Lecture 26 - Using the Canvas object to draw grids of shapes
The program skeleton for the examples and exercises

All the programs in this lecture have the following code skeleton. The `draw_shapes()` function is different for each exercise.

```python
from tkinter import *
def draw_shapes(a_canvas):
    #The function code here is different for each slide
def main():
    window = Tk()
    window.title("My first Canvas")
    window.geometry("400x300+10+20")
a_canvas = Canvas(window)
a_canvas.config(background="pink")
a_canvas.pack(fill=BOTH, expand = True)
draw_shapes(a_canvas)
window.mainloop()
main()
```
To create rows and columns of shapes we need to use nested loops.

Set up all the variables needed for the nested loop

**for ... in loop** which dictates how many rows:
Set everything up ready for drawing the row

**for ... in loop** which handles one single row:
draw a single shape
change the x value to move along the row

change the y value ready for the next row down

Size of each square is 50 pixels by 50 pixels. There are five rows with seven squares along each row.
Drawing a grid of black unfilled boxes:

```python
def draw_shapes(a_canvas):
    number_along_row = 7
    number_of_rows = 5
    size = 50
    left_hand_side = 20
    y_down = 30

    for row in range(number_of_rows):
        x_left = left_hand_side

        for col in range(number_along_row):
            rect = (x_left + 2, y_down + 2, x_left + size - 2, y_down + size - 2)
            a_canvas.create_rectangle(rect)
            x_left = x_left + size

        y_down = y_down + size
```
def draw_shapes(a_canvas):
    number_along_row = 7
    number_of_rows = 5
    left_hand_side = 20
    y_down = 30
    size = 40
    first_in_row_filled = True
    for row in range(number_of_rows):
        x_left = left_hand_side
        is_filled = first_in_row_filled
        for col in range(number_along_row):
            rect = (x_left, y_down, x_left + size, y_down + size)
            if is_filled:
                a_canvas.create_rectangle(rect, fill="blue")
            else:
                a_canvas.create_rectangle(rect)
            x_left = x_left + size
            is_filled = not is_filled
        y_down = y_down + size
    first_in_row_filled = not first_in_row_filled
def draw_shapes(a_canvas):
    number_of_rows = 5
    left_hand_side = 20
    y_down = 30
    size = 40
    first_is_circle = True
    for number_to_do in range(number_of_rows, 0, -1):
        x_left = left_hand_side
        is_circle = first_is_circle
        for col in range(number_to_do):
            rect = (x_left + 2, y_down + 2, x_left + size - 2,
                    y_down + size - 2)

            if is_circle:
                a_canvas.create_oval(rect)
            else:
                a_canvas.create_rectangle(rect)

        x_left = x_left + size
        is_circle = not is_circle

    y_down = y_down + size
    first_is_circle = not first_is_circle
def draw_shapes(a_canvas):
    number_of_rows = 6
    size = 30
    y_down = 0
    left_hand_side = size

    for row_number in range(1, number_of_rows + 1):
        x_left = left_hand_side

        for col in range(row_number):
            rect = (x_left + 2, y_down + 2, x_left + size - 2,
                    y_down + size - 2)
            a_canvas.create_oval(rect, fill="blue")
            x_left = x_left + size * 2

        y_down = y_down + size

Nested Shapes - Exercise 1

Draw the canvas (the gridlines are of size 30 pixels)
def draw_shapes(a_canvas):
    number_of_rows = 5
    left_hand_side = 0
    y_down = 0
    size = 50
    first_is_circle = True

    for number_to_do in range(1, number_of_rows + 1):
        x_left = left_hand_side
        is_circle = first_is_circle
        for col in range(number_to_do):
            rect = (x_left + 2, y_down + 2, x_left + size - 2, y_down + size - 2)

            if is_circle:
                a_canvas.create_oval(rect, fill="blue")
            else:
                a_canvas.create_rectangle(rect)
            x_left = x_left + size * 2
        is_circle = not is_circle
        y_down = y_down + size
    first_is_circle = not first_is_circle
def draw_shapes(a_canvas):
    number_across = 11
    size = 30
    y_down = 0
    x_start_left = 0
    is_square = True
    while number_across > 0:
        x_left = x_start_left
        for col in range(number_across):
            rect = (x_left + 2, y_down + 2, x_left + size - 2, y_down + size - 2)
            if is_square:
                a_canvas.create_rectangle(rect)
            else:
                a_canvas.create_oval(rect)
            x_left = x_left + size
        number_across = number_across - 2
        x_start_left = x_start_left + size
        y_down = y_down + size
        is_square = not is_square

Nested Shapes Exercise 3

Draw the canvas
(the gridlines are of size 30 pixels)