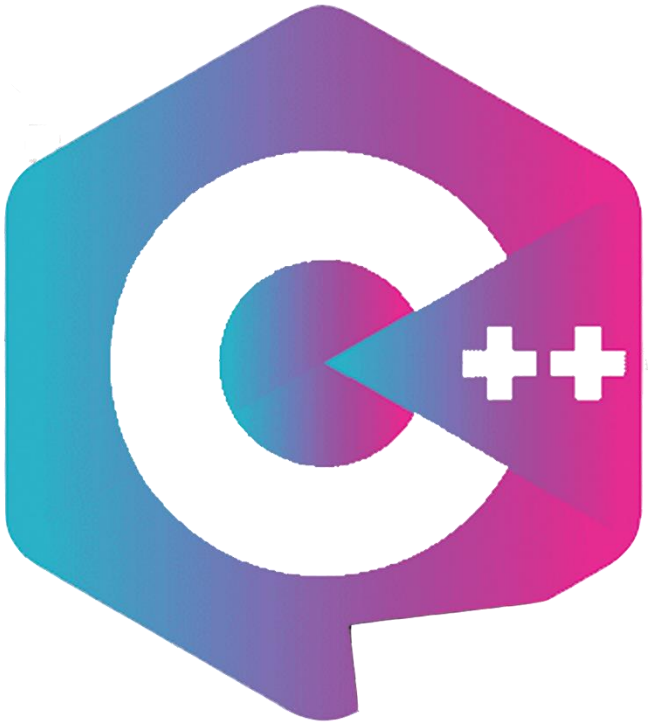
Report of the project

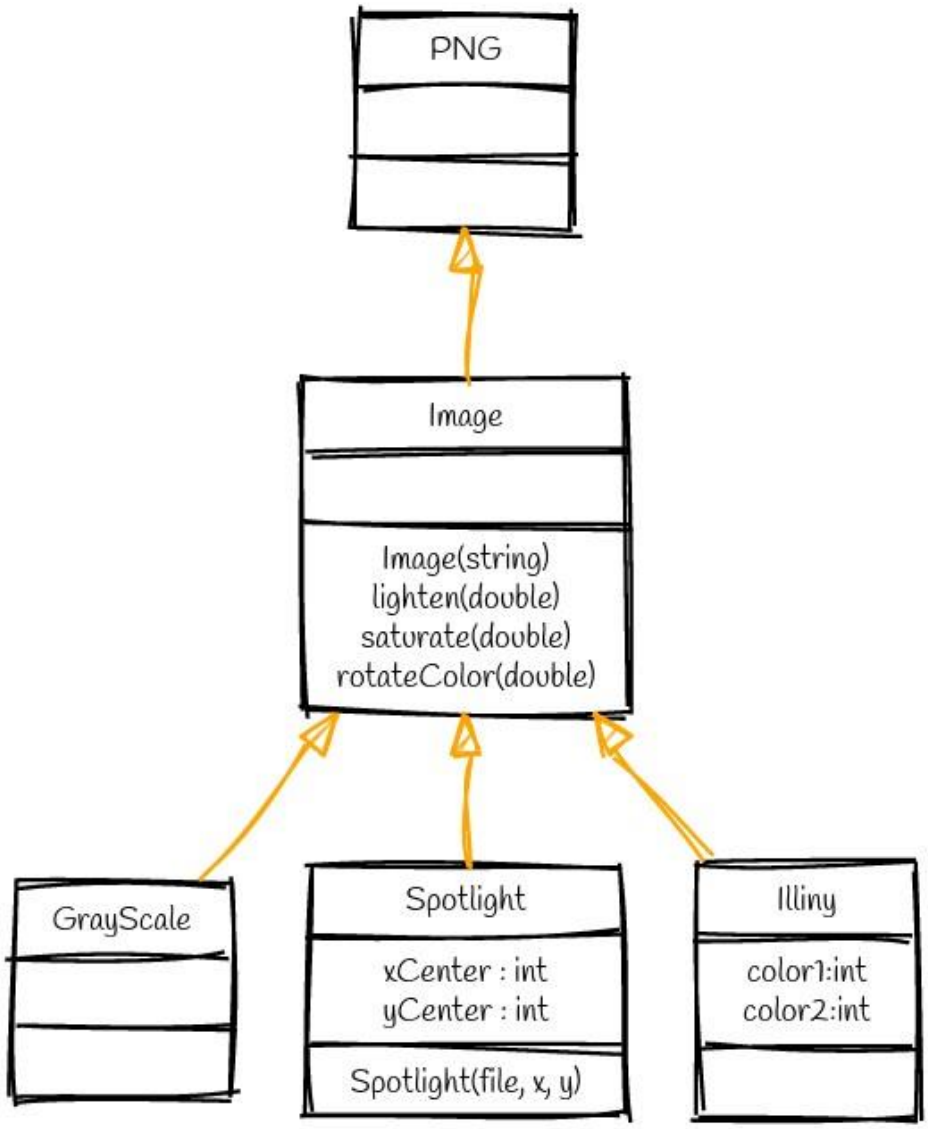


HSLA IMAGES

MADE BY :

MOHAMMED ELIAS LAKHMIRI

UML class diagram :



UML class diagram for the additional Images classes.

