Implementarea grafica a Hash-urilor

Enachioiu Sorin-Catalin

Profesor coordonator – Rodica Smîntîna

Instituţia – Colegiul National Gheorghe Şincai

Clasa – a XII-a E

Anul şcolar – 2021-2022

Prezentare generala

Proiectul utilizeaza limbajul de programare C++ pentru a transforma un text in trei matrici de valori printr-o functie de hashing. Aceste matrici sunt copiate in p5.js pentru a genera o imagine grafica unica.

Un hash este generat de un algoritm specializat, care, pe baza unui sir de caractere de intrare (teoretic de lungime infinita) furnizeaza la iesire o valoare fixa, unica. Functiile hash nu sunt inversabile.

Un hash poate fi generat atunci cand se trimite un mesaj, asigurand utilizatorii ca nu exista nici o pierdere de informatie (informatia receptata este identica cu informatia transmisa). Astfel, acest proiect poate fi utilizat in momentul in care este nevoie de transmiterea de informatii, mai ales a celor importante, care trebuie sa ramana nemodificate. Totodata, proiectul poate fi utilizat si in scopul autentificarii unui utilizator, deoarece acest hash permite ca o valoare numerica / text sa genereze o imagine unica. Exista, ca si in cazul oricarui hash, posibilitatea aparitiei unor asa numite *"coliziuni"* , insa spre deosebire de functia de hashing SHA-256 acest proiect are o probabilitate mai mica de a intampina coliziuni, deoarece numarul de imagini care pot fi generate este, in teorie, infinit, acesta fiind limitat doar de constrangeri hardware.

Din punct de vedere hardware, specificatiile pot varia in functie de dimensiunea textului de intrare, intrucat, complexitatea de memorie este strict dependenta de aceasta (O(n), unde n = lungimea textului introdus). De asemenea, timpul de executare este dependent atat de dimensiunea textului cat si de dimensiunea imaginii rezultate(in acest caz, complexitatea din punct de vedere al timpului de executare este fie $O(l^4)$, fie $O(l^2 * n)$, unde l = numarul de linii ale imaginii generate, si n = lungimea textului).

Din punct de vedere software, este necesar un compilator C++, cat si o conexiune la internet pentru a accesa pagina web p5.js.

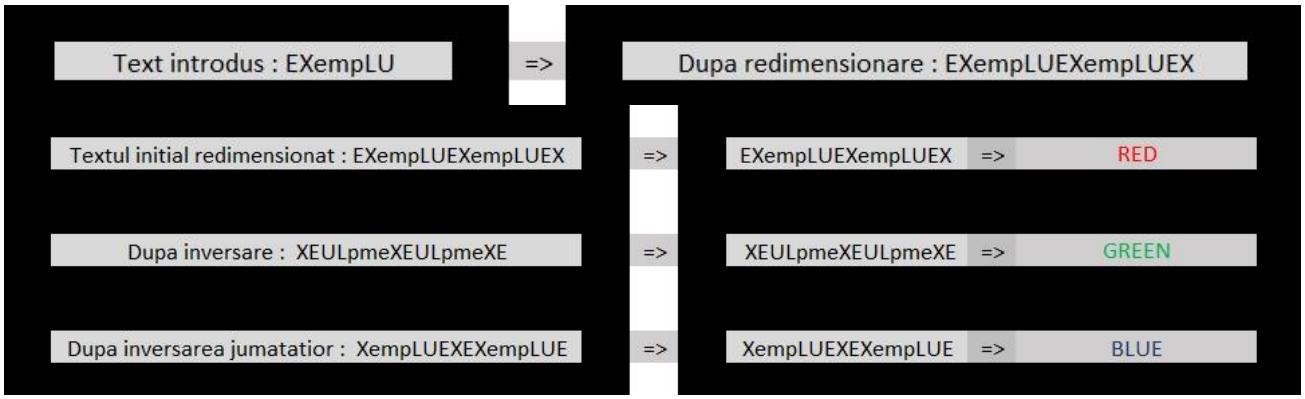
Descrierea proiectului

1. *Scurta descriere a aplicatiei*

Aplicatia primeste un text de orice dimensiune si genereaza in trei fisiere distincte (cu ajutorul limbajului de programare C++) trei tablouri bidimensionale care contin valori cuprinse intre 0-255 corespunzatoare celor trei culori necesare pentru a colora un pixel, respectiv rosu, verde si albastru. Cele trei matrici trebuie sa fie copiate de catre utilizator in p5.js, unde vor fi reunite pentru a genera un patrat unic format doar din pixeli.

2. *Algoritmul utilizat*

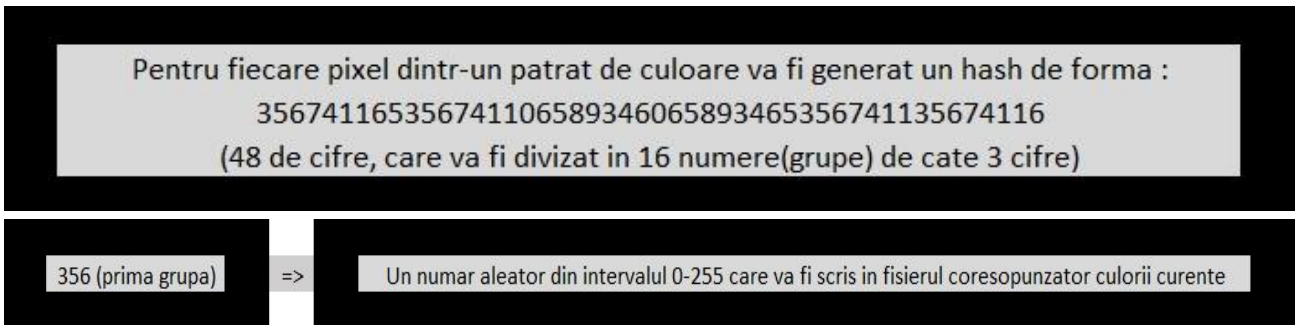
Proiectul preia un text dintr-un fisier de intrare si il redimensioneaza in cazul in care acesta are o dimensiune mai mica de 16 caractere. Pentru a genera valorile corespunzatoare culorilor rosu, verde si albastru se folosesc trei siruri de caractere – sirul initial (pentru culoarea rosu), sirul inversat (pentru culoarea verde), sirul in care prima si a doua jumatate a sa sunt interschimbate (pentru culoarea albastru).



Pentru a genera tabloul bidimensional corespunzator unei culori, programul genereaza in mod repetat hash-uri(ale caror valori depind de randul si coloana curenta, cat si de sirul initial transmis ca parametru): prin apelarea functiei HashGenerator.

```
243  /**
244   * functie care transforma in mod repetat sirul intr-un hash de 48 de caractere numerice,
245   * hash-ul fiind apoi folosit pentru a calcula culorile
246   **/
247  void HashGenerator(string s, int color)
248  {
249      string initial_s;
250      initial_s = s;
251
252      for(int crt_row = 1; crt_row <= MAX_ROWS; crt_row++)
253      {
254          for(int crt_col = 1; crt_col <= MAX_ROWS/16; crt_col++)
255          {
256              s = GenerateCurrentHash(s, initial_s, crt_row, crt_col);
257              WriteColors(s, color, crt_row, crt_col);
258          }
259      }
260  }
```

Fiecare hash este format din 48 de cifre. In momentul in care un hash este calculat, cifrele sale vor fi divizate in 16 grupe de cate 3 cifre care sunt utilizate pentru a genera o intensitate de culoare(din intervalul 0-255). Aceste intensitati sunt scrise in fisierul culorii curente.



```
200   /**
201    * functie care divizeaza hash-ul in grupe de cate 3 valori,
202    * pe care le converteste in valori intregi,
203    * folosite pentru a genera valori intre 0-255,
204    * corespunzatoare unei culori
205    **/
206   void WriteColors(string s, int color, int crt_row, int crt_col)
207   {
208       int to_color;
209
210       for(int crt_poz = 0; crt_poz < 48; crt_poz = crt_poz + 3)
211       {
212           to_color = FromSToInt(s, crt_poz);
213           to_color = (P * to_color + crt_poz * crt_row * crt_col) % 256
214           functions[color](1, "", to_color);
215       }
216
217   }
```

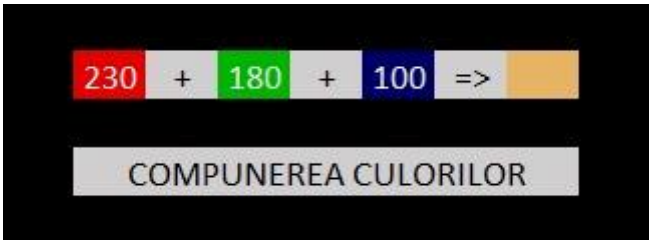
Pentru a simplifica afisarea a fost definit un vector ce contine pointeri spre trei functii, fiecare dintre acestea fiind conceputa pentru a scrie o valoare (atat de tip sir de caractere cat si numeric) in fisierul corespunzator unei culori.

```
16   // definim un pointer spre o functie
17   typedef void *(*functii) (bool op, string s, int nr);
```

