Computer Science

COMPSCI 105

Assignment ONE

Due: **11:59 pm Friday 29th April 2016**
Worth: **4% of the final mark**

Introduction - The Shape Program

The program, as given, contains a canvas which allow users to add/remove shapes. You are going to use three different data structures to store rectangles, circles and arcs.

What you are to do

Firstly, become familiar with the template program supplied. You are required to add a few classes by yourself. Your assignment is divided into several stages for ease of completion. Please complete the assignment in order of the stages.

Stage 1: Title (1 mark)

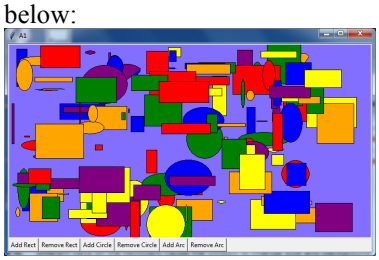
The title bar of the given program window displays "A1" only. Add your UPI to the title bar of the program, i.e. the title bar should display the string, "A1 – yourUPI", e.g., A5 – kng001

Stage 2: Random Region (5 mark)

You are required to create a new class which defines a random rectangular region for all the rectangles, circles and arc shapes that we will use in our program. For example, the size of the canvas is 700 x 400 and the maximum width of the rectangular region is 100 and the maximum height of the rectangular region is 80, and the objects created by the `RandRegion` class must be at a random position within the boundaries of the canvas. The width of the region must be less than 100 and the height of the region must be less than 80. For example, the following statement:

```
r = RandRegion(self.canvas_width, self.canvas_height)
```

will create a random rectangular region based on the width and height of the canvas. Some examples are given below:



Tasks:

- Complete the constructor to create a random rectangular region within the boundaries
- Complete the `get_coord()` method to return the x, y coordinates of the top left corner, the width and the height of the rectangular region
- Complete the `__repr__` method and the `__str__` method

Event Handling

As was mentioned earlier, a `Tkinter` application spends most of its time inside an event loop (entered via the `mainloop` method). Events can come from various sources, including key presses and mouse operations by the user, and redraw events from the window manager (indirectly caused by the user, in many cases). In `tkinter`, a callback is Python code that is called by `Tk` when something happens. For example, the Button widget provides a command callback which is called when the user clicks the button. You also use callbacks with event bindings. You can use any callable Python object as a callback. For example, you can use functions. To use a function object as a callback, pass it directly to `tkinter`. Here's a simple example:

```
from tkinter import *
from tkinter import messagebox

class Sample:
  def __init__(self, master):
    my_button = Button(master, text ="Hello", command = self.helloCallBack)
    my_button.pack(side = LEFT)
  def helloCallBack(self):
    messagebox.showinfo( "Hello Python", "Hello World")

root = Tk()
app = Sample(root)
root.mainloop()
```

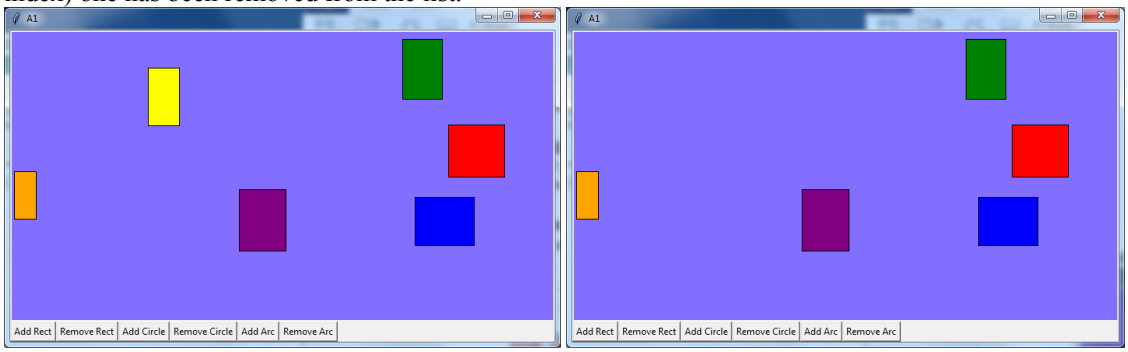
Run the above program and click in the button that appears. Each time you click, a hello world message is showed.

Stage 3: Rectangles (8 marks)

Tasks:

- Create a **Python list** in the constructor of your program.
- Create two buttons in the canvas. Pack them in the left bottom corner of your program
- Add a method to create a random size and position **rectangle** and add it to the **Python list** created above. The colour of the rectangles are based on the rainbow colours ('red', 'orange' 'yellow' 'green', 'blue' and 'purple')
- Add a method to remove a rectangle at a random index in the Python list.

