Practical 11: Poster Maker (Iteration)

In this practical you will be creating a poster made up of coloured tiles.

Task 1

Please download the NetBeans project called **posterMaker** accompanying this practical sheet on the KLE.

The above project contains two classes called `Application` and `Poster`.

You are not required to understand the code in the `Poster` class at this point. This is written for you and you should not need to modify it for this practical. All you need to know is that the objects created form `Poster` have two *accessible* attributes called `width` and `height`. These are both *constant* integers and are initialised to 84 and 59 respectively. These are the dimensions of your poster measured in *tiles* (coloured squares).

Objects created from `Poster` also have behaviour. The two methods that implement this behaviour are:

```
void setTileColour(int x, int y, float red, float green, float blue)
```

This method sets the colour of a poster tile at position (`x`, `y`). `x` defines the column of the tile and `y` defines the row of the tile. Note that the coordinates of the top left tile in the poster are (`0`,`0`). **What are the coordinates of the tiles in the remaining three corners of the poster?** A colour is defined by the three parameters `red`, `green` and `blue`. These can range from 0.0 to 1.0 and specify the amount of each corresponding colour component that will make up the tile colour.

```
void displayAndSave()
```

This method displays the poster in a window and saves it in an image file called **poster.gif**. This is an image file that you can open with a web browser or almost any image viewer application (for example, Windows Paint).

The `Application` class is an example using the `Poster` class. You should be able to understand all the code in this class implementation. In short, the program creates a `Poster` object called `myPoster`, paints a few tiles in it and then display's and saves the poster.

Please make sure you understand the `Application` class before proceeding to the next task.

Task 2

You are required to modify the `main` method in class `Application` in order to paint your poster the way you like.

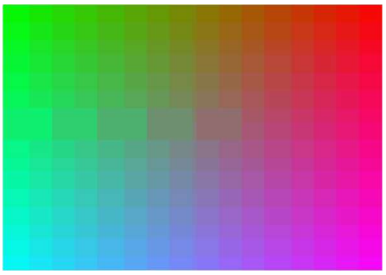
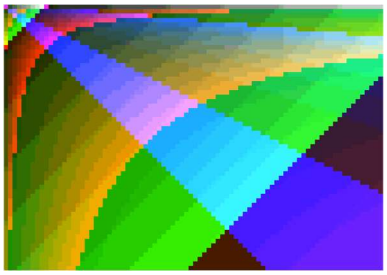
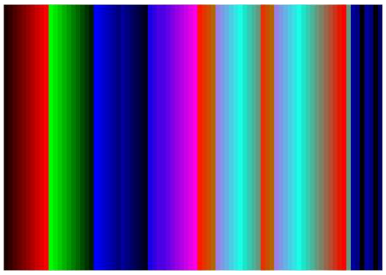
One way of painting your poster is to do it tile-by-tile for all 4,956 tiles but this will take a long time and effort to achieve! **This is not what you are expected to do in this practical.**

Instead, you are expected to use iteration to paint all the tiles in the poster. Here is a pseudo-code clue to get you started:

```
...
for every row of tiles in the poster
{
        for every tile in the row
        {
                paint tile some colour…
        }
}
...
```

Make sure that you do not stray out of the poster's dimensions. Also, make sure you do not use values for any of the colour components (`red`, `green` and `blue`) that fall out of the [0.0, 1.0] range. If you want to see what happens when you do there is only one way to find out: introduce the errors deliberately.

Here are some **good** examples of posters by previous students:

Here are some **not so good** examples of posters by previous students. This is because there has been a lot of tile-by-tile painting in these: