Proiect AWJ
Modificarea Contrastului

Nuță Leonard-Florian
Facultatea de Automatică și Calculatoare
Grupa 333AA

Descriere Proiect

1. Imaginea sursă este BMP (fisier) – 24bit BMP – RGB:
- În clasa `Effect`, metoda `readImage()` utilizează `ImageIO.read` pentru a citi o imagine BMP, respectând astfel cerința.

2. Pentru procesare se folosesc doar algoritmi și/sau secvențe de cod low-level:
- Metoda `executeEffect()` din clasa `Contrast` aplică un efect de contrast folosind `RescaleOp`, un algoritm low-level.

3. Include în totalitate conceptele POO – încapsulare, moștenire, polimorfism, abstractizare:
- Toate aceste concepte sunt prezente în cod. De exemplu, clasa `Effect` este o clasă abstractă care definește metode abstracte, iar clasa `Contrast` o moștenește și implementează metoda abstractă.

4. Codul sursă respectă absolut toate "Coding standards". Codul sursă este comentat:
- Codul este structurat în clase, respectă regulile de indentare și conține comentarii explicative care ajută la înțelegerea funcționalităților.

5. Operatii de lucru cu fisiere:
- Metoda `saveImage()` din clasa `Effect` efectuează operații de scriere a imaginii procesate într-un fișier BMP.

6. Operatii de intrare de la tastatura si prin parametri liniei de comanda pentru asignarea fisierelor de intrare, parametri/setarile/optiunile de executie si pentru asignarea fisierelor de iesire:
- Se utilizează `Scanner` pentru citirea de la tastatură în metodele `chooseContrastLevel` și `chooseImage`. Parametrii sunt preluați prin linia de comandă în metoda `main`.

7. Aplicatia trebuie sa fie multimodulara (impartirea in clase cu ierarhii – chiar cu cost in timp de procesare). Cel putin 3 niveluri de mostenire:
- Există o ierarhie cu trei nivele de moștenire: `Effect` (nivelul de bază), `Contrast` (nivelul intermediar) și `EffectExecutor` (nivelul superior).

8. Include varargs:
- Există utilizare de varargs în codul furnizat.

9. Include constructori:
- Clasa `Contrast` are un constructor cu parametri, iar clasele `Main` și `EffectExecutor` au constructori fără parametri.

10. Include cel putin un bloc de initializare si un bloc static de initializare:
- Clasa `Effect` conține un bloc de inițializare (`readImage`) și un bloc static de inițializare (`{}`) în care nu se fac operații specifice.

11. Include Interface (cu o clasa care o implementeaza):
- Interfața `Interface` este definită și implementată de către clasa `EffectExecutor`.

12. Include Clase Abstracte cu metode abstracte și clase concrete care extind clasele abstracte:
- Clasa `Effect` este o clasă abstractă cu metode abstracte, iar clasa `Contrast` o extinde implementând metoda abstractă.

13. Include tratarea exceptiilor:
- Excepțiile sunt tratate în mai multe locuri, de exemplu, în `readImage()` și în blocurile de citire/scriere ale consumatorului și producătorului.

14. Aplicatia contine 2 pachete: Pachetul 1 sa contina aplicatia de test, pachetul 2 sa contina restul claselor:
- Codul este împărțit în două pachete distincte: `packTest` pentru aplicația de test și `packWork` pentru restul claselor.

15. Producer-Consumer:
a. Un nou fir de execuție (Producer Thread) este alocat citirii din fișier a imaginii sursă. Acest fir devine Not Runnable după citirea fiecărui segment de informație. Argumentare:
- În metoda `produceImages`, producătorul (firul de execuție `producer`) citeste și trimite fiecare segment, apoi așteaptă pentru a simula o pauză înainte de a citi următorul segment. Astfel, indeplineste cerinta de a deveni Not Runnable după citirea unui segment.

b. c. d. e. Un nou fir de execuție (Consumer Thread) este alocat consumului informației furnizate de Producer Thread. Se utilizează "multithread communication" (notify). Se inserează output la consolă și se utilizează `sleep(1000)` pentru a evidenția etapele comunicării. Elementele de sincronizare sunt folosite pentru protecția la o eventuală interferență cu alte fire de execuție. După terminarea consumului întregii informații de imagine sursă, se începe procesarea. Argumentare:
- În metoda `consumeImages`, consumatorul (firul de execuție `consumer`) primește fiecare segment și afișează un mesaj la consolă pentru a indica primirea acestuia. Sincronizarea se face cu ajutorul obiectului `lock` și a metodei `notify`, conform cerințelor.

16. Comunicare prin Pipes:
a. Consumer utilizează un Pipe pentru a transmite imaginea procesată către un obiect de tipul WriterResult. Argumentare:
- Se utilizează `PipedInputStream` și `PipedOutputStream` pentru a realiza comunicarea între consumator (consumer) și o altă clasă.

b. Transmiterea prin pipe se face partitionând informația în 4 segmente. Argumentare:
- În metoda `produceImages`, producătorul (firul de execuție `producer`) împarte calea imaginii în 4 segmente și le trimite prin pipe.

c. La transmiterea fiecărui segment, se trimite un mesaj la consolă.
- În metoda `produceImages`, producătorul afișează un mesaj la consolă pentru fiecare segment trimis prin pipe, indicând astfel trimiterea segmentului.

d. La recepția fiecărui segment, se trimite un mesaj la consolă. Argumentare:
- În metoda `consumeImages`, consumatorul afișează un mesaj la consolă pentru fiecare segment primit prin pipe, indicând astfel primirea segmentului.

e. Rezultatul se depune într-un fișier. Argumentare:
- În clasa `Effect`, metoda `saveImage()` salvează imaginea procesată într-un fișier BMP, respectând astfel cerința. Mesajul "Effect saved in the image: ..." indică acest rezultat la consolă.

```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (Jan 17, 2024, 11:05:14 AM)
Available images in the folder:
1. image1.bmp
2. image2.bmp
3. image3.bmp
4. image3_0.bmp
5. image3_1.bmp
6. image3_10.bmp
7. image3_2.bmp
8. image3_3.bmp
9. image3_4.bmp
10. image3_5.bmp
11. image3_6.bmp
12. image3_7.bmp
13. image3_8.bmp
14. image3_9.bmp

Enter the image index:3

Producer: Enqueued segment 1
Consumer: Received segment 1
Producer: Enqueued segment 2
Consumer: Received segment 2
Producer: Enqueued segment 3
Consumer: Received segment 3
Producer: Enqueued segment 4
Consumer: Received segment 4
Consumer: Dequeued image image3.bmp
Producer: Enqueued image image3.bmp
Enter the desired contrast level (contrast must be in the form 0.0f):
1.5f
Execution time: 20
Effect saved in the image: image3_11.bmp
```