The following screenshots shows that 6 rectangles have been added into a Python list and the yellow (a random index) one has been removed from the list.



Stage 4: Circles (8 marks)

Tasks:

- Create a **stack** in the constructor of your program.
- Create two buttons in the canvas. Pack them in the left bottom corner of your program
- Add a method to create a random size and position **circle** and add it to the **stack** created above. The colour of the circles are based on the rainbow colours ('red', 'orange' 'yellow' 'green', 'blue' and 'purple')
- Add a method to remove a circle from the top of the stack.

The following screenshots shows that 6 circles have been added into a stack and the purple one has been removed from the stack.

Stage 5: Arcs (8 marks)

Tasks:

- Create a **queue** in the constructor of your program.
- Create two buttons in the canvas. Pack them in the left bottom corner of your program
- Add a method to create a random size and position **arc** and add it to the **queue** created above. The colour of the arcs are based on the rainbow colours ('red', 'orange' 'yellow' 'green', 'blue' and 'purple')
- Add a method to remove an arc from the front of the queue.

The following screenshots shows that 6 arcs have been added into a queue and the red one has been removed from the queue.

Note: You can use the following properties to create an arc:

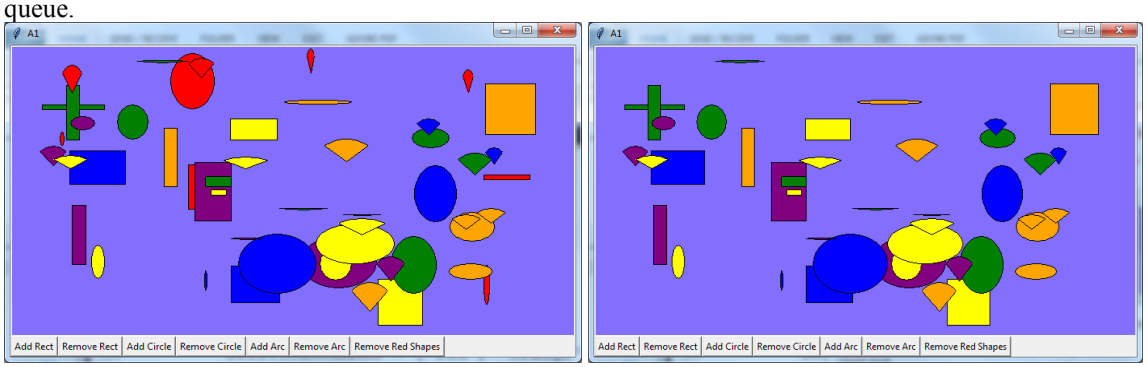
- start=45
- extent=90

Stage 6: Removing all RED shapes (10 marks)

Tasks:

- Create a button in the canvas. Pack it in the left bottom corner of your program.
- Add a method to remove all red rectangles in the Python list, all red circles in the stack and all red arcs in the queue.

The following screenshots shows that all red shapes have been removed from the Python list, the stack and the queue.



Submission

You may electronically submit your assignment through the Web Dropbox (https://adb.auckland.ac.nz/) at any time from the first submission date up until the final date. You can make more than one submission. However, every submission that you make replaces your previous submission. Submit ALL your files in every submission. Only your very latest submission will be marked. Please double check that you have included all the files required to run your program.

No marks will be awarded if your program does not compile and run. You are to electronically submit all the following files:

1. **A1.py**
2. **RandRegion.py**
3. **Stack.py**
4. **Queue.py**

DO NOT SUBMIT SOMEONE ELSE'S WORK:

- The work done on this assignment must be your own work. Think carefully about any problems you come across, and try to solve them yourself before you ask anyone for help.
- Under no circumstances should you take or pay for an electronic copy of someone else's work. This will be penalized heavily.
- Under no circumstances should you give a copy of your work to someone else
- The Computer Science department uses copy detection tools on the files you submit. If you copy from someone else, or allow someone else to copy from you, this copying will be detected and disciplinary action will be taken.
- To ensure you are not identified as cheating you should follow these points:
    - Always do individual assignments by yourself.
    - Never give another person your code.
    - Never put your code in a public place (e.g., forum, your web site).
    - Never leave your computer unattended. You are responsible for the security of your account.
    - Ensure you always remove your USB flash drive from the computer before you log off.