O imagine este o matrice de date.

O celulă a acestei matrici se numește pixel. Un pixel poate fi un numar natural, real sau complex.

Un pixel se reprezinta pe un număr finit de biți.

În funcție de tipul pixelilor, imaginile pot fi:

- Scalare – pixelii au doar 2 valori posibile, adică au un conținut binar, de obicei acesta fiind alb-negru/ gri.
- Vectoriale – pixelii sunt vectori, cel mai comun caz este cel RGB, când vectorul are 3 elemente corespunzătoare culorilor roșu, verde și albastru.

În cadrul proiectului meu am avut de convertit o imagine vectorială într-una scalară (binară – alb-negru).

Pentru acest lucru, m-am folosit de media valorilor R G B (practic, am convertit imaginea mai întâi în Greyscale) după care am comparat această valoare cu un threshold stabilit inițial (l-am setat ca fiind 127 , adica jumatate din maximul 255), iar dacă valoarea mediei culorilor este peste acest threshold, celula devine full negru și invers devine full alb. Imaginea este convertită , astfel, într-una binară.

Codul java corespunzător algoritmului este următorul:

```
for (int i = 0; i < binaryHeight; i++) {
                  for (int j = 0; j < binaryWidth; j++) {

                        Color c = new Color(inImg.img.getRGB(j, i));
 // descompunem in RGB fiecare culoare a imaginii
                        int red = (int) (c.getRed());
                        int green = (int) (c.getGreen());
                        int blue = (int) (c.getBlue());

                        int threshold = 127; //threshold-ul folosit pt
comparatie, de obicei jumatate din maximul de 255
```

```
                        if((red + green + blue) / 3 < threshold)  //conversia
are loc din grayscale in binary,

            //adica folosim media celor 3 culori in comparatie

            // pentru a face mai usor transformarea din gri in alb-negru
                        {
                              red = 0;
                              green = 0;  // full white
                              blue = 0;
                        }
                        else
                        {
                              red = 255;
                              green = 255; // full black
                              blue = 255;
                        }

                        Color newColor = new Color(red, green, blue); // este
preluata noua culoare de full white/full black


            binaryImage.setRGB(j, i, newColor.getRGB()); // pixelul respectiv
preia valoarea noua

}
}
```

Pentru implementare am utilizat următoarele clase:

- Interfața “ImageFetch” cu 2 metode pentru citirea căii spre fișier și pentru citirea fișierului
- Clasa abstractă ImageDimensions cu atributele width și height și metodele de tip getter/setter și afișare
- Clasa ReadImage ce implementează interfața și extinde clasa abstractă
- Clasa BinaryImage utilizată pentru conversie, având un atribut de tip ReadImage
- Clasa Main folosită pentru un aspect modular al aplicației

Ca exemple de funcționare a conversiei, putem observa în stânga imaginile inițiale, iar în dreapta, imaginile binare rezultate: