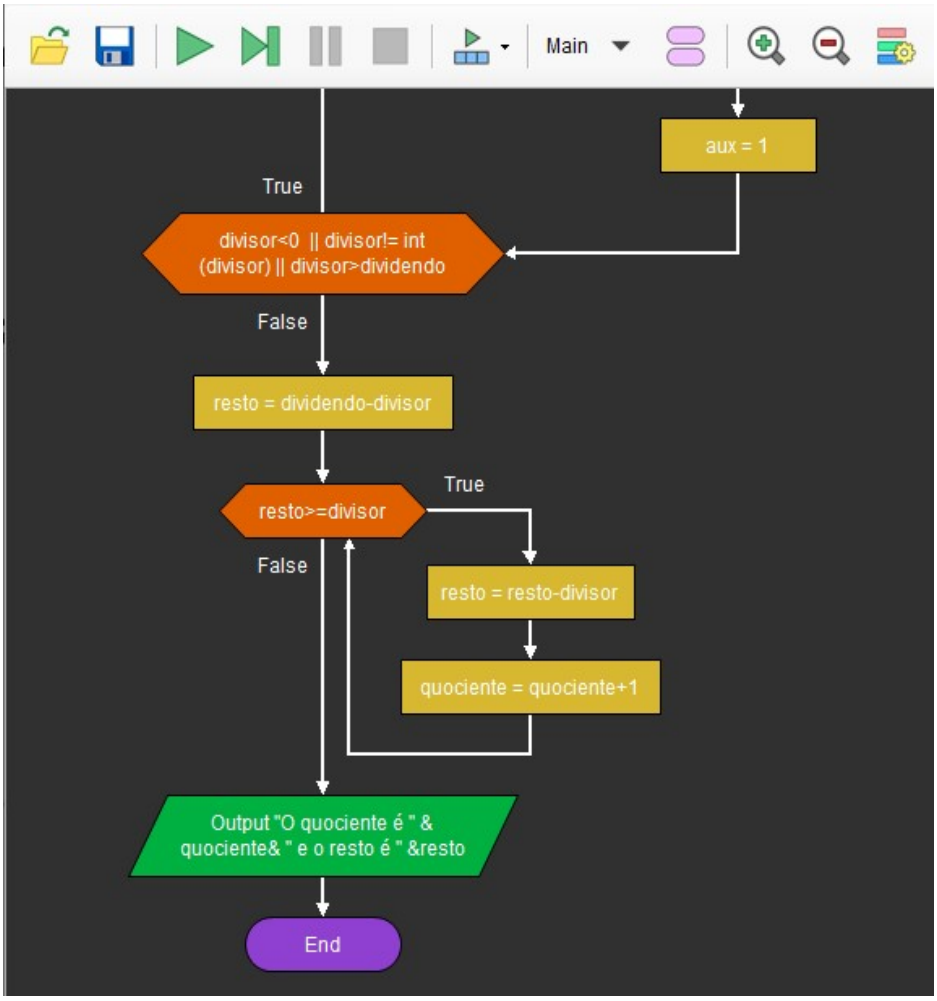
Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e Multimédia

LEIM

Tecnologias de Informação – TI – 2122SI

Trabalho Prático 2



Trabalho Realizado por: Pedro Silva Nº48965

Docente Engº Carlos Júnior

Lisboa, 23 de dezembro de 2021

Índice de matérias

1. Ciclos 1
1.1. TP2_01_MostrarNumerosEntre....1
1.2. TP2_02_CapitalFirstChar....2
1.3. TP2_03_TrianguloCardinais 3
1.4. TP2_04_PPMCross 4
2. – Métodos....5
2.1. TP2_05_EscreveDigitos 5
2.2. TP2_06_DesenhaEmPPM 6
2.3. TP2_07_DesenhaBandeirasEmPPM 11
2.4. TP2_08_DesenhaDegradeEmPPM....16
3. Arrays....18
3.1. TP2_09_MergeWithoutRepetitions....18

Índice de figuras

Figura 1-TP2_01_MostrarNumerosEntre -code....1
Figura 2-TP2_01_MostrarNumerosEntre - output....1
Figura 3-TP2_02_CapitalFirstChar - code....2
Figura 4-TP2_02_CapitalFirstChar - output....2
Figura 5- TP2_03_TrianguloCardinais - code....3
Figura 6- TP2_03_TrianguloCardinais - output....3
Figura 7-TP2_04_PPMCross - code....4
Figura 8-TP2_04_PPMCross - output....4
Figura 9- TP2_05_EscreveDigitos - code(1) Figura 10- TP2_05_EscreveDigitos - code(2) ..5
Figura 11- TP2_05_EscreveDigitos - output....5
Figura 12-TP2_06_DesenhaEmPPM-code(1) Figura 13-TP2_06_DesenhaEmPPM-code(2)..6
Figura 14-TP2_06_DesenhaEmPPM-code(3)....7
Figura 15-TP2_06_DesenhaEmPPM-code(4)....7
Figura 16-TP2_06_DesenhaEmPPM-code(5)....7
Figura 17-TP2_06_DesenhaEmPPM-code(6)....8
Figura 18-TP2_06_DesenhaEmPPM-code(7)....8
Figura 19-TP2_06_DesenhaEmPPM-code(8)....9
Figura 20-TP2_06_DesenhaEmPPM-code(9)....9
Figura 21-TP2_06_DesenhaEmPPM-output(1) Figura 22-TP2_06_DesenhaEmPPM-output(2) 10
Figura 23-TP2_06_DesenhaEmPPM-output(3) Figura 24-TP2_06_DesenhaEmPPM-output(4) 10
Figura 25-TP2_06_DesenhaEmPPM-output(5)....10
Figura 26-TP2_07_DesenhaBandeirasEmPPM-code(1)....11
Figura 27-TP2_07_DesenhaBandeirasEmPPM-code(2)....11
Figura 28-TP2_07_DesenhaBandeirasEmPPM-code(3)....12
Figura 29-TP2_07_DesenhaBandeirasEmPPM-code(4)....12
Figura 30-TP2_07_DesenhaBandeirasEmPPM-code(5)....13
Figura 31-TP2_07_DesenhaBandeirasEmPPM-code(6)....13
Figura 32-TP2_07_DesenhaBandeirasEmPPM-output(1)....14
Figura 33-TP2_07_DesenhaBandeirasEmPPM-output(2)....14
Figura 34-TP2_07_DesenhaBandeirasEmPPM-output(3) Figura 35-TP2_07_DesenhaBandeirasEmPPM-output(4)....15

Figura 36-TP2_07_DesenhaBandeirasEmPPM-output(5) Figura 37-TP2_07_DesenhaBandeirasEmPPM-output(6) 15

Figura 38-TP2_08_DesenhaDegradeEmPPM - code....16

Figura 39-TP2_08_DesenhaDegradeEmPPM - output(1) 17

Figura 40-TP2_08_DesenhaDegradeEmPPM - output(2) 17

Figura 41-TP2_08_DesenhaDegradeEmPPM - output(3) 17

Figura 42-TP2_09_MergeWithoutRepetitions-code(1) Figura 43-TP2_09_MergeWithoutRepetitions-code(2) 18

Figura 44-TP2_09_MergeWithoutRepetitions-output....18

1. Ciclos

1.1. TP2_01_MostrarNumerosEntre

Conceber a classe TP2_01_MostrarNumerosEntre que peça e leia ao/do utilizador dois números inteiros e que mostre os números inteiros desde o menor dos números introduzidos até ao maior (ambos incluídos). Por cada 10 números mostrados deve-se mudar de linha. Exemplo de interação: Introduza dois inteiros: 20 7 ENTER Entre 7 e 20 há os seguintes números: 7 8 9 10 11 12 13 14 15 16 17 18 19 20

```java
import java.util.Scanner;

public class TP2_01_MostrarNumerosEntre {

 public static void main(String[] args) {

  Scanner keyboard = new Scanner(System.in);

  System.out.println("O programa vai mostrar os numeros entre dois valores introduzidos pelo utilizador");

  System.out.print("Introduza o primeiro valor -> ");

  int val1 = keyboard.nextInt();

  System.out.print("Introduza o segundo valor -> ");

  int val2 = keyboard.nextInt();

  int i=1, aux=0 ;

    keyboard.close();

    if(val1>val2){
      aux=val1;
      val1=val2;
      val2=aux;
    }

  System.out.println(String.format("O valores entre %d e %d sao: ", val1, val2));

  for (i=1;val1!=val2+1;i++ ) {

    System.out.print(" " + val1);
    val1++;
    if(i%10==0){
    System.out.println("\n");
    }

  }
  }
}
```

Figura 1-TP2_01_MostrarNumerosEntre -code

```
O programa vai mostrar os numeros entre dois valores introduzidos pelo utilizador
Introduza o primeiro valor -> 5
Introduza o segundo valor -> 20
O valores entre 5 e 20 sao:
 5 6 7 8 9 10 11 12 13 14

 15 16 17 18 19 20
```

Figura 2-TP2_01_MostrarNumerosEntre - output

1.2. TP2_02_CapitalFirstChar

Conceber a classe TP2_02_CapitalFirstChar.java que no seu main: peça para ler e leia um texto/String/nome; e que depois o mostre mas todo em minúsculas, exceto que a primeira letra de cada palavra deverá ficar em maiúsculas. A primeira letra de cada palavra poderá ocorrer depois de alguns dígitos que possam iniciar a palavra. Considera-se um palavra a colocar a primeira letra em maiúscula, as palavras com 3 ou mais caracteres. Exemplo de interação: Introduza um texto: 50CENT na prÓXIMA SEXTA-feira ENTER Texto processado: 50Cent na Próxima Sexta-feira

```java
import java.util.Scanner;

public class TP2_02_CapitalFirstChar {

 public static void main(String[] args) {

  Scanner keyboard = new Scanner(System.in);

  System.out.println("O programa vai transformar a frase introduzida pelo utilizador mudando a primeira letra de cada palavra para maiuscula e as restantes minusculas\n");

  System.out.print("Introduza a frase -> ");

  String frase = keyboard.nextLine();

  String frasef="";

  int i,p=1;

  char space= ' ';

    keyboard.close();

  for (i=0;i<frase.length();i++ ) {

    char c=frase.charAt(i);

    if(i==0){
      c= Character.toUpperCase(c);
      frasef= frasef+c;
      continue;
    }

    if (Character.isDigit(c)==true && p==i){
      frasef= frasef+c;
      i++;
      p++;
      c=frase.charAt(i);
      if(Character.isLetter(c)==true){
        c= Character.toUpperCase(c);
        frasef= frasef+c;
        continue;
      }
      else{
        frasef= frasef+c;
        continue;

    }
    }

      if(c==space){
        frasef= frasef+c;
        i++;
        c=frase.charAt(i);
        if(Character.isLetter(c) && Character.isLetter(frase.charAt(i+1))  && Character.isLetter(frase.charAt(i+2)))
          c= Character.toUpperCase(c);
          frasef= frasef+c;
          continue;
      }
      if (i>=frase.length()){
        continue;
      }

          c=frase.charAt(i);
          c= Character.toLowerCase(c);
          frasef= frasef+c;

  }

  System.out.println(frasef);
```

Figura 3-TP2_02_CapitalFirstChar - code

```
O programa vai transformar a frase introduzida pelo utilizador mudando a primeira letra de cada palavra para maiuscula e as restantes minusculas

Introduza a frase -> 50CENT Na pROXIma SExta fEIRA
50Cent Na Proxima Sexta Feira
```

Figura 4-TP2_02_CapitalFirstChar - output

1.3. TP2_03_TrianguloCardinais

Conceber a classe TP2_03_TrianguloCardinais.java que no seu main: peça para ler e leia o número de linhas; e que depois mostre um triângulo de cardinais a começar com 1 cardinal, e ir progredindo com mais dois cardinais por linha, até ao número de linhas pretendido, em que os cardinais têm de ficar centrados e a formar um triângulo como se mostra no exemplo. Exemplo de interação: Indique o número de linhas: 3 ENTER

```
import java.util.Scanner;

public class TP2_03_TrianguloCardinais {

public static void main(String[] args) {

  Scanner keyboard = new Scanner(System.in);

  System.out.println("\n O programa vai pedir ao utilizador o numero de linhas em que vao ser introduzidos mais um cardinal que na linha anterior \n");

  System.out.print("Introduza o numero de linhas -> ");

  int n = keyboard.nextInt();

  int i,p,l=1;

  keyboard.close();

  for (i=1;i<n+1;i++){
    int space=50-i;
    String format = "%" + space + "c";

    System.out.printf(format,' ');

    for (p=0;p<l;p++){
      System.out.print("#");
    }
    l=l+2;
    System.out.print("\n");
  }
 }
}
```

Figura 5- TP2_03_TrianguloCardinais - code

```
 O programa vai pedir ao utilizador o numero de linhas em que vao ser introduzidos mais um cardinal que na linha anterior

Introduza o numero de linhas -> 10
                                        #
                                       ###
                                      #####
                                     #######
                                    #########
                                   ###########
                                  #############
                                 ###############
                                #################
                               ###################
```

Figura 6- TP2_03_TrianguloCardinais - output

1.4. TP2_04_PPMCross

Conceber a classe TP2_04_PPMCross.java que no seu main: peça para ler e leia a dimensão de uma imagem a gerar, e que, depois, gere uma imagem PPM, com esse número de píxeis em x e em y, e com o seu conteúdo com as cores de vermelho e branco, de modo a formar uma cruz, com as diferentes zonas a ocuparem 1/3 da imagem, tal como se mostra na figura ao lado. Exemplo de interação: Indique a dimensão da imagem: 100 ENTER [ver imagem ao lado]. Nota: para os problemas em PPM deve-se ver o respetivo anexo no 2º grupo de exercícios e usar o método disponibilizado para gravar em ficheiro uma imagem.

```java
import java.util.Scanner;

import java.io.*;

public class TP2_04_PPMCross{

  public static final String RED = "255 0 0 ";
  public static final String WHITE = "255 255 255 ";

  public static void main (String[] args){

    System.out.println("Indique a dimensao da imagem");
    Scanner keyboard = new Scanner(System.in);
    int dim = keyboard.nextInt();
    StringBuilder image = new StringBuilder();
    for (int y=0; y<dim;y++){
      for (int x=0; x<dim;x++){
        if ((x<(dim)/3 && y<dim/3) || (x>(dim*2)/3 && y<dim/3) || (x>(dim*2)/3 && y>(dim*2)/3) || (x<(dim)/3 && y>(dim*2)/3) ){

          image.append(WHITE);
        }else{

          image.append(RED);
        }


      }
      image.append("\n");


    }
    writeToPPMFile("CROSS.ppm",dim,dim,image.toString());
    keyboard.close();
    System.out.println("Done...");


  }

  public static void writeToPPMFile(String fileName, int xPixels, int yPixels,
  String content) {
  try (PrintWriter pw = new PrintWriter(new File(fileName))) {
  pw.println("P3"); // magic PPM P3 number
  pw.println(xPixels + " " + yPixels); // nXPixels nYPixels
  pw.println("255"); // max color value
  pw.println(content); // image content
 } catch (FileNotFoundException e) {
  e.printStackTrace();
  }
  }


}
```

Figura 7-TP2_04_PPMCross - code

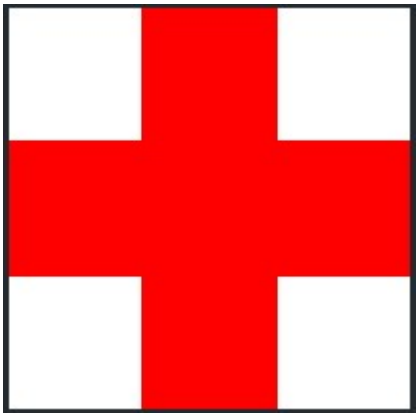


Figura 8-TP2_04_PPMCross - output

2. – Métodos

2.1. TP2_05_EscreveDigitos

Conceber a classe TP2_05_EscreveDigitos.java que peça ou utilizador um número inteiro positivo, e que escreva por ordem os seus dígitos. Exemplo de interação: Introduza um número inteiro positivo: 2490 ENTER O número 2490 é composto pelos dígitos: dois, quatro, nove e zero. Para tal, conceba e use os seguintes métodos: String getDigitoEmString(byte n), que recebendo um dígito em formato byte, devolve o valor textual dele ("um", "dois", …); int getDigito(int n, int idxDigito), que recebendo um número inteiro positivo e o índice de dígito (0 corresponde ao dígito de menor peso) devolve o dígito com esse índice do número recebido; int getNumDigitos(int n), que recebendo um inteiro positivo devolve o número de dígitos que ele tem; e void mostrarDigitos(int n), que recebendo um número inteiro positivo mostre na consola os seus dígitos, tal como pretendido – este método deverá chamar os outros três métodos – mas só deverão chamar o método getNumDigitos uma única vez no código.

Figura 9- TP2_05_EscreveDigitos - code(1)

Figura 10- TP2_05_EscreveDigitos - code(2)

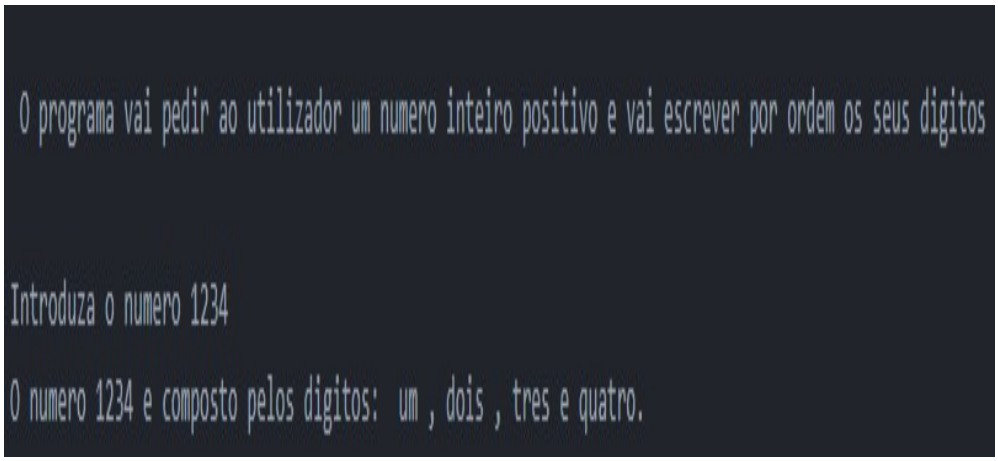
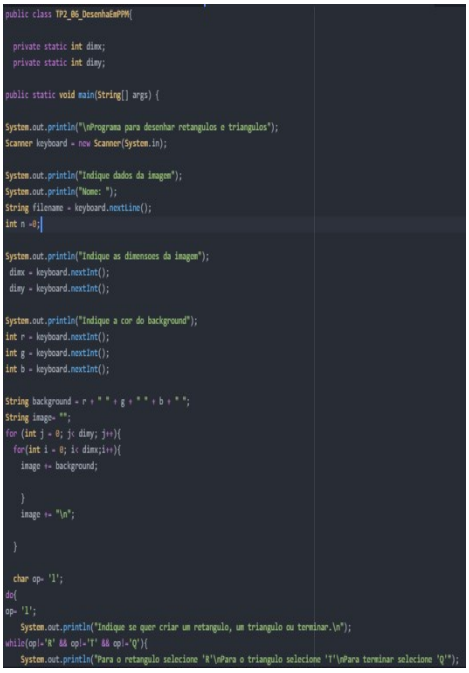


Figura 11- TP2_05_EscreveDigitos - output

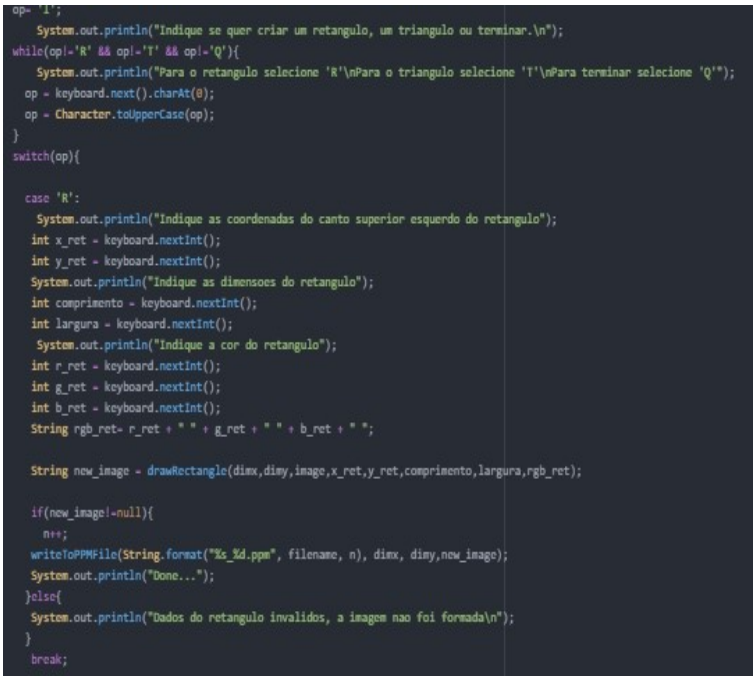
2.2. TP2_06_DesenhaEmPPM

Conceber a classe TP2_06_DesenhaEmPPM.java que permita criar desenhos, com retângulos e triângulos, em ficheiros PPM. O programa deverá começar por pedir os dados da imagem a gerar (nome da imagem, dimensões (dimx, dimy) e RGB de fundo) e depois pedir ao utilizador para indicar o que quer fazer: colocar um retângulo ou um triângulo, ou terminar. Caso escolha pela colocação de uma forma, deve pedir os dados da mesma: num retângulo, deve pedir o canto superior esquerdo (x, y), dimX, dimY e RGB da forma; num triângulo, que será um triângulo retângulo, deve pedir o vértice reto (x, y), deltaX, deltaY e RGB da forma, em que deltaX e deltaY podem ser negativos. Para tal deverão serem criados os métodos: drawRectangle e drawTriangle, que deverão receber: uma imagem PPM numa String; as dimensões da imagem (dimx, dimy); e os dados para gerar a figura. No método drawRectangle a String com a imagem PPM em texto, poderá vir a null, o que indica que se deve criar uma imagem de raiz com os dados da imagem, dos dados do retângulo apenas é considerado o RGB que irá preencher a totalidade da imagem. Ambos os métodos só deverão desenhar a figura caso esta respeite os limites da imagem. Ambos os métodos deverão devolver a String com a nova imagem. Após cada chamada a drawRectangle e drawTriangle o programa deverá gerar um ficheiro com a imagem corrente, onde cada ficheiro deverá ter um nome, que será composto com o nome inicial da imagem e "_n.ppm", em que n será o número de versões gravadas (utilizar String.format("%s_%d.ppm", fileName, n) para gerar o nome de forma dinâmica.



```
public class TP2_06_DesenhaEmPPM{

  private static int dimx;
  private static int dimy;

public static void main(String[] args) {

System.out.println("\nPrograma para desenhar retangulos e triangulos");
Scanner keyboard = new Scanner(System.in);

System.out.println("Indique dados da imagem");
System.out.println("Nome: ");
String filename = keyboard.nextLine();
int n =0;

System.out.println("Indique as dimensoes da imagem");
 dimx = keyboard.nextInt();
 dimy = keyboard.nextInt();

System.out.println("Indique a cor do background");
int r = keyboard.nextInt();
int g = keyboard.nextInt();
int b = keyboard.nextInt();

String background = r + " " + g + " " + b + " ";
String image= "";
for (int j = 0; j< dimy; j++){
  for(int i = 0; i< dimx;i++){
    image += background;

    }
    image += "\n";

  }

  char op= 'l';
do{
op= 'l';
    System.out.println("Indique se quer criar um retangulo, um triangulo ou terminar.\n");
while(op!='R' && op!='T' && op!='Q'){
    System.out.println("Para o retangulo selecione 'R'\nPara o triangulo selecione 'T'\nPara terminar selecione 'Q'");
```

Figura 12-TP2_06_DesenhaEmPPM-code(1)



```
op= 'l';
    System.out.println("Indique se quer criar um retangulo, um triangulo ou terminar.\n");
while(op!='R' && op!='T' && op!='Q'){
    System.out.println("Para o retangulo selecione 'R'\nPara o triangulo selecione 'T'\nPara terminar selecione 'Q'");
  op = keyboard.next().charAt(0);
  op = Character.toUpperCase(op);
}
switch(op){

  case 'R':
    System.out.println("Indique as coordenadas do canto superior esquerdo do retangulo");
   int x_ret = keyboard.nextInt();
   int y_ret = keyboard.nextInt();
   System.out.println("Indique as dimensoes do retangulo");
   int comprimento = keyboard.nextInt();
   int largura = keyboard.nextInt();
    System.out.println("Indique a cor do retangulo");
   int r_ret = keyboard.nextInt();
   int g_ret = keyboard.nextInt();
   int b_ret = keyboard.nextInt();
   String rgb_ret= r_ret + " " + g_ret + " " + b_ret + " ";

   String new_image = drawRectangle(dimx,dimy,image,x_ret,y_ret,comprimento,largura,rgb_ret);

   if(new_image!=null){
     n++;
   writeToPPMFile(String.format("%s_%d.ppm", filename, n), dimx, dimy,new_image);
   System.out.println("Done...");
  }else{
   System.out.println("Dados do retangulo invalidos, a imagem nao foi formada\n");
  }
   break;
```

Figura 13-TP2_06_DesenhaEmPPM-code(2)

```
   case 'T':
   System.out.println("Indique as coordenadas do vertice reto do triangulo (x,y)");
    int x_tri = keyboard.nextInt();
    int y_tri = keyboard.nextInt();
   System.out.println("Indique as dimensoes do triangulo");
    int deltax = keyboard.nextInt();
    int deltay = keyboard.nextInt();
    System.out.println("Indique a cor do triangulo");
    int r_tri = keyboard.nextInt();
    int g_tri = keyboard.nextInt();
    int b_tri = keyboard.nextInt();
    String rgb_tri= r_tri + " " + g_tri + " " + b_tri+ " ";

    new_image = drawTriangle(dimx,dimy,image,x_tri,y_tri,deltax,deltay,rgb_tri);

    if(new_image!=null){
      n++;
    writeToPPMFile(String.format("%s_%d.ppm", filename, n), dimx, dimy,new_image);
      System.out.println("Done...");
    }else{
      System.out.println("Dados do triangulo invalidos, a imagem nao foi formada\n");
    }

   break;

   case 'Q':
    System.out.println("End...\n");
   break;

}

}while(op!='Q');

keyboard.close();
```

Figura 14-TP2_06_DesenhaEmPPM-code(3)

```
public static String drawRectangle(int dimx,int dimy,String image, int x_ret, int y_ret, int comprimento, int largura, String rgb_ret){
  String new_image ="";
  if(image==null){
   for (int j = 0; j< dimy; j++){
     for(int i = 0; i< dimx;i++){
       new_image += rgb_ret;
     }
       new_image += "\n";
   }
   }else{

  int startPointX = x_ret;
  int endpointX = x_ret+comprimento;
  int startPointY = y_ret;
  int endpointY = y_ret+largura;

  if (startPointX>=dimx || endpointX>dimx || startPointY>=dimy || endpointY>dimy
  || startPointX<0 || comprimento<0 || startPointY<0 || largura<0){

    return null;
  }

  String background="";
  int p=0;
  for(int i=0;p<3;i++){
  if(image.charAt(i)==' ' ){
    background+=" ";
     p++;

  }else{
    background+=image.charAt(i);
  }
  }
  for(int y = 0; y<dimy; y++){
    for(int x = 0; x< dimx; x++){
      if (x >= startPointX && x< endpointX && y >= startPointY && y < endpointY)  {

            new_image +=  rgb_ret;
```

Figura 15-TP2_06_DesenhaEmPPM-code(4)

```
   }else{
     background+=image.charAt(i);
   }
   }
   for(int y = 0; y<dimy; y++){
     for(int x = 0; x< dimx; x++){
       if (x >= startPointX && x< endpointX && y >= startPointY && y < endpointY)  {

           new_image +=  rgb_ret;
         }else{
           new_image+= background;
         }
       }
         new_image+="\n";
       }
     }
   return new_image;
  }

  public static String drawTriangle(int dimx,int dimy,String image, int x_tri, int y_tri, int deltax, int deltay, String rgb_tri){
    int startPointX = x_tri;
    int startPointY = y_tri;
    int endpointX = 0;
    int endpointY = 0;
    StringBuilder new_image = new StringBuilder();
    String background="";
    int p=0;

    if(endpointX<0 || endpointY<0 || startPointX<0 || startPointY<0 ||
    startPointX>=dimx || endpointX>dimx || startPointY>=dimy || endpointY>dimy  ){
      return null;
    }

    for(int i=0;p<3;i++){
    if(image.charAt(i)==' ' ){
      background+=" ";
       p++;
```

Figura 16-TP2_06_DesenhaEmPPM-code(5)

```
    }else{
      background+=image.charAt(i);
    }
  }
rgb_tri= " "+ rgb_tri;
background= " "+ background;


 if(deltax>0 && deltay>0){
  endpointX = startPointX+deltax;
  endpointY = y_tri+deltay;
  int j=0;
  String rgb=rgb_tri;

  if(endpointX<0 || endpointY<0 || startPointX<0 || startPointY<0 ||
  startPointX>=dimx || endpointX>dimx || startPointY>=dimy || endpointY>dimy  ){
    return null;
  }

  for(int y = 0; y<dimy; y++){
    for(int x = 0; x< dimx; x++){
      if (x == startPointX && y >= startPointY && y < endpointY )  {
            new_image.append(rgb);
            rgb+=rgb_tri;
            x+=j;
            j++;
          }else{
             new_image.append(background);
            }
          }
          new_image.append("\n");
  }

}else if(deltax<0 && deltay>0){
  endpointX = x_tri-deltax;
  endpointY = y_tri+deltay;
```

Figura 17-TP2_06_DesenhaEmPPM-code(6)

```
}else if(deltax<0 && deltay>0){
  endpointX = x_tri-deltax;
  endpointY = y_tri+deltay;
  int j=0;
  String rgb=rgb_tri;

  if(endpointX<0 || endpointY<0 || startPointX<0 || startPointY<0 ||
  startPointX>=dimx || endpointX>dimx || startPointY>=dimy || endpointY>dimy  ){
    return null;
  }

  for(int y = 0; y<dimy; y++){
    for(int x = 0; x< dimx; x++){
      if (x == startPointX && y >= startPointY && y < endpointY )  {
            new_image.append(rgb);
            rgb+=rgb_tri;
            x+=j;
            j++;
            startPointX--;
          }else{
             new_image.append(background);
            }
          }
          new_image.append("\n");
  }

}else if(deltax>0 && deltay<0){
  endpointX = x_tri+deltax;
  endpointY = y_tri-deltay;
  startPointY=y_tri;

  int j=deltax-1;

  if(endpointX<0 || endpointY<0 || startPointX<0 || startPointY<0 ||
  startPointX>=dimx || endpointX>dimx || startPointY>=dimy || endpointY>dimy  ){
    return null;
  }

  for(int y = 0; y<dimy; y++){
```

Figura 18-TP2_06_DesenhaEmPPM-code(7)

```
    for(int y = 0; y<dimy; y++){
      for(int x = 0; x< dimx; x++){
        if (x == startPointX && y >= startPointY && y < endpointY )  {
          for(int i =0;i <deltax;i++){
            new_image.append(rgb_tri);
          }
              x+=j;
              j--;
              deltax--;
            }else{
               new_image.append(background);
              }
            }
            new_image.append("\n");;
    }

  }else if(deltax<0 && deltay<0){
    endpointX = x_tri-deltax;
    endpointY = y_tri-deltay;
    startPointY=y_tri;
    deltax=-deltax;
    int j=deltax-1;

    if(endpointX<0 || endpointY<0 || startPointX<0 || startPointY<0 ||
    startPointX>=dimx || endpointX>dimx || startPointY>=dimy || endpointY>dimy  ){
      return null;
    }

    for(int y = 0; y<dimy; y++){
      for(int x = 0; x< dimx; x++){
        if (x == startPointX && y >= startPointY && y < endpointY )  {
          for(int i =0;i <deltax;i++){
            new_image.append(rgb_tri);
          }
              x+=j;
              j--;
              deltax--;
              startPointX++;
            }else{
               new_image.append(background);
              }
```

Figura 19-TP2_06_DesenhaEmPPM-code(8)

```
              x+=j;
              j--;
              deltax--;
              startPointX++;
            }else{
               new_image.append(background);
              }
            }
            new_image.append("\n");;
    }
  }
   return new_image.toString();
   }

  public static void writeToPPMFile(String fileName, int xPixels, int yPixels,
  String content) {
  try (PrintWriter pw = new PrintWriter(new File(fileName))) {
  pw.println("P3"); // magic PPM P3 number
  pw.println(xPixels + " " + yPixels); // nXPixels nYPixels
  pw.println("255"); // max color value
  pw.println(content); // image content
  } catch (FileNotFoundException e) {
  e.printStackTrace();
  }
  }

  }
```

Figura 20-TP2_06_DesenhaEmPPM-code(9)

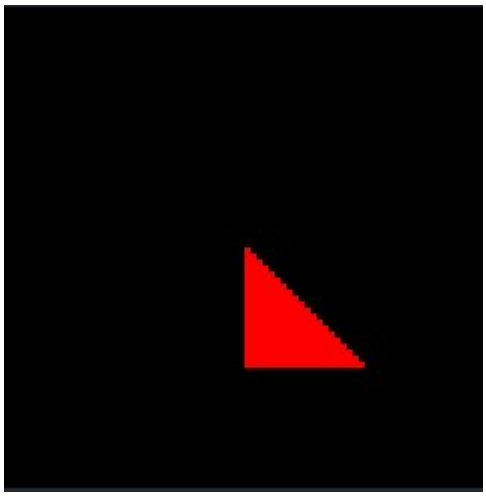


Figura 21-TP2_06_DesenhaEmPPM-output(1)

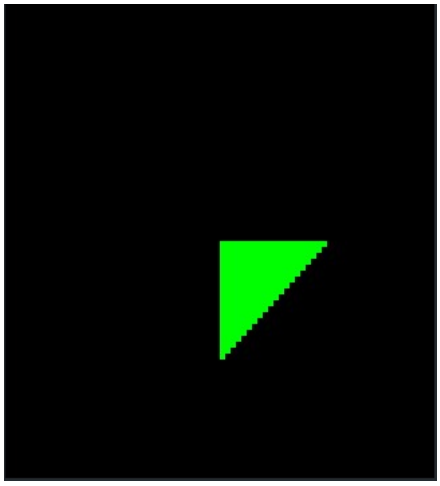


Figura 22-TP2_06_DesenhaEmPPM-output(2)

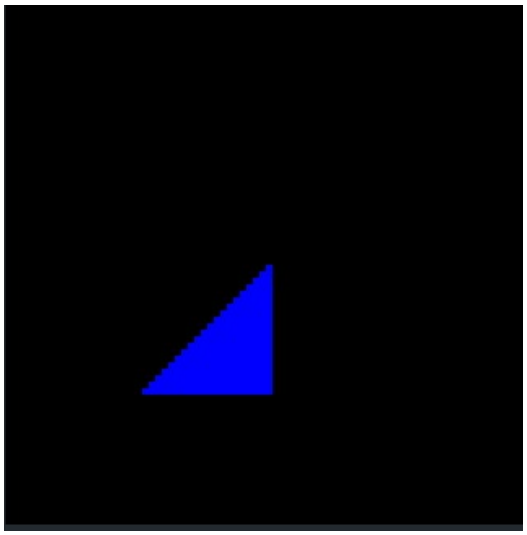


Figura 23-TP2_06_DesenhaEmPPM-output(3)

Figura 24-TP2_06_DesenhaEmPPM-output(4)

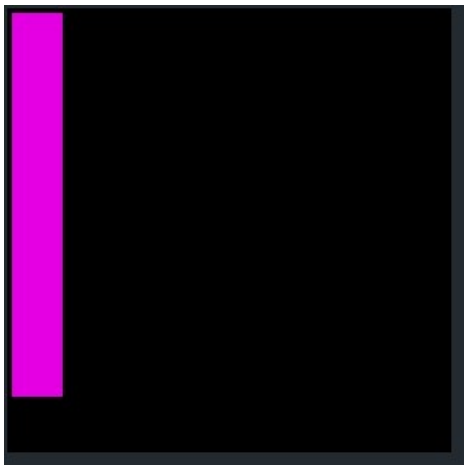


Figura 25-TP2_06_DesenhaEmPPM-output(5)

2.3. TP2_07_DesenhaBandeirasEmPPM

Conceber a classe TP2_07_DesenhaBandeirasEmPPM.java que permita gerar as seguintes bandeiras (somente com as formas e cores de fundo) em ficheiros com o nome do país e com dimensões proporcionais de (x 1.5, y 1): Portugal (2/5, 3/5), Espanha (1/4, 2/4, 1/4), Alemanha (1/3, 1/3, 1/3), Bélgica (1/3, 1/3, 1/3) e Filipinas (1/2 vertical e triângulo equilátero). As cores devem obtê-las da internet e podem ficar aproximadas. Cada bandeira deve ser gerada num método próprio com nome gerarBandeiraPais, onde para Portugal ficará gerarBandeiraPortugal, em que apenas deve receber a altura da bandeira (as restantes dimensões devem ser calculadas de forma proporcional). A altura das bandeiras deve ser pedida ao utilizador. Estes métodos devem chamar os métodos da classe TP2_06_DesenhaEmPPM.

```
import java.util.Scanner;

import java.lang.*;

import java.io.*;

public class TP2_07_DesenhaBandeirasEmPPM{
  public static void main(String[] args) {
    System.out.println("\nPrograma para bandeiras");
    Scanner keyboard = new Scanner(System.in);

    char bandeira='1';

    do{
    System.out.println("Indique qual bandeira vai querer desenhar:");

    bandeira='1';
    while(bandeira!='P' && bandeira!='E' && bandeira!='A' && bandeira!='B' && bandeira!='F' && bandeira!='Q'){
      System.out.println("Portugal (P)\nEspanha (E)\nAlemanha (A)\nBelgica (B)\nFilipinas (F)\nQuit (Q)\n");
        bandeira = keyboard.next().charAt(0);
        bandeira = Character.toUpperCase(bandeira);
    }
    if(bandeira=='Q'){
      System.out.println("End...");
      System.exit(0);
    }
    int altura =0;
    while(altura<=0){
    System.out.println("Indique a altura da bandeira");
     altura = keyboard.nextInt();
    }
    switch(bandeira){

      case 'P':
      gerarBandeiraPortugal(altura);
      break;

      case 'A':
      gerarBandeiraAlemanha(altura);
      break;
```

Figura 26-TP2_07_DesenhaBandeirasEmPPM-code(1)

```
      }
    }while(bandeira!='Q');

    keyboard.close();

  }


    public static void gerarBandeiraPortugal(int altura){

      float aux=Math.round(altura*1.5);
      int dimx= Math.round(aux);
      int dimy= altura;
      String rgb_ret="255 0 0 ";
      String image = null;
      int x_ret = 0;
      int y_ret = 0;
      int comprimento = (dimx*2)/5;
      int largura = altura;


      image= TP2_06_DesenhaEmPPM.drawRectangle(dimx,dimy,image,x_ret , y_ret, comprimento, largura, rgb_ret);

      rgb_ret="0 255 0 ";

      String Portugal= TP2_06_DesenhaEmPPM.drawRectangle(dimx,dimy,image,x_ret , y_ret, comprimento, largura, rgb_ret);

      writeToPPMFile("Portugal.ppm", dimx, dimy, Portugal);

      System.out.println("Done...");


    }
```