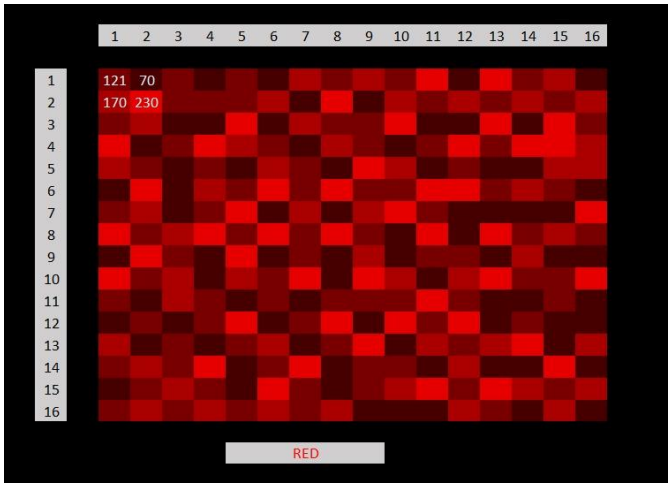
```
112  /**
113   * functie care scrie in fisierul corespunzator
114   * culorii rosii din matricea finala
115   * op = 1 => o sa fie scris un numar
116   * op = 0 => o sa fie scris un sir de caractere
117   **/
118  void *WriteToRed(bool op, string s, int nr)
119  {
120      if(op == 0)
121          red_color << s;
122      else
123          red_color << nr << ", " ;
124  }
```

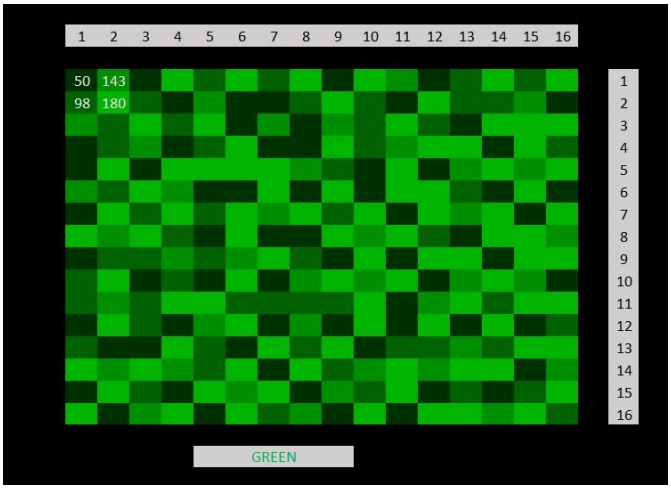
```
154   // cream un vector de pointeri spre
155   // functiile de scris in fisiere
156   functii functions[] =
157   {
158       WriteToRed,
159       WriteToGreen,
160       WriteToBlue
161   };
```

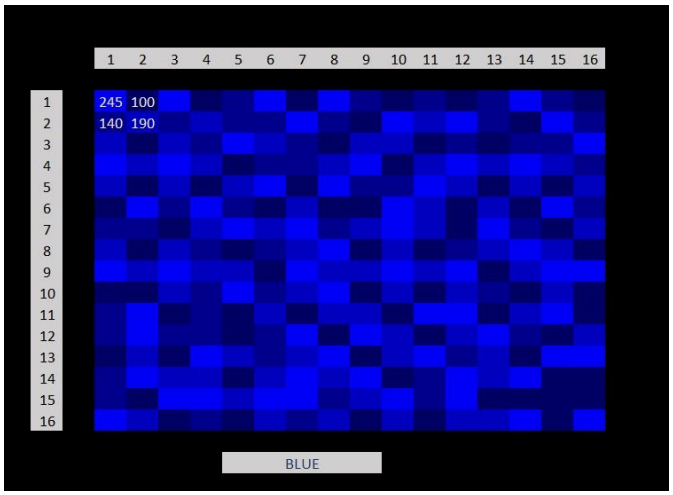
Tablourile bidimensionale generate sunt incarcate de utilizator in libraria p5.js , unde va avea loc afisarea grafica, combinand valorile din cele trei matrici.



