Objectives for class 9

--- Chapter 5 ---

5.2 To control a loop with the user’s confirmation and a sentinel value (§5.5).

5.3 To develop loops following the loop design strategy (§5.4).

5.4 To use for loops to implement counter-controlled loops (§5.6).

5.6 To implement program control with break and continue (§5.10).

Ending a Loop with an input(sentinel) value

- Use an **input value** to signify the **end of the loop**

- When to use?
  - Number of iterations **not predetermined**
  - For example, stop reading input if user enters 0

Example: read and calculate the sum of an unspecified number of integers until zero

System Design

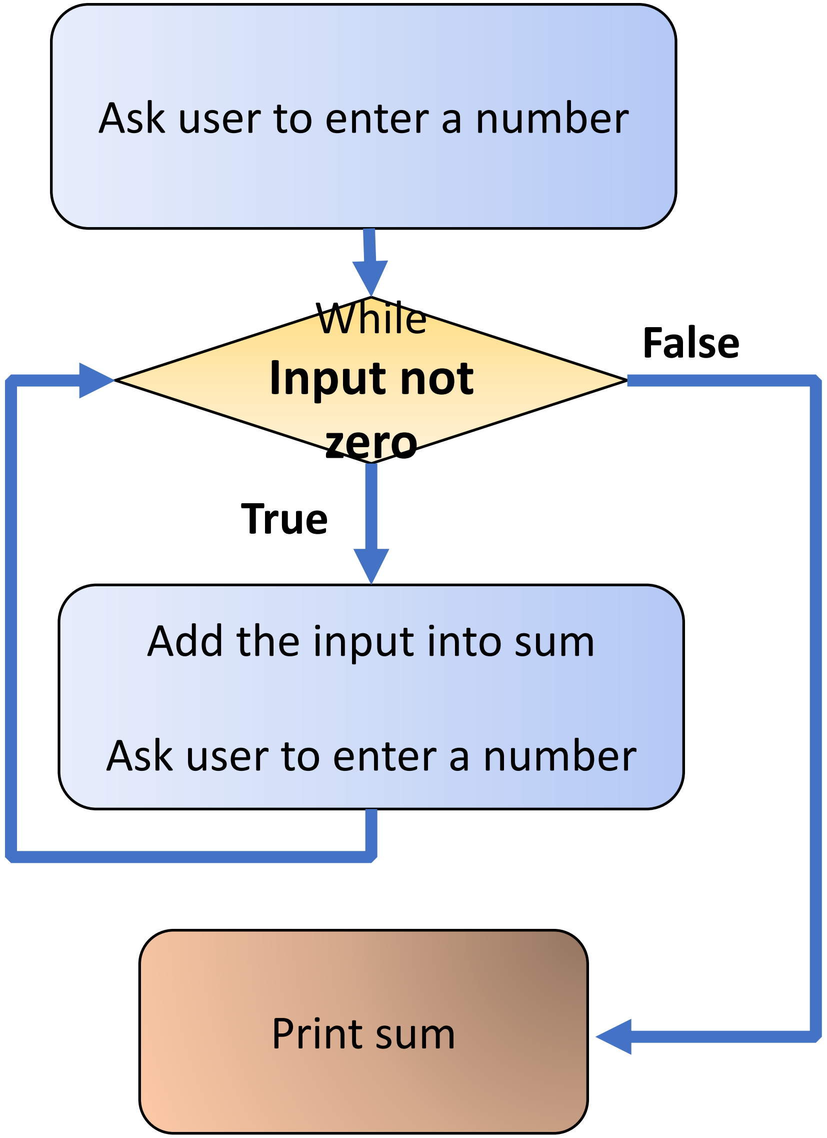
Step 1: Ask user to enter a number

While input != 0 do step 2 and 3

Step 2: Add the input into sum

Step 3: Ask user to enter a number

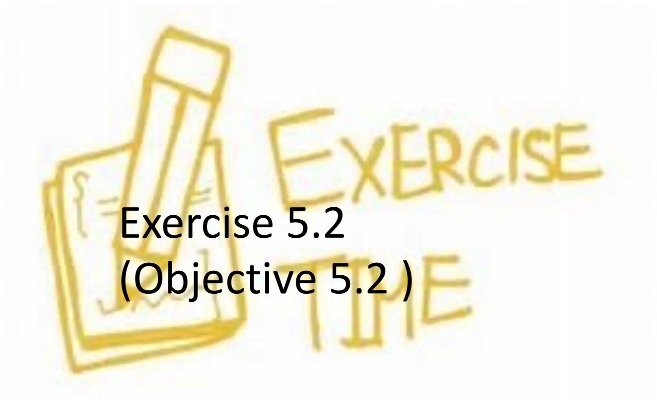
Step 4: print sum



Implementation

```
data = int(input("Enter an integer (the input exits " +