Figura 27-TP2_07_DesenhaBandeirasEmPPM-code(2)

```java
public static void gerarBandeiraEspanha(int altura){

  float aux=Math.round(altura*1.5);
  int dimx= Math.round(aux);
  int dimy= altura;
  String rgb_ret="255 0 0 ";
  String image = null;
  int x_ret = 0;
  int y_ret = (altura*1)/4;
  int comprimento = dimx;
  int largura = (altura*2)/4;


   image= TP2_06_DesenhaEmPPM.drawRectangle(dimx,dimy,image,x_ret , y_ret, comprimento, largura, rgb_ret);

   rgb_ret="241 191 0 ";



  String Espanha= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  writeToPPMFile("Espanha.ppm", dimx, dimy, Espanha);

  System.out.println("Done...");

}
```

Figura 28-TP2_07_DesenhaBandeirasEmPPM-code(3)

```java
public static void gerarBandeiraAlemanha(int altura){

  float aux=Math.round(altura*1.5);
  int dimx= Math.round(aux);
  int dimy= altura/2;
  String rgb_ret="0 0 0 ";
  String image = null;
  int x_ret = 0;
  int y_ret = (altura*1)/3;
  int comprimento = dimx;
  int largura = (y_ret/2)+1;

  image= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  rgb_ret="255 0 0 ";

  String Up= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  image = null;

  rgb_ret="255 255 0 ";

  image= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  rgb_ret="255 0 0 ";

  largura = (altura)/6;

  y_ret = 0;

  String Down= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  String Alemanha= Up+  Down;

  dimy=altura;

  writeToPPMFile("Alemanha.ppm", dimx, dimy, Alemanha);
```

Figura 29-TP2_07_DesenhaBandeirasEmPPM-code(4)

```java
public static void gerarBandeiraBelgica(int altura){

  float aux=Math.round(altura*1.5);
  int dimx= Math.round(aux);
  int dimy= altura;
  String black="0 0 0 ";
  String red = "255 0 0 ";
  String yellow = "255 255 0 ";
  String Belgica ="";

  for(int y=0;y<dimy;y++){
    for(int x=0;x<dimx;x++){
      if(x<=dimx/3){
        Belgica+=black;

      }else if (x>dimx/3 && x<(dimx*2)/3) {
        Belgica+=yellow;

      }else{
        Belgica+=red;
      }


    }
  }


  writeToPPMFile("Belgica.ppm", dimx, dimy, Belgica);

  System.out.println("Done...");
```

Figura 30-TP2_07_DesenhaBandeirasEmPPM-code(5)

```java
public static void gerarBandeiraFilipinas(int altura){

  float aux=Math.round(altura*1.5);
  int dimx= Math.round(aux);
  int dimy= altura/2;
  String rgb_ret="0 56 168 ";
  String image = null;
  int x_tri = 0;
  int y_tri = 0;
  int deltax = dimx/3;
  int deltay = dimy;
  int x_ret=0;
  int y_ret=0;
  int comprimento=0;
  int largura=0;


  image= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  String rgb_tri="255 255 255";

  String Up= TP2_06_DesenhaEmPPM.drawTriangle(dimx,dimy,image,x_tri,y_tri,deltax,deltay,rgb_tri);

  image = null;

  rgb_ret="206 17 38 ";

  image= TP2_06_DesenhaEmPPM.drawRectangle(dimx, dimy, image, x_ret, y_ret, comprimento, largura, rgb_ret);

  deltay= -deltay;

  String Down= TP2_06_DesenhaEmPPM.drawTriangle(dimx,dimy,image,x_tri,y_tri,deltax,deltay,rgb_tri);

  String Filipinas= Up+  Down;

  dimy=altura;

  writeToPPMFile("Filipinas.ppm", dimx, dimy, Filipinas);

  System.out.println("Done...");
```

Figura 31-TP2_07_DesenhaBandeirasEmPPM-code(6)

```
Programa para bandeiras
Indique qual bandeira vai querer desenhar:
Portugal (P)
Espanha (E)
Alemanha (A)
Belgica (B)
Filipinas (F)
Quit (Q)

p
Indique a altura da bandeira
40
Done...
Indique qual bandeira vai querer desenhar:
Portugal (P)
Espanha (E)
Alemanha (A)
Belgica (B)
Filipinas (F)
Quit (Q)
```

Figura 32-TP2_07_DesenhaBandeirasEmPPM-output(1)

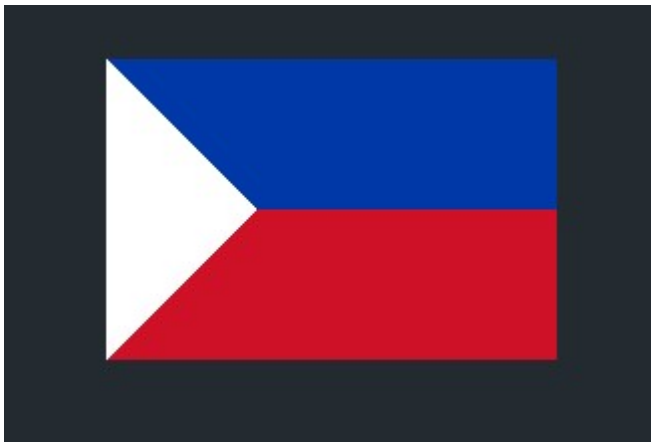


Figura 33-TP2_07_DesenhaBandeirasEmPPM-output(2)

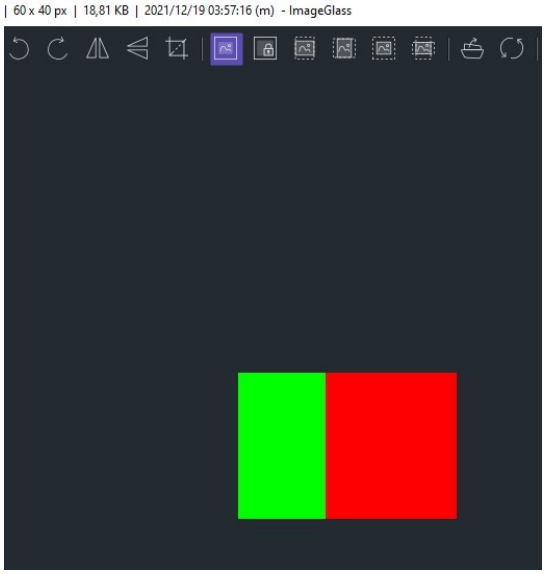


Figura 34-TP2_07_DesenhaBandeirasEmPPM-output(3)

Figura 35-TP2_07_DesenhaBandeirasEmPPM-output(4)

Figura 36-TP2_07_DesenhaBandeirasEmPPM-output(5)

Figura 37-TP2_07_DesenhaBandeirasEmPPM-output(6)