IMAGE CLASS :

Source Code

```
#ifndef IMAGE_H
#define IMAGE_H
#include "PNG.h"
class Image : public PNG
{
public:
    using PNG::PNG;
     //METHODE
     Image(string filename);
     void lighten(double amount=0.1);
     void saturate(double amount=0.1);
     void rotateColor(double angle);
};
```

IMAGE.H

IMAGE CLASS:

Source Code

LIGHTEN METHOD

```
#include "image.h"
Image::Image(string filename):PNG()
{
readFromFile(filename);
}
void Image::lighten(double amount){
     for(unsigned i=0 ;i<width();i++)
          for(unsigned j=0 ;j<height();j++){
               HSLAPixel &P=getPixel(i,j);
               P.l+=amount;
               P.l=(P.l>0) ? P.l:0;
               P.l=(P.l<=1)? P.l:1;
          }

}
```

IMAGE.CPP

SATURATE METHOD

```
void Image::saturate(double amount){
     for(unsigned i=0 ;i<width();i++)
          for(unsigned j=0 ;j<height();j++){
               HSLAPixel &P=getPixel(i,j);
               P.s+=amount;
               P.s=(P.s>0) ? P.s:0;
               P.s=(P.s<=1)? P.s:1;
          }
}
```

ROTATECOLOR METHOD

```
void Image::rotateColor(double angle){
    for(unsigned i=0 ;i<width();i++)
        for(unsigned j=0 ;j<height();j++){
            HSLAPixel &P=getPixel(i,j);
            P.h+=angle;
            while(P.h>360)
                P.h-=360;
            while(P.h<0)
                P.h+=360;
        }
}
```

RESULTS:

```
    Image I;
I.readFromFile("res/euromed_image.png");
I.lighten(0.3);
I.writeToFile("res/lighten.png");
```

MODIFYING the luminance of the image using lighten(0.3) :

INPUT:

INPUT:

RESULTS:

```
    Image I;
I.readFromFile("res/euromed_image.png");
I.saturate(0.5);
I.writeToFile("res/saturate.png");
```

Changing the saturation of the image using saturate(0.5)

INPUT:

OUTPUT:

RESULTS:

```
    Image I;
I.readFromFile("res/euromed_image.png");
I.rotateColor(200);
I.writeToFile("res/rotateColor.png");
```

Adding the value of an angle to the image using rotatecolor(200) :

INPUT:

OUTPUT:

GRAYSCALE CLASS:

Source Code

```
#ifndef GRAYSCALE_H
#define GRAYSCALE_H
#include "Image.h"

class Grayscale : public Image
    {
public:
   using Image::Image;
    using PNG::writeToFile;

    //METHODE
    Grayscale(string filename);
//    void saturate();
};
```

GRAYSCALE.H

```
#include "grayscale.h"
#include "image.h"
Grayscale::Grayscale(string filename):Image()
{
    readFromFile(filename);
saturate(-1);
}
```

GRAYSCALE.CPP

RESULTS:

Eliminating all the colors of the image :

INPUT:

OUTPUT:

ILLINI CLASS:

Source Code

```
#ifndef ILLINI_H
#define ILLINI_H
#include "image.h"
class Illini: public Image
{
public:
    using Image::Image;
    using PNG::writeToFile;
    int color1=11;
    int color2=216;
     //METHODE
     Illini(string filename,int color1=11,int color2=216);
//void saturate();
};
```

ILLINI.H

```
 #include "illini.h"
 #include "image.h"
 Illini::Illini(string filename,int color1,int color2):Image()
 {
     readFromFile(filename);
     for(unsigned x = 0; x < width() ; x++)
       for(unsigned y = 0; y < height(); y++)
       {
          //reference on the pixel
          HSLAPixel &P = getPixel(x, y);
          //modifiy the element of P
 if(P.h>11 && P.h<318)
 {
 int distance1=abs(P.h-color1);
 int distance2=abs(P.h-color2);
 if(distance1<distance2)
     P.h=color1;
 else P.h=color2;
 }
 else
     P.h=color1;
       }
 }
```

ILLINI.CPP

RESULTS:

```
Illini I("res/euromed_image.png",11,216);
    I.writeToFile("res/illini.png");
     return 0;
```

Replacing the hue of each pixel that are either the first or the second color using color1 = 11 (orange) and color2 = 216(blue).