    "if the input is 0): "))


# Keep reading data until the input is 0
sum = 0
while data != 0:
    sum += data
    data = int(input("Enter an integer (the input exits " +
        "if the input is 0): "))


print("The sum is", sum)
```

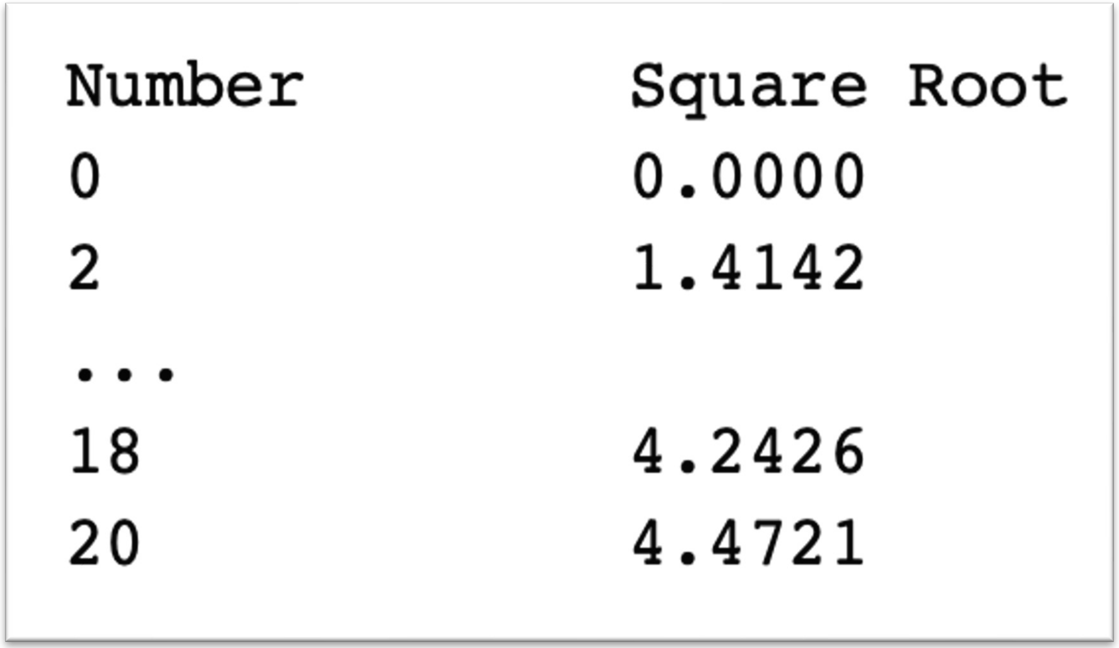
Practice



Exercise 5.2
(Objective 5.2)

Loop Design Strategy – Let us try a sqrt table problem

Write a program that prints a table below



```
Number           Square Root
0                0.0000
2                1.4142
...
18               4.2426
20               4.4721
```

- Numbers from 0, 2, 4 to 20
- math.sqrt()
- Round to 4 decimal places

Step 1 – Identify statements/actions to be repeated

- **Print** a number and corresponding square root

0	0.0000
2	1.4142
4	2.0000
6	2.4495
8	2.8284
10	3.1623
12	3.4641
14	3.7417
16	4.0000
18	4.2426
20	4.4721

Display the table row by row

Step 2 – Wrap the repeated statements/action in a loop.

While **True**

- **Print** a number and corresponding square root

```
0       0.0000
2       1.4142
4       2.0000
6       2.4495
8       2.8284
10      3.1623
12      3.4641
14      3.7417
16      4.0000
18      4.2426
20      4.4721
```

Step 3 – Control the loop

Condition added

```
Set number as 0
While number <=20,
    • Print the number and its square root
    • Increase number by 2
```

Control the condition and make it false eventually

```
0       0.0000
2       1.4142
4       2.0000
6       2.4495
8       2.8284
10      3.1623
12      3.4641
14      3.7417
16      4.0000
18      4.2426
20      4.4721
```

System Design

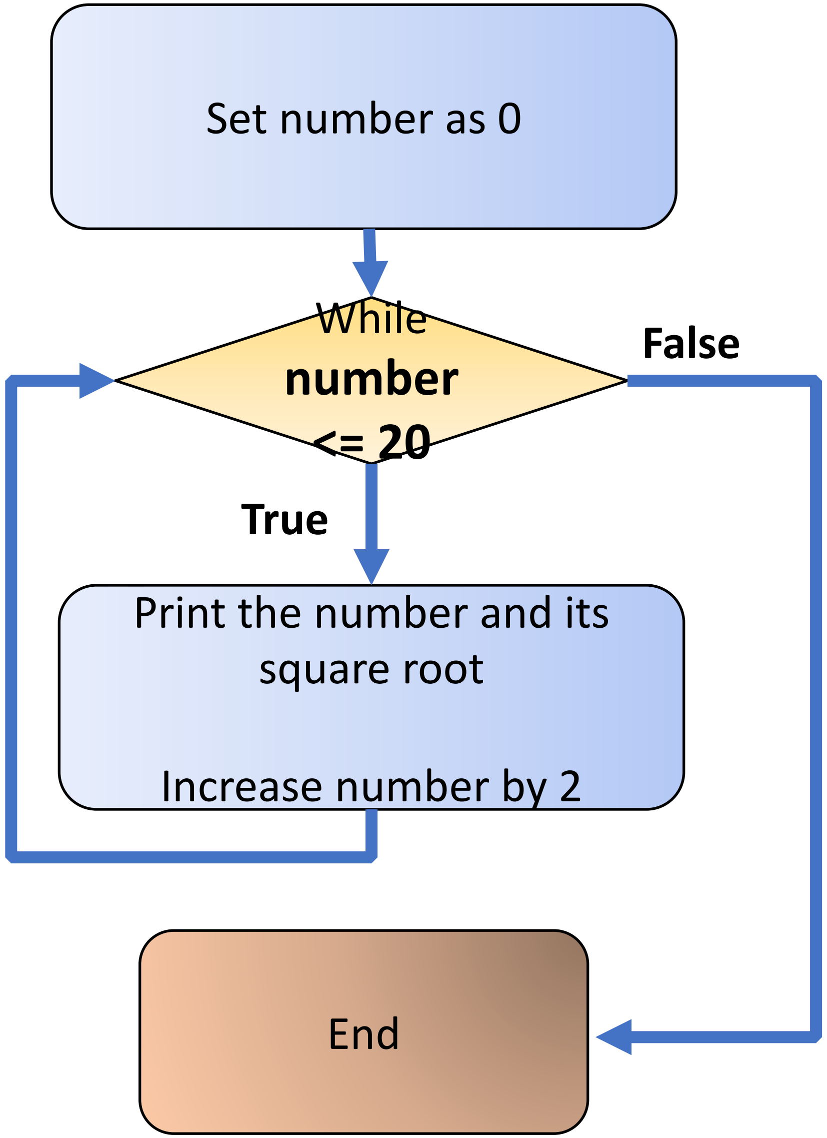
Step 1: Set number as 0

Step 2: print header

