Laboratory 09 —tkinter, canvas widgets
Drawing patterns using nested for loops

TKINTER

Tk is a robust and platform independent windowing toolkit and is available to Python programmers through the tkinter package. Tk provides the definitions of many "widgets" (labels, buttons, text boxes, etc.) – the components of a GUI.

Tkinter is not the only GUI-programming toolkit for Python. It is however the most commonly used one.

Canvas
The canvas is a general purpose widget, which is typically used to display and edit graphs and other drawings.

Each pixel in the Canvas area has an x position (across the canvas) and a y position (down the canvas). Position (0, 0) is the top left corner of the canvas.

What can be done in the Canvas window?

`create_line``(x0, y0, x1, y1, `*`options`*`)`
The line is defined by 2 points (x0, y0) and (x1, y1). For example:

```
my_canvas.create_line(200,250, 150,175, fill = "purple", width = 3)
```

`create_rectangle``(x0, y0, x1, y1, `*`options`*`)`
The rectangle is defined by 2 points: (x0, y0) the top left position and (x1, y1) the bottom right position. For example:

```
my_canvas.create_rectangle(50,100, 150,200, fill = "blue", outline ="red")
```

`create_oval``(x0, y0, x1, y1, `*`options`*`)`
The oval is defined by 2 points: (x0, y0) the top left position of the bounding rectangle and (x1, y1) the bottom right position of the bounding rectangle. For example:

```
my_canvas.create_oval(50,100, 150,200, fill = "pink", outline ="black")
```

`create_polygon``(coordinates of points, `*`options`*`)`
The polygon is defined by a series of points: (x0, y0, x1, y1, …. xn, yn). For example:

```
my_canvas.create_polygon(50,100, 125,25, 200,100, fill = "blue", outline = "black")
```

`create_text``(coordinates of position, `*`options`*`)`
Draws text in the canvas. For example:

```
my_font = ("Courier", 12, "bold")
my_canvas.create_text(50,100, text = "Hello", fill = "green",font = my_font)
```

By default, the text is centred on the position. You can override this with the anchor option. For example, if the position specified is the upper left corner, set the anchor to NW.

```
my_canvas.create_text(50,100, text = "Upper left corner", anchor = NW)
```

EXERCISES

IMPORTANT: for each of your programs you need to add a docstring at the top of the program. The docstring should contain your name, your username, date and a short description of the program.

EXERCISE 9.1 – DRAWING PATTERNS USING NESTED FOR LOOPS.

The Skeleton code for the Lab09Program1.py program is shown below. Complete the `print_ox_xo_pattern()` function in the program.

```
def main():
    print_ox_xo_pattern(5, 4)
    print()

main()
```

- **`print_ox_xo_pattern(number_of_rows, number_of_columns)`**

  Parameters: number_of_rows - the number of rows to be printed
  number_of_columns - the number of columns to be printed

  Prints: a specific pattern shown as below:

The function takes two integers as parameters and prints a pattern of alternating 'x' and 'o' characters. Even rows will start with an 'o' and odd rows will start with an 'x'. For example, the following statement:

```
  print_ox_xo_pattern(5, 4)
```

prints:

```
oxox
xoxo
oxox
xoxo
oxox
```

EXERCISE 9.2 – DRAWING PATTERNS USING NESTED FOR LOOPS.

The Skeleton code for the Lab09Program2.py program is shown below.
Complete the `print_ mirrored_right_angle_triangle()` function in the program.

```
def main():
    print_mirrored_right_angle_triangle(4, '*')
    print_mirrored_right_angle_triangle(4, 1)
main()
```

- **`print_mirrored_right_angle_triangle (number_of_rows, style)`**

  Parameters: number_of_rows - the number of rows to be printed
  style - the specified character to be printed. If style is an integer, the pattern is made up of an ascending sequence of numbers starting with the number in the `style` parameter. You can use the `str()` function and `isdigit()` method to check whether the style parameter is an integer. If style is not an integer then use style as the character to print.

  Prints: a specific right-angle triangle shown as below:

The function takes an integer and a style as parameters and prints a specific right angle triangle using either numbers or any specific characters. For example: the following statement:

```
  print_mirrored_right_angle_triangle(4, '*')
```

prints:

```
    *
   **
  ***
 ****
```

and the following statement:

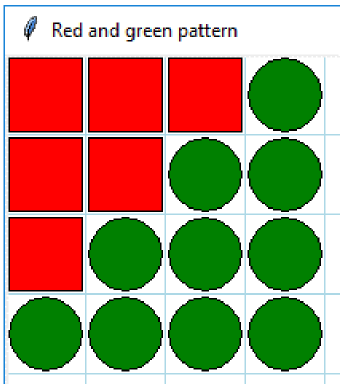
```
  print_mirrored_right_angle_triangle(4, 1)
```

prints:

```
   1
  12
 123
1234
```

EXERCISE 9.3 – DRAWING A PATTERN IN THE CANVAS WIDGET USING NESTED FOR LOOPS

Complete the code in the draw_pattern_in_canvas(a_canvas, number_of_rows) function in the Lab09Program3.py file so that the pattern in the screenshot below is drawn. The gridlines are 50 pixels apart and have already been drawn for you but they will be partially covered once you have drawn the correct pattern. The squares are red and the circles are green. **Note: You must replace the title "Red and green pattern" with your UPI.**



- **`draw_pattern_in_canvas(a_canvas, number_of_rows)`**

  Parameters: a_canvas – the `Canvas` object
  number_of_rows - the number of rows to be printed. Note that the number of columns in each row is equal to the number of rows.

  Prints: a specific pattern shown as above.

  Note: Take a screenshot of your image and save it so that you can upload it to CodeRunner.