INPUT:

OUTPUT:

SPOTLIGHT CLASS:

Source Code

```
#ifndef SPOTLIGHT_H
#define SPOTLIGHT_H
#include "image.h"
#include"PNG.h"

class Spotlight: public Image
{
public:
    using Image::Image;
    using PNG::writeToFile;
    int centerX;
    int centerY;
    Spotlight(string filename,int centerX,int centerY);
    void changeSpotPoint(int centerX,int centerY);
};
```

spotlight.H

```
#include "spotlight.h"
#include"image.h"
#include "PNG.h"
#include "math.h"
Spotlight::Spotlight(string filename,int centerX,int centerY):Image(){
    readFromFile(filename);

    for(unsigned x=0;x<width();x++)
        for(unsigned y=0;y<height();y++){
            //reference
            double distance = sqrt((x-centerX)*(x-centerX)+(y-centerY)*(y-centerY));
            HSLAPixel &P =getPixel(x,y);
            if(distance<160){

                P.l=abs(P.l-(distance)*0.005*P.l);
            }
            else {
               P.l=0.2*P.l;
            }
}
}
```

spotlight.cpp

RESULTS:

```
Spotlight I("res/euromed_image.png",300,420);
   I.writeToFile("res/Spotlight.png");
return 0;
```

Creating a spotlight centred at the point (300,420)

INPUT:

OUTPUT:

All Of The Above :

Tests from PROVIDED_TEST

Correct (PROVIDED_TEST, main.cpp:64) Image : lighten1

Correct (PROVIDED_TEST, main.cpp:77) Image lighten() does not lighten a pixel above 1.0

Correct (PROVIDED_TEST, main.cpp:87) Image darken(0.2) darkens pixels by 0.2

Correct (PROVIDED_TEST, main.cpp:96) Image darken(0.2) does not darken a pixel below 0.0

Correct (PROVIDED_TEST, main.cpp:105) Image saturate() saturates a pixels by 0.1

Correct (PROVIDED_TEST, main.cpp:114) Image rotateColor(double) rotates the color

Correct (PROVIDED_TEST, main.cpp:122) Image rotateColor(double) keeps the hue in the range [0, 360]

Correct (PROVIDED_TEST, main.cpp:134) Grayscale Image

Correct (PROVIDED_TEST, main.cpp:145) illini

Correct (PROVIDED_TEST, main.cpp:158) Pixels closest to blue become blue

Correct (PROVIDED_TEST, main.cpp:168) Pixels closest to orange become orange

Correct (PROVIDED_TEST, main.cpp:177) Hue wrap-arounds are correct (remember: h=359 is closer to orange than blue)

Correct (PROVIDED_TEST, main.cpp:186) Spotlight does not modify the center pixel

Correct (PROVIDED_TEST, main.cpp:193) Spotlight creates an 80% dark pixel >160 pixels away

Correct (PROVIDED_TEST, main.cpp:199) Spotlight is correct at 20 pixels away from center

Correct (PROVIDED_TEST, main.cpp:206) Spotlight is correct at 5 pixels away from center

Tests from main.cpp

Correct (PROVIDED_TEST, line 64) Image : lighten1

Correct (PROVIDED_TEST, line 77) Image lighten() does not lighten a pixel above 1.0

Correct (PROVIDED_TEST, line 87) Image darken(0.2) darkens pixels by 0.2

Correct (PROVIDED_TEST, line 96) Image darken(0.2) does not darken a pixel below 0.0

Correct (PROVIDED_TEST, line 105) Image saturate() saturates a pixels by 0.1

Correct (PROVIDED_TEST, line 114) Image rotateColor(double) rotates the color

Correct (PROVIDED_TEST, line 122) Image rotateColor(double) keeps the hue in the range [0, 360]

Correct (PROVIDED_TEST, line 134) Grayscale Image

Correct (PROVIDED_TEST, line 145) illini

Correct (PROVIDED_TEST, line 158) Pixels closest to blue become blue

Correct (PROVIDED_TEST, line 168) Pixels closest to orange become orange

Correct (PROVIDED_TEST, line 177) Hue wrap-arounds are correct (remember: h=359 is closer to orange than blue)

Correct (PROVIDED_TEST, line 186) Spotlight does not modify the center pixel

Correct (PROVIDED_TEST, line 193) Spotlight creates an 80% dark pixel >160 pixels away

Correct (PROVIDED_TEST, line 199) Spotlight is correct at 20 pixels away from center

Correct (PROVIDED_TEST, line 206) Spotlight is correct at 5 pixels away from center

Passed 32 of 32 tests. Congratulations!

UEMF
الجامعة الأورومتوسطية بفاس
EURO-MEDITERRANEAN UNIVERSITY OF FES
UNIVERSITÉ EURO-MÉDITERRANÉENNE DE FÈS

EIDIA
École d'Ingénierie Digitale et d'Intelligence Artificielle

THANK YOU