While number<=20 do step 3 and 4

Step 3: Print the number and its square root

Step 4: increase number by 2

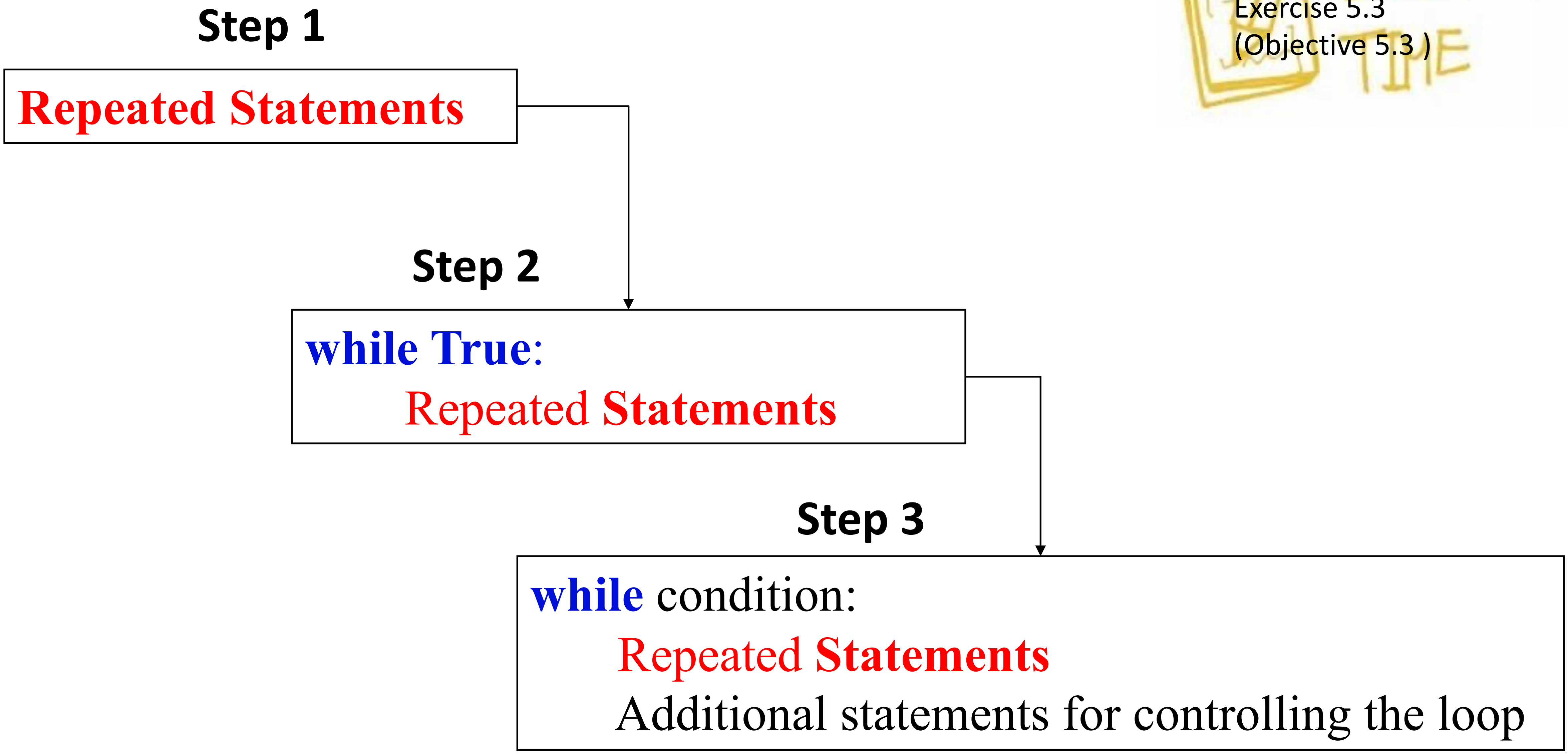


Implementation

```
import math

number = 0
print('Number\tSquare Root’)

# Keep printing row until number is not <= 20
while number <= 20:
    print(str(number)+'\t'+str(round(math.sqrt(number),4)))

    number=number+2
```

Loop Design Strategy – Three Steps

Exercise 5.3
(Objective 5.3)



Step 1

```
Repeated Statements
```

Step 2

```
while True:
    Repeated Statements
```

Step 3

```
while condition:
    Repeated Statements
    Additional statements for controlling the loop
```

Use **For** Loops When You Know Number Of Repetitions

while loop (counter-controlled)