Documentatie proiect

Operatii AND, OR, XOR Intre Doua Imagini

Nume student: Sava Nicolae Bogdan

Grupa: 331

Seria: AB

An universitar: 2022-2023

Cuprins

[1. Functionalitate 3](#_Toc124156149)

[a. Selectare mod functionare 3](#_Toc124156150)

[b. Selectare actiune dorita 3](#_Toc124156151)

[c. Introducerea datelor 3](#_Toc124156152)

[d. Citirea imaginilor 4](#_Toc124156153)

[e. Producerea imaginilor si trimiterea catre consumator 4](#_Toc124156154)

[f. Trimiterea catre writer si scrierea imaginii finale 5](#_Toc124156155)

[2. Implementare 6](#_Toc124156156)

[a. Structura claselor 6](#_Toc124156157)

[b. Documentatie clase 6](#_Toc124156158)

[a. Image 6](#_Toc124156159)

[b. ThreadedImage 7](#_Toc124156160)

[c. ImageProducer 7](#_Toc124156161)

[d. ImageConsumer 7](#_Toc124156162)

[e. Buffer 7](#_Toc124156163)

[f. ImageWriter 8](#_Toc124156164)

[g. Timer 8](#_Toc124156165)

[c. Documentatie interfete 9](#_Toc124156166)

[a. ThreadedFunctionality 9](#_Toc124156167)

[b. Runnable 9](#_Toc124156168)

[d. Documentatie clase abstracte 9](#_Toc124156169)

[a. AbstractBuffer 9](#_Toc124156170)

[3. Exemplificarea functionalitatii cu imaginii 9](#_Toc124156171)

[a. Operatia AND 9](#_Toc124156172)

[b. Operatia OR 10](#_Toc124156173)

[c. Operatia XOR 11](#_Toc124156174)

# Functionalitate

## Selectare mod functionare

In momentul rulariile programului veti fi rugat sa alegeti daca doriti utilizarea unor imagini default (utilizate pentru exemplificare), sau daca doriti sa introduceti manual path-urile imaginilor ce urmeaza a fi procesate. Puteti face acest lucru introducand **y** (yes) sau **n** (no) de la tastatura.



Daca ati ales utilizarea path-urilor default, vor fi utilizate doua imagini aflate in folderul **src/imagini** (tot acolo se va afla si imaginea finala rezultata in urma procesarii) si puteti trece la pasul [1.b.](#_Selectare_actiune_dorita). Daca ati ales introducerea manuala a path-urilor pentru imaginile ce urmeaza a fi procesate, puteti trece la pasul [1.c.](#_Introducerea_datelor).

Daca alegeti modul default, dimensiunile imaginilor vor fi si ele cele default (1920x1080).

## Selectare actiune dorita

La acest pas sunteti rugat sa selectati ce tip de operatie sa se efectueze asupra celor doua imagini. Puteti face acest lucru introducand de la tastatura cifrele 1,2 sau 3 ce reprezinta operatiile AND, OR respectiv XOR.



In urma alegerii operatiei va incepe procesarea imaginilor si vor fi afisate mesajele corespunzatoare (pasul [1.d.](#_Citirea_imaginilor))

## Introducerea datelor

La acest pas veti fi rugat sa introduceti urmatoarele:

* Path-ul absolut al primei imagini ce urmeaza a fi procesata + extensia imaginii
* Path-ul absolut a celei de-a doua imagini ce urmeaza a fi procesata + extensia imaginii
* Path-ul absolut a imaginii finale (ce va rezulta in urma procesarii) + extensia imaginii
* Latimea imaginilor (intrucat cele 3 imagini trebuie sa aiba dimensiuni corespunzatoare, latimea si inaltimea trebuie sa corespunda, asa ca trebuie introduse o singura data)
* Inaltimea imaginilor

Mai jos este atasat un exemplu corect de completare a datelor:

Text, letter

Description automatically generated

In urma acestui pas puteti trece la pasul [1.b.](#_Selectare_actiune_dorita) pentru selectarea actiunii dorite.

## Citirea imaginilor

In prima etapa a procesului, vor fi citite cele doua imagini. Acest pas este unul esential intrucat are loc verificarea existentei celor doua imagini, dar si verificarea corespondentei dimensiunilor date cu cele ale imaginilor citite din fisier. In cazul in care una dintre verificari va esua, vor fi aruncate erori corespunzatoare; altfel vor fi afisate doua mesaje ca in imaginea alaturata:



Urmatoarea etapa este cea de producere si procesare a imaginilor.

## Producerea imaginilor si trimiterea catre consumator

In aceasta etapa cele doua imginii vor fi sectionate in 4 parti de catre doi producatori ce vor trimite asincron datele imaginilor catre un consumator prin intermediul unui buffer ce poate memora cate doua bucati de imagine la un moment dat.

Pe parcursul acestui proces vor fi afisate date despre cat la suta din imaginea ficarui consumator a fost trimisa si cat la suta din imaginea finala a fost procesat de catre consumator.

La final va fi afisata durata intregii etape de trimitire si procesare.

Text, letter

Description automatically generated

In urma acestei etape, va incepe trimiterea imaginii finale de catre producator catre Writer.

## Trimiterea catre writer si scrierea imaginii finale

In aceasta etapa va avea loc procesul de trimitere pe fragmente a imaginii finale (rezultate in urma pasului de procesare) catre writer. In urma asamblarii imaginii finale, aceasta va fi scrisa in fisier.

Pe parcursul acestei etape vor fi afisate mesaje despre statusul trimiterii imaginii catre writer, si, ulterior, va fi afisat un mesaj specific despre scrierea in fisier. In cazul in care scrierea este imposbilia (din cauza unui path introdus gresit/extensiei scrise gresit, etc..), va fi aruncata o exceptie corespounzatoare.

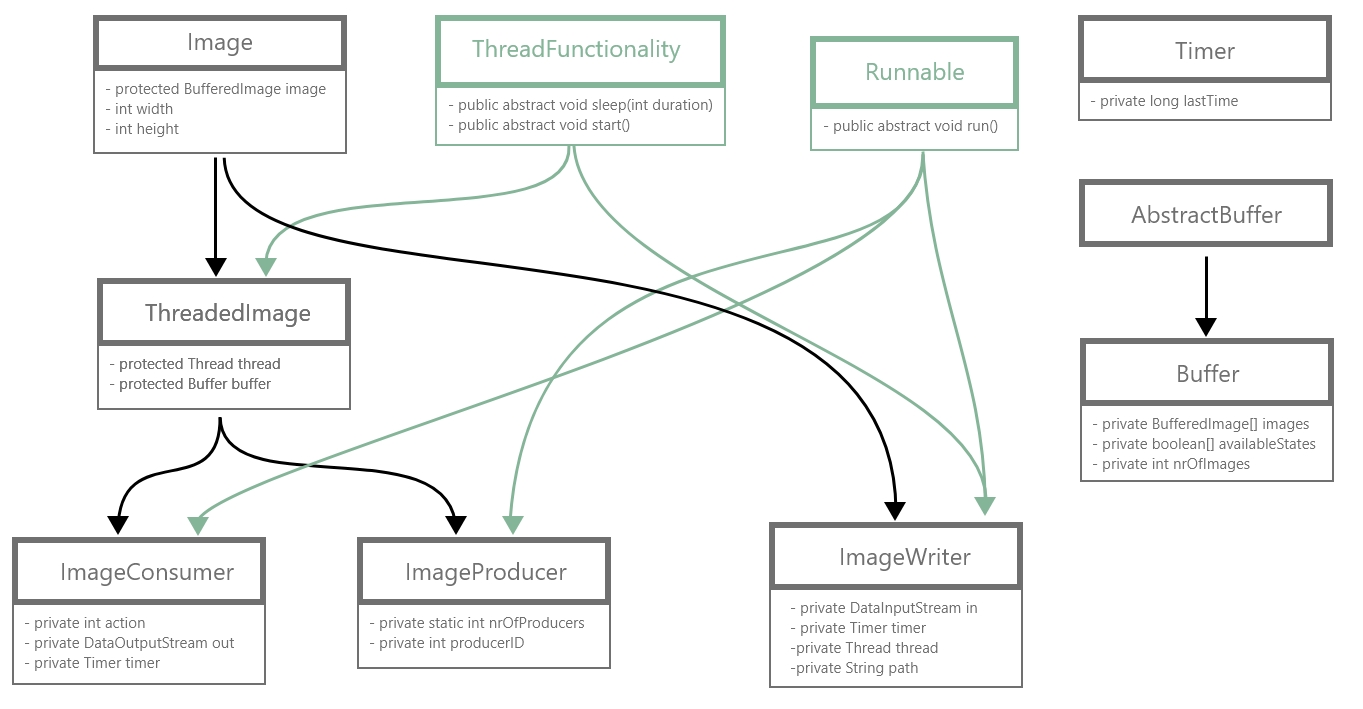
La finalul acestei etape va fi afisata durata procesului de primire si scriere a imaginii.