2.4. TP2_08_DesenhaDegradeEmPPM

Conceber a classe TP2_08_DesenhaDegradeEmPPM.java que permita gerar uma imagem com dimensões, nome, RGB do lado esquerdo e do lado direito, escolhidos pelo utilizador, e com degradê contínuo de cor entre o lado esquerdo e o lado direito. O main deve chamar o método desenhaDegrade, o qual recebendo os dados introduzidos deverá gerar em ficheiro a imagem pretendida. Sugestões: cada componente de RGB deve ser trabalhada em separado, e para cada uma, a diferença entre o valor inicial para o final e o número de colunas, que a imagem tem, define o passo de progressão do degradê da componente.

```
import java.io.*;

public class TP2_08_DesenhaDegradeEmPPM{

public static void main(String[] args) {

System.out.println("\nPrograma para fazer degrade entre duas cores escolhidas pelo utilizador");
Scanner keyboard = new Scanner(System.in);

System.out.println("Indique dados da imagem");
System.out.println("Nome: ");
String fileName = keyboard.nextLine();

System.out.println("Indique as dimensoes da imagem");
int dimx = keyboard.nextInt();
int dimy = keyboard.nextInt();

System.out.println("Indique a cor da esquerda");
int r_left = keyboard.nextInt();
int g_left = keyboard.nextInt();
int b_left = keyboard.nextInt();

System.out.println("Indique a cor da direita");
double r_right = keyboard.nextInt();
double g_right = keyboard.nextInt();
double b_right = keyboard.nextInt();

desenhaDegrade(fileName, dimx, dimy, r_left, g_left, b_left, r_right, g_right, b_right);

keyboard.close();
}


public static void  desenhaDegrade(String fileName, int dimx, int dimy, int r_left, int g_left, int b_left, double r_right, double g_right, double b_right){
  double r_double=0;
  double g_double=0;
  double b_double=0;
  long r=0;
  long g=0;
  long b=0;
  String degrade="";
  String rgb="";


  for(int y=0;y<dimy;y++){
    for(int x=0;x<dimx;x++){
      r_double = r_left + x * ((r_right - r_left) / (dimx - 1));
      g_double = g_left + x * ((g_right - g_left) / (dimx - 1));
      b_double = b_left + x * ((b_right - b_left) / (dimx - 1));

      r=Math.round(r_double);
      g=Math.round(g_double);
      b=Math.round(b_double);


      rgb= r + " " + g + " " + b+ " ";

      degrade= degrade+ rgb ;

    }
    degrade+="\n";
  }

  System.out.println("Done...");

  writeToPPMFile(String.format("%s.ppm",fileName),dimx,dimy,degrade);


}
```

Figura 38-TP2_08_DesenhaDegradeEmPPM - code

```
Programa para fazer degrade entre duas cores escolhidas pelo utilizador
Indique dados da imagem
Nome:
degrade
Indique as dimensoes da imagem
40
40
Indique a cor da esquerda
255
0
0
Indique a cor da direita
0
0
255
Done...
```

Figura 39-TP2_08_DesenhaDegradeEmPPM - output(1)

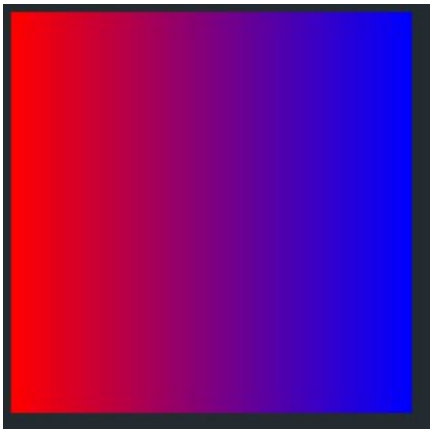


Figura 40-TP2_08_DesenhaDegradeEmPPM - output(2)

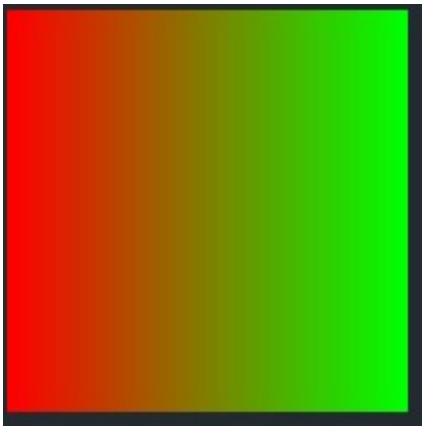


Figura 41-TP2_08_DesenhaDegradeEmPPM - output(3)

3. Arrays

3.1. TP2_09_MergeWithoutRepetitions

Conceber na classe TP2_09_MergeWithoutRepetitions o método int[] mergeWithoutRepetitions(int[] array1, int[] array2), o qual devolve um novo array com os valores únicos entre os dois arrays recebidos. Cada array pode conter valores repetidos e pode haver repetições entre arrays. Devolve null caso ambos os arrays forem null.

```
import java.io.*;

import java.util.Scanner;

public class TP2_09_MergeWithoutRepetitions{

  public static void main(String[] args) {
    int[] a = {1, 2, 3, 4, 8, 5, 7, 9, 6, 0};
    int[] b = {2,3,4,3,1,5,2,10};
    int[] r= mergeWithoutRepetitions(a,b);
    for (int i=0;i<r.length;i++){
             System.out.println(r[i]);
           }
  }

  public static int[] mergeWithoutRepetitions(int[] a, int[] b) {
    if(a==null && b==null){
      return null;
    }
    int [] c = merge(a,b);
    int [] r = remove(c);

    return r;
}


private static  int[] merge(int[] a, int[] b) {
    int[] c = new int[a.length + b.length];
    int j=b.length;
      for (j=0;j<b.length;j++){
        c[j]=b[j];
      }
      for (int i=0;i<a.length;i++){
      c[j]=a[i];
      j++;
    }
    return c;
}

private static  int[] remove(int[] c) {
    int len=c.length;
    int k=0;
    int[] iguais = new int[len];
    for (int i=0;i<len;i++){
        for (int j=i+1;j<len;j++){
            if (c[i] == c[j]) {
              iguais[k]=c[i];
              k++;
            }
          }
        }
```

Figura 42-TP2_09_MergeWithoutRepetitions-code(1)

```
    int x=0;
    int p=0;
    int[] f= new int [len];
    for (int i=0;i<len;i++){
      for (int j=0;j<k;j++){
        if(c[i]!=iguais[j]){
          x++;
        }
        if(x==k){
        f[p]=c[i];
        p++;
            }
        }
        x=0;
      }
      int [] r = new int [p];
      for (int i=0;i<r.length;i++){
        r[i]=f[i];
      }

    return r;
}

}
```

Figura 43-TP2_09_MergeWithoutRepetitions-code(2)

Figura 44-TP2_09_MergeWithoutRepetitions-output