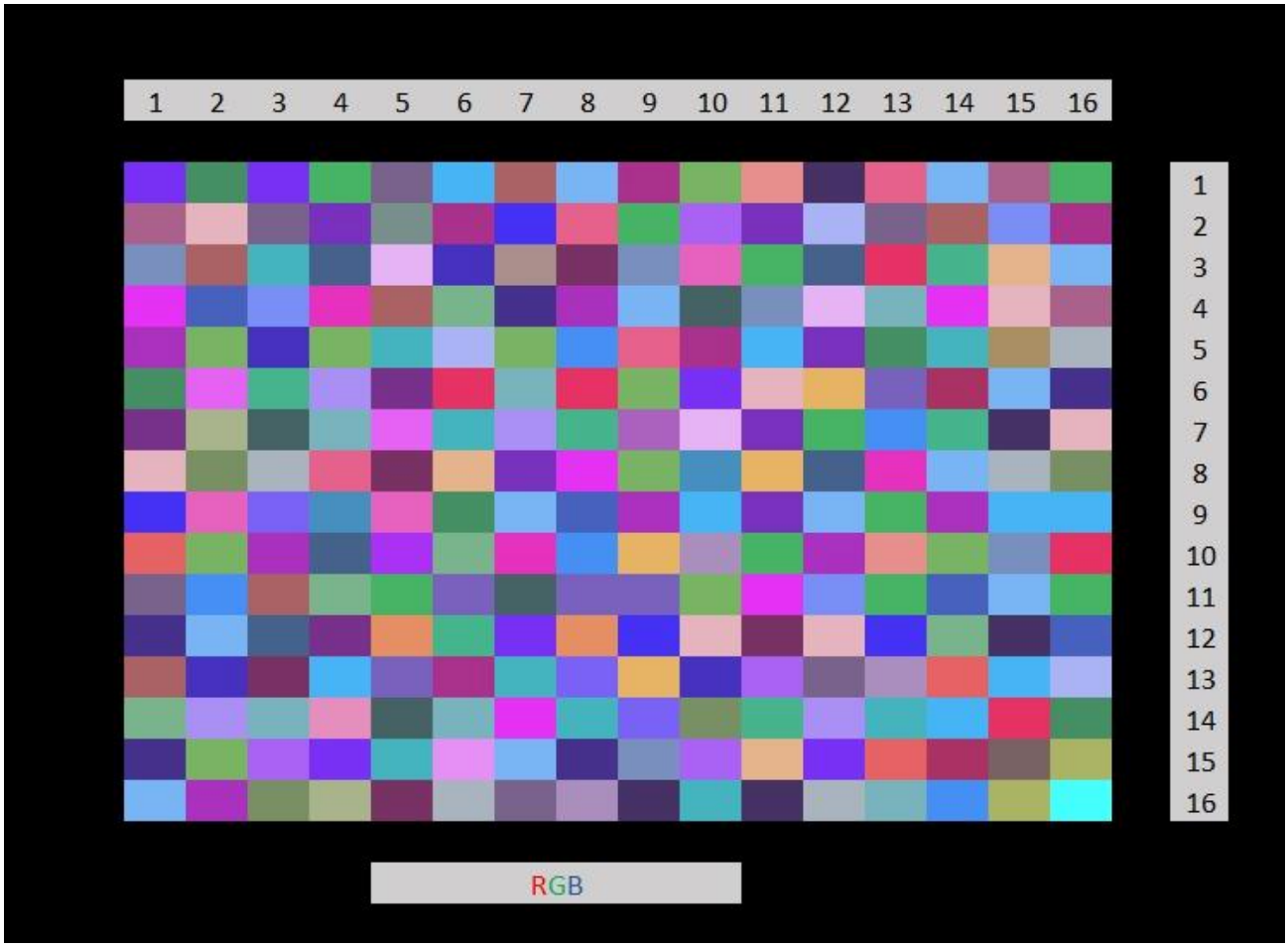
Separat, cele trei matrici ar arata astfel (in acest exemplu au fost folosite numai patru intensitati cu scopul de a facilita vizualizarea):







Rezultatul final dupa compunerea celor trei tablouri bidimensionale arata astfel:



3. Erori si corectarea lor

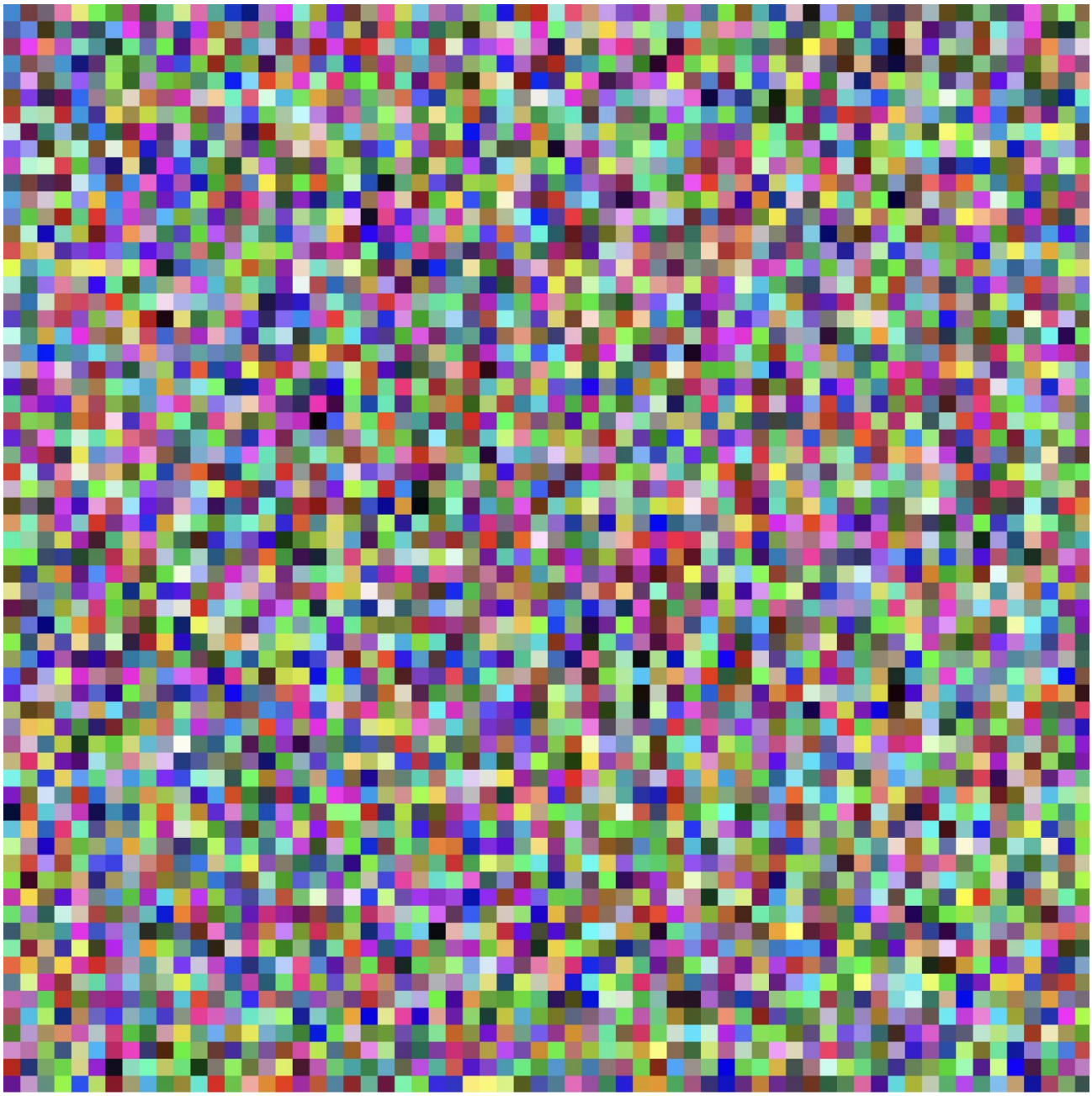
In cazul in care copierea celor trei tablouri bidimensionale(generate in C++) de catre utilizator nu este efectuata corect, atunci cand programul este executat exista doua erori posibile:

- Pe ecran apare un mesaj de eroare care semnaleaza ca una dintre matrici nu este definita corect.
- Imaginea generata contine in mod repetat aceeasi culoare, ceea ce inseamna ca una dintre matrici a fost copiata incomplet.

Pentru corectarea acestor erori este necesara repetarea copierii tablourilor bidimensionale.

4. Exemplu de utilizare

Daca fisierul de intrare contine sirul de caractere „EXempLU”, in urma executiei programului va fi afisata imaginea(cu 64 de randuri si 64 de coloane):



5. Facilitati de modificare

Pentru imbunatatirea atat a performantelor programului cat si a expereientei utilizatorului afisarea imaginii ar putea fi realizata direct in limbajul C++, aceasta ajutand simultan si la prevenirea posibilelor erori de utilizare.

In cazul in care programul va fi utilizat pentru autentificare, este recomandat ca in timpul procesarii parolei, acesteia sa i se adauge si un asa-numit *salt* (un *salt* este un sir de caractere unic si aleator adaugat fiecarei parole inainte sa fie hash-uita).