Text

Description automatically generated

# Implementare

## Structura claselor

In imaginea de mai jos este prezentata structura si ierarhia claselor utilizate. Interfetele sunt reprezentate cu culoarea verde. Relatiile de mostenire sunt reprezentate prin cate o sageata (culoarea neagra este pentru mostenirea claselor, iar culoarea verde este pentru implementarea interfetelor).

Am adaugat in imaginea anterioara si variabilele de clasa alaturi de tipul si nivelul lor.

## Documentatie clase

### Image

In aceasta clasa sunt stocate datele despre imagine precum inaltimea, latimea si obiectul BufferedImage ce reprezinta imaginea propriu-zisa.

Clasa dispune de doi constructori:

* Primul primeste ca parametrii doar dimensiunile imaginii, permitand instantierea ulterioara a imaginii propriu-zise
* Al 2-lea primeste ca parameterii atat dimensiunile imaginii cat si path-ul imaginii. In acest constructor imaginea va fi citita din fisier prin metode specifice si va fi incarcata in variabila de clasa image de tipul BuffredImage. Acest constructor este prevazut cu un sistem de gestionare al erorilor (daca dimensiunile nu corespund/daca imaginea nu poate fi citita

Clasa detine getteri si setter pentru latime si inaltime, intrucat acestea sunt variabile private.

### ThreadedImage

Aceasta clasa extinde clasa [Image](#_Image) si implementeaza interfata ThreadedFunctionality. Aceasta clasa este o clasa utilizata pentru pastrarea obiectului de tip Thread, cat si pentru executarea operatiilor specifice thread-urilor, de aceea are si doua metode suprascrise **start()** si **sleep(int duration)** .

Metoda **sleep()** apeleaza metoda **start()** a obiectului de tip Thread, porninf astfel un nou Thread. Metoda **sleep()** apeleaza intr-un bloc try catch metoda **sleep()** a obiectului Thread si ,,blocheaza” thread-ul un numar specificat de secunde (specificat prin intermediul parametrului).

Clasa dispune de 2 constructori, ce corepuns clasei Image mostenite. Acestia insa au inca un parameteru, acela fiind un obiect de tipul Buffer.

### ImageProducer

Aceasta clasa extinde clasa [ThreadedImage](#_ThreadedImage) si implementeaza interfata Runnable.

Una dintre metodele functiei este cea **run().** In aceasta metoda imaginea ce se afla in posesia producer-ului curent, este impartita in 4 zone ce sunt trimise ulterior drept sub-imagini catre buffer-ul ce se afla in posesia clasei mostenite. Aceasta printeaza in consola statusul fiecarei etape de trimitere.

Aceasta clasa dispune de un constructor ce are ca parametri path-ul imaginii ce va apartine producer-ului curent, dimensiunile imaginii si buffer-ul. Acest constructor apeleaza constructorul clasei mostenite cu parametrii coresp;unzatori. In aceasta clasa numarul de produceri este incrementat (mecanism ce ne ajuta la selectarea unui ID pentru fiecare producer deoarece avem de procesat 2 imagini) si este atribuit un ID pe baza acestui nou numar. Tot in acest constructor este creat un nou Thread.

### ImageConsumer

Aceasta clasa extinde clasa [ThreadedImage](#_ThreadedImage) si implementeaza interfata Runnable.

Una dintre metodele functiei este cea run(). In aceasta metoda are loc preluarea a cate doua bucati de imagine din buffer si executarea operatiei alese de utilizator la nivel de pixel asupra celor doua imagini. La fiecare pas este afisat un mesaj specific ce ne informeaza despreparcursul imaginilor. La finalul acestei metode este afisata durata procesarii si primirii imaginii, apoi se apeleaza metoda **SendProcessedDataToWriter**().

