Documentaţia algorimului 1 - SmartBCM

I. Descrierea algorimului:

Algoritmul nostru construieşte figura finală unind piesele de 1x1 de pe fiecare nivel încercând să unească piesele într-o piesa cât mai mare pentru a creşte stabilitatea stucturii.

Deci, avem o sortare descrescăoare a pieselor şi în cel mai rău caz o să rămână o piese de 1x1.

Pentru a creşte stabilitatea ne mai uităm la nivelul anterior şi încercăm să punem piesa curenta cu orientare diferită de piesa\piesele de sub ea "sudând" astfel unele spaţii goale de la nivelul anterior.

Deci algoritmul nostru ar fi:

-pentru toate nivelele figurii de la bază înspre vârf:

-parcurgem matricea nivelului curent

-dacă piesa curentă nu este deja ocupată de o piesă pusă anterior construim o nouă piesă, alftel continuăm

-dacă putem pune o piesă încercăm să o punem astfel încât să creştem cât mai mult stabilitatea structurii

II. Design patterns folosite:

1) Mediator

def: Defines simplified communication between classes

in cazul nostru:

- În mod normal ar fi trebui să facem Get la toată matricea, să iterăm prin ea ca să putem obţine informaţii necesare pentru parcurgerea formei cum ar fi o anumită coloană.
- Clasa SmartMatrix are funcţii care simplifică foarte mult comunicarea între clase pentru că avem metode de Get, Set, Update atât pentru linii cât şi pentru coloane.

1) Strategy

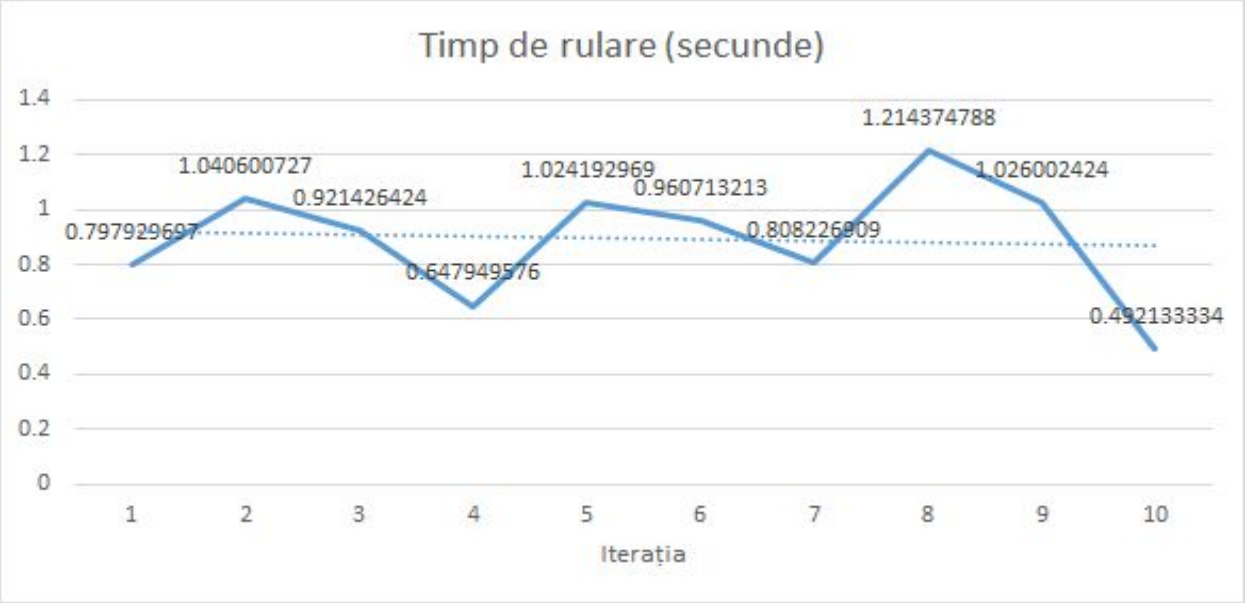
def: Encapsulates an algorithm inside a class

in cazul nostru:

- Clasa SmartBCM reprezintă algoritmul nostru (BucevschiCosminMădălin)

III. Performanţe şi îmbunătăţiri:

Pentru a stabili performanţele algorimului, am rulat de 10 ori şi am notat timpii de execuţie folosind un fişier de input de 4991 de piese de tip 1x1. (forma fişierului de input poate fi văzută la paragraful [VI]), obţinând următoarele rezultate:



Timpul mediu de rulare a algorimului pe acest set de date este 0.893355006 secunde.

IV. Forma fişierului de input:

Programul nostru primeşte ca date de intrare (dintr-un fişier) în format CSV forma (T, X, Z, Y, C, O) pe care trebuie să o contruim, astfel:

-T (tipul piesei care iniţial e 0 pentru toate piesele):

0 piesa 1x1
1 piesa 1x2
2 piesa 1x3
3 piesa 1x4
4 piesa 1x6
5 piesa 1x8
6 piesa 2x2
7 piesa 2x3
8 piesa 2x4
9 piesa 2x6
10 piesa 2x8

-X, coordonata X (orizontală)

-Z, coordonata Z (înălţime)

-Y, coordonata Y (verticală)

-C, un string reprezentând codul în hexadecimal pentru culoarea piesei

-O, orientarea piesei.

0 piesa e pusă normal
1 piesa e rotită cu 90 de grade

V. Link-ul catre github:

https://github.com/alex-bucevschi/Lego

VI. Documentaţie:

https://lgg.epfl.ch/publications/2013/lego/