Chapter 4 deals with modifying picture pixels in a rectanglar portion using *nested for-loops*, i.e., a for-loop inside of a for-loop. We can use the `range` function to create a sequence of x (or y) coordinates to loop through. The syntax of: `range([start,] end, [, step])`, where **[ ] are used to denote optional parameters.** The optional parameter `start` has a default value of 0, and `step`'s default is 1. Some examples:

- `range(5)` generates the sequence of values: `0, 1, 2, 3, 4`
- `range(2,7)` generates the sequence of values: `2, 3, 4, 5, 6`
- `range(10,2,-1)` generates the sequence of values: `10, 9, 8, 7, 6, 5, 4, 3`

For example:

```
for count in range(1,7,2):
  print count
print "Done"
```

```
1
3
5
Done
```

1. What range function calls would generate each of the following sequences?

a) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

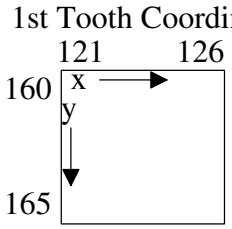
b) 9, 10, 11, 12, 13, 14

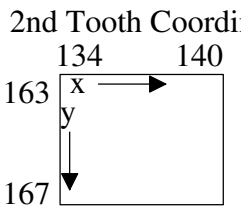
c) 10, 20, 30, 40, 50, 60, 70, 80, 90, 100

d) 15, 12, 9, 6, 3, 0

Suppose we wanted to blacken two teeth in `barbara.jpg` using nested for-loops instead of the addRectFilled calls of:

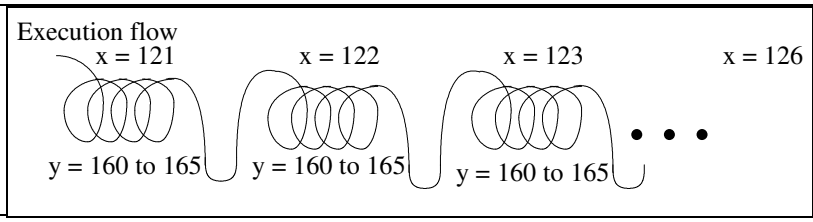
```
addRectFilled(pict, 121, 160, 6, 6, black)
addRectFilled(pict, 134, 163, 6, 5, black)
```





The nested for-loops needed to blacken the 1st tooth are:.

```
pict = makePicture(fileName)
for x in range(121,127):
  for y in range(160,166):
    px = getPixel(pict, x, y)
    setColor(px, black)
```



2. Write the nested for-loops needed to blacken the 2nd tooth.

3. Using nested for-loops, complete the following function that behaves like the `addRectFilled` function.

```
def myAddRectFilled(picture, startX, startY, width, height, color = black):
```

4. How would you call the above `myAddRectFilled` function to blacken two teeth in `barbara.jpg`?

```
pict = makePicture(fileName)  # Assume barbara.jpg selected
myAddRectFilled(                                                    )
myAddRectFilled(                                                    )
```

5. Using for-loops, complete the following function that behaves like the `addRect` function.

```
def myAddRect(picture, startX, startY, width, height, color = black):
```

6. In the remaining time, let's think about how we might do each of the following:

- mirror a picture horizontally
- create a collage, i.e., several pictures merged into a single picture
- rotate a picture 90° counter-clockwise
- scale a picture to make it smaller
- scale a picture to make it larger