Metoda **SendProcessedDataToWriter**() executa trimiterea imaginii finale procesate catre clasa ImageWriter prin intermediul unui pipe prin care este trimis fiecare pixel.

Clasa dispune de un constructor ce primeste ca parametrii dismeniunile imaginilor ce vor fi procesate, un obiect de tipul Buiffer, actiunea aleasa de utilizator si Pipe-ul prin care va fi trimisas informatia catre clasa de scriere. Acest constructor apeleaza constructorul clasei mostenite cu parameterii specifici, instantiaza o imagine goala cu dimensiunile date unde vor fi salavti pixelii procesati, si creaza un nou Thread.

### Buffer

Aceasta clasa este folosita pentrun transmiterea datelor intre consumator si producator, si extinde clasa abstracat [AbstractBuffer](#_AbstractBuffer).

Aceasta clasa are ca variabile un vector de **BufferedImages** (deoarece trebuie prelucrate cate doua bucati de imagine simultan), un vector de boolean-uri (deoarece trebuie sa stim daca fiecare bucata de imagine este disponibila sau nu pentru a fi preluata de catre consumator), si numarul de imagini ce se pot afla simulatn in buffer.

Constructorul acestei clase primeste ca parametru numarul de imagini ce se pot afla simultanm in buffer si instantiaza cu valori nule cei doi vectori.

Metodele clasei sunt urmatoarele:

* *public synchronized BufferedImage[]* **GetImages**() – aceasta metoda are rolul de returna vectorul format din imaginile ce vor fi procesate ulterior (daca ambele sunt disponibile) si setarea ulterioara a statusurilor tuturor imaginilor ca fiind indisponibile
* *public synchronized void* **put**(BufferedImage image, int currentImage) – aceasta imagine adauga cate o imagine in buffer pe locul dat drept parametru (acest loc este de fapt ID-ul producatorului care vrea sa adauge o imagine noua). In urma adaugarii se seteaza si statusul imaginii ca fiind available
* *private void* **setAvailable**(boolean state) – aceasta metoda seteaza toate valorile din vectorul de statusuri ca fiind valoarea data ca aparametru
* *private boolean* **getAvailable**() – aceasta metoda verifica daca toate imaginile sunt disponibile. Returneaza true daca toate sunt sau false daca exista cel putin una care nu este disponibila

### ImageWriter

Aceasta clasa extinde clasa Image si implementeaza interfetele Runnable si ThreadedFunctionality.

Aceasta clasa dispune de un constructor ce primeste ca parameterii path-ul imaginii finale, dimensiunile imaginilor si un Pipe prin intermediul caruia va primi datele rezultate in urma procesarii. In acest constructor este creat un nou Thread. Constructorul acesta apeleaza constructorul clasei mostenite cu parameterii corespunzatori.

In aceasta clasa se gaseste metoda **run**() in care este instantiat obiectul image din clasa mostenita cu dimensiunile date. Aici sunt primiti pixelii procesati prin intermediul obiectului de tip DataInputStream. La fiecare sfertb de informatie primit va fi afisat un mesaj corespunzator ce ne va spune progresul curent. L afasritul primirii imaginii este afisat un mesaj corespunzator si durata procvesului de primire a datelor. Ulterior imaginea finala este scrisa in fisier, proces ce se afla intr-un bloc try/catch ce va arunca erori corespunzatoare cauzate de diferite defectiuni tehnice. In final va fi afisat un mesaj corespunzator finalizarii scrierii imaginii in fisier.

In aceasta clasa se afla si metodele **start**() si **sleep**() suprascrise similar ca in cazul clasei [ThreadedImage](#_ThreadedImage).

### Timer

Aceasta este o clasa auxiliara ce ne ajuta sa masuram durata fiecarei etape prin intermediul unor timeStamp-uri.

Metodele clasei sunt urmatoarele:

* **Constructorul** – initiaza lastTime cu 0
* *public void* **Set**() – seteaza timeStamp-ul la momentul exact in care este apelata functia
* *public double* **GetSeconds**() – returneaza diferenta de timp dintre timeStamp-ul curentr si cel setat anterior (valoarea este returnata in secunde)
* *public long* **GetMillis**() – returneaza durata similar cu metoda descrisa anterior, dar in milisecunde

## Documentatie interfete

### ThreadedFunctionality

Aceasta este o interfata cu doua metode, specifice Thread-urilor, folosita pentru a preveni implementarea anevoioasa a claselor ce au un obiect de tip Thread in posesie.

