Faculty of Arts & Sciences
Department of Computer Science
CMPS 200—Introduction to Programming
Assignment 10 – Due Sunday Nov 23, 2014

Exercises

1. Write a method `processGraphics` that reads from a file a list of graphic shapes and draws them on the `DrawingPanel`. The input file must have on each line a record of one shape, such that the information for each corresponding record is formatted as follows:

a) For the shapes `Oval` and `Rect`, the corresponding record must follow the following format:

```
Shape r g b x y width height D-or-F.
```

where,

1. `r`,`g`,`b` are the values of the color with which the shape must be drawn.
2. `x y` are the (x,y) coordinates.
3. `width` and `height` are the width and height of the shape.
4. D or F is to either draw or fill the shape.

Examples: (1) `Oval 255 0 0 50 50 200 200 F;` (2) `Rect 112 12 122 30 20 100 100 D`

b) For the shape `Poly`, the corresponding record must follow the following format:

```
Shape r g b x1 y1 x2 y2 x3 y3 ... xN yN D-or-F
```

where, N is the number of points.
Example: `Poly 0 10 55 20 100 200 200 60 200 F.` Note that, the `Poly` record must at least contain one set of coordinates (i.e., `x1 y1`).

c) For the shape `Line`, the corresponding record must follow the following format:

```
Shape r g b x1 y1 x2 y2 D
```

Example: `Line 0 0 255 50 50 100 100 D.` Note that, the shape `Line` can only be drawn (i.e., D).

d) For the shape `Text`, the corresponding record must follow the following format:

```
Shape r g b x y text
```

where, `text` is the string to be drawn. Example: `Text 22 10 255 20 20 Go Lebanon Go`

Write a program `GraphicsProcessor.java` that can accept multiple graphic input files from the user as command line arguments and draws the graphics' records in the files using the method `processGraphics`. The program must then store all the shapes information in an output file such that the first record in the output file will consist of the width and height of the drawing panel.
A sample output file looks like this:

```
500 500
Oval 255 0 0 50 50 200 200 F
Rect 255 5 25 100 50 200 200 F
Rect 0 0 25 50 400 200 200 D
Poly 0 10 55 20 100 200 200 60 200 F
Line 0 0 255 50 50 100 100 D
Text 22 10 255 20 20 Go Lebanon Go
```

Example:

```
> java GraphicsProcessor 500 500 3 input1.txt input2.txt input3.txt out.txt
```

where `500 500` are the `DrawingPanel`'s width and height, respectively. 3 is the number of input files followed by the input files names, and then the output file name.

2. Given a file where each of its lines contains 5 integers (e.g., tutcar1.txt downloaded from Moodle):

a. The first line of the file contains only two integers representing the width and the height respectively.

b. On the following lines:

i. The first two integers represent the pixel's coordinate.

ii. The last three integers represent the red, green, blue values of the pixel.

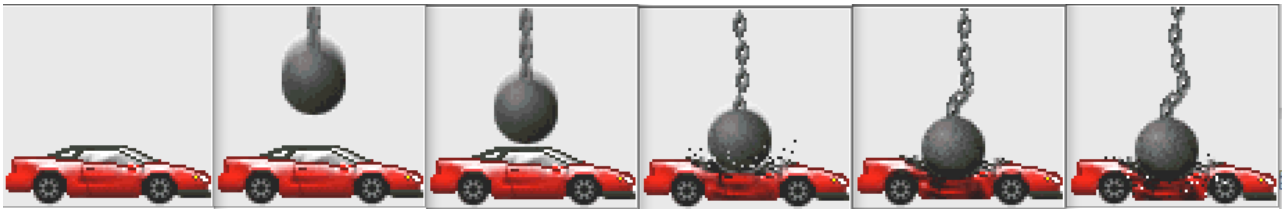
Write a method that takes as input a `DrawingPanel` and a file name that follows the format above. The method should draw all the pixels in the given panel.

Download from Moodle 6 files (tutcar1.txt, ..., tutcar6.txt). Create a method `playMovie` that displays all the images in sequence. After displaying an image, you may sleep for 10 milliseconds.

Hint. To draw a pixel at position (x,y) you may use `g.fillOval(x, y, 1, 1).`

Your program should produce the following output in the same drawing panel and the figures should overlap every 10ms (as if it is a video).

t=0	t=10ms	t=20ms	t=30ms	t=40ms	t=50ms



3. Write a program `LinuxDisplay.java` that takes four command line arguments. The first argument represents an input file name. The second argument may be the strings "top", "tail", or other. The third argument is a positive integer N. The fourth argument represents an output file name. For example:

```
> java LinuxDisplay input.txt tail 5 output.txt
```

If the second argument is equal to top (resp. tail), your program must generate the `output.txt` file containing the first (resp. last) N lines of the `input.txt` file. If the input file contains less than N lines, the output file should be the same as the input file. If the second argument is neither equal to "tail" nor "top", your program must generate the message: "`Invalid Argument`".

Submission Instructions

- As usual, submit your commented source code in a zip file named s*#*`_asst10_`*netid*, where *#* is your section number (between 1 and 12) and *netid* stands for your AUBnet user name.