereze in baza de date, iar mai apoi sa apese pe butonul ‘GENERATE CHART’. Exemplu de generare grafic pentru masurarea timpului de executie: completam campurile pentru introducerea unui produs in baza de date:

Apasand butonul ‘GENERATE CHART’ obtinem:



Testarea componentelor de pe coloana din dreapta (a timpului de executie si a numarului de executii):

- Pentru seria Fibonacci 40, introducem valoarea 40 in casuta si apasam butonul ‘FIBONACCI’.

Rezultatul obtinut in consola:

```
Time taken for Fibonacci=40: 9761 milliseconds
Number of executions for Fibonacci=40: 331160281
```

- Daca se doreste vizualizarea timpului de executii si a numarului de apeluri recursive pentru a sorta un array de numere intregi folosind spre exemplu metoda Selection Sort, se va introduce in casuta alaturata butonului ‘SELECTION SORT’ un array (ex: -69, -98, -87, 93, -63, 31, -85, 82, 34, -18, 62, 96, 84, 98, 50, 54, -57, 5, -3, 27, 60, -29, 82, 91, -12, 11, 78, -24, 74, 33, -25, -65, 24, 70, 77, -39, 70, 56, -63, 90, -71, 2, 31, 33, 77, -70, -69, -32, 22, -34, 23, 21, 49, 18, 74, 21, 31, 4, -12, 32, -57, 53, -55, 76, -49, 15, 100, -91, 88, 11, 16, 15, 6, 46, -4, 90, -48, -38, -49, 3, 75, -56, 49, -65, 69, -3, 86, -20, -34, -42, 3, 95, 81, 12, -71, -5,

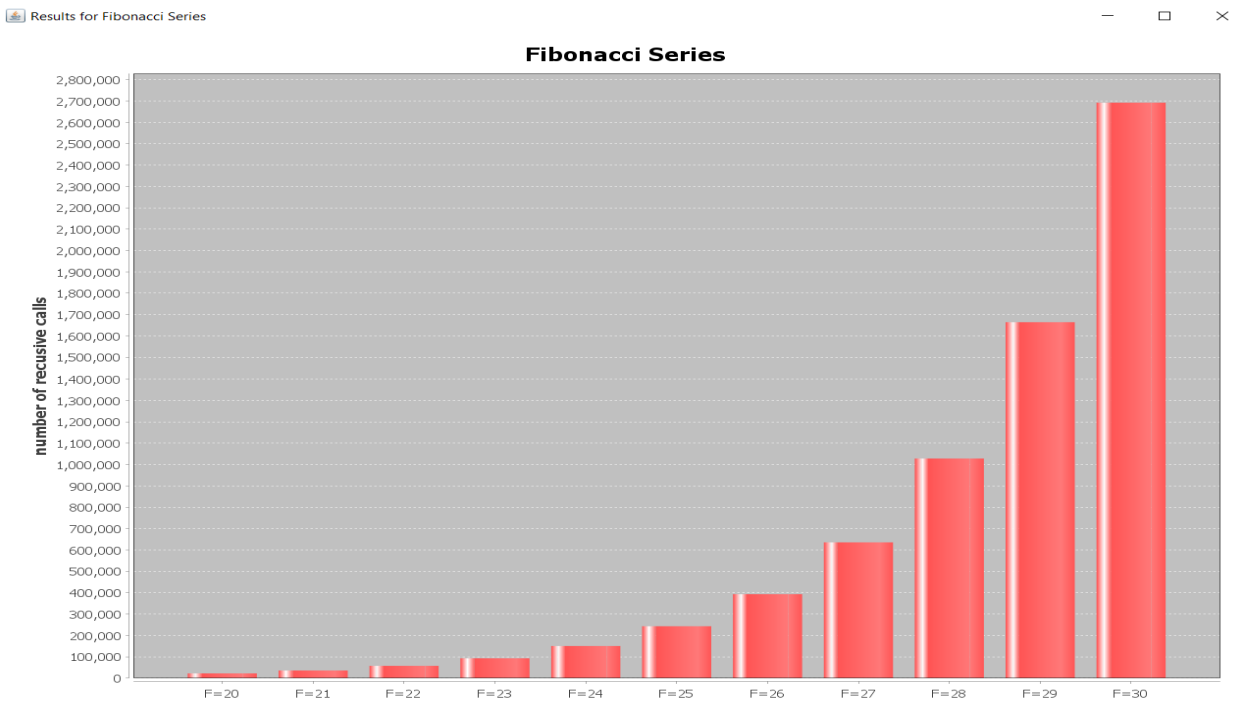
-91, 77, 32, 85, -43, 19, -36, 66, -42, -74, -22, 95, 79, 16, 57, -83, -32, 42, -75, -90, -92, 51, 82, 53, 10, -39, -23, 67, 45, 6, 30, -78, 69, 40, -20, -17, 8, -50, 62, 19, -13, 81, -97, -23, -19, -76, 59, 84, 36, 66, -23, 11, 30, 78, -6, -44, -25, -59, 21, 4, 58, 64, 50, -69, 50, -73, 98, 23, 85, 6, -44, 72, 66, 88, -61, 10, -67, -44, -97, -6, 27, 95, -16, 18, -74, -34, -90, -17, 68, -86, 49, -47, -34, 78, -36, 51, 7, 99, -32, 50, 14, 60, 47, -37):

Prin apasarea butonului ‘SELECTION SORT’ obtinem in consola rezultatul:

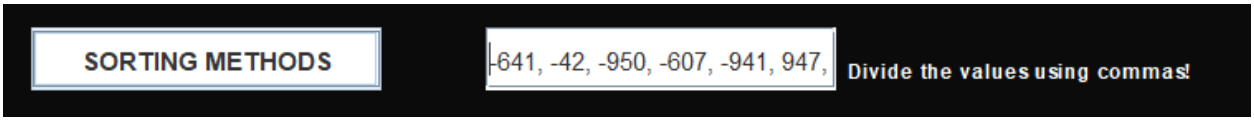
```
Time taken to sort an array using resursive selection sort: 16 milliseconds
Number of executions for Recursive Selection Sort: 201
```

- Daca dorim sa vizualizam in mod grafic numarul de executii a seriei Fibonacci de la o anumita valoare pana la o alta valoare, sub textul ‘View results:’ apare butonul ‘FIBONACCI’ alaturi de doua casete text unde introducem spre exemplu doua valori: de la 20 la 30:

Si apasand butonul ‘FIBONACCI’ obtinem graficul:



- Daca dorim sa vizualizam o comparatie grafica a numarului de executii dintre metodele de sortare implementate, in dreptul butonului ‘SORTING METHODS’ introducem un array (ex: -641, -42, -950, -607, -941, 947, 169, -825, 350, 690, 404, -18, -503, -330, 610, -433, -190, 83, -913, -468, -120, -740, 829, -97, -452, 301, -49, -885, 583, -755, -204, 562, -304, 174, -751, -227, 203, 116, 739, 254, 904, 267, 978, -570, -794, -263, 566, -473, -929, 537, -364, 817, -188, 545, -53, -250, 839, 257, -635, 648, 15, 455, -461, -316, 829, -975, -912, -839, 889, 395, -426, -415, -534, -190, -632, 179, 338, 40, 352, 54, 42, -679, -492, -691, 124, 410, 320, -615, 29, -442, -725, -219, 707, 752, 433, 897, -116, 18, 844, -174, 207, 859, -606, -710, -964, 697, -901, -840, -913, 33, 576, -927, -850, 596, 731, -973, -469, -22, -897, -558, 682, -488, -169, -710, -963, -574, -935, -597, 315, -350, 566, -845, 649, -646, -212, -852, 190, -359, 900, 574, -696, -820, -55, 96, 420, 306, 33, -921, -482, -31, -199, 939, -955, -939, 684, -880, 468, -389, 636, -128, 86, -678, 703, 959, -679, 623, 360, 343, 74, 658, 345, -200, -713, -714, -203, -291, 357, 877, -635, 396, -215, 306, 173, -293, -737, -185, 490, -809, -273, -134, -525, -341, 913, 753, -916, -876, -865, 981, 808, 411, -906, -553, -429, 43, -743, 489, -932, 772)



Apasand butonul ‘SORTING METHODS’ obtinem graficul:

5. Concluzii

In concluzie, in cadrul acestui proiect s-a implementat o biblioteca de functii care masoara timpul de executie si numarul de executii pentru diferite componente scrise in limbajul de programare Java.

Acest pachet de functii poate fi importat in cadrul oricarui proiect Java si este util pentru dezvoltatorii de programe care doresc sa isi optimizeze aplicatiile din punctul de vedere al timpului de executie, dar totodata si din punctul de vedere al numarului de executii pentru partile de program scrise in mod recursiv.