### Runnable

Aceasta este interfata Java utilizata pentru lucurul cu Thread-uri, ce obliga implemenetarea metodei **run**().

## Documentatie clase abstracte

### AbstractBuffer

Aceasta este clasa abstracta utilizata pentru definirea metodelor **GetImages**() si **put**() specifice unui buffer. Aceste metode sunt override in clasa [Buffer](#_Buffer).

# Exemplificarea functionalitatii cu imaginii

## Operatia AND

Mai jos se pot observa doua imaginin alese aleator drept input, si o a 3-a imagine ce reprezinta iesirea programului atunci cand alegem sa efectuam operatia AND intre cele doua imagini.

**Imaginile alese sunt urmatoarele**



**Imaginea rezultata va arata astfel**

## Operatia OR

Mai jos se pot observa doua imaginin alese aleator drept input, si o a 3-a imagine ce reprezinta iesirea programului atunci cand alegem sa efectuam operatia OR intre cele doua imagini.

**Imaginile alese sunt urmatoarele**

A picture containing grass, field, nature, green

Description automatically generatedA bridge over a river

Description automatically generated

**Imaginea rezultata va arata astfel**

## Operatia XOR

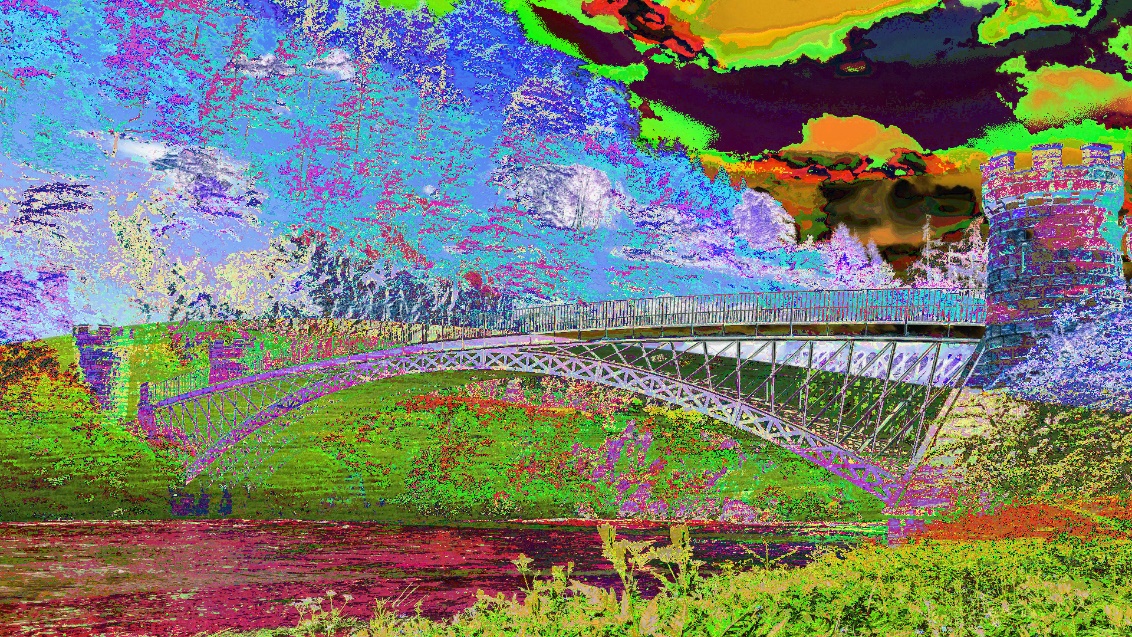
Mai jos se pot observa doua imaginin alese aleator drept input, si o a 3-a imagine ce reprezinta iesirea programului atunci cand alegem sa efectuam operatia XOR intre cele doua imagini.

**Imaginile alese sunt urmatoarele**

A picture containing grass, field, nature, green

Description automatically generatedA bridge over a river

Description automatically generated

**Imaginea rezultata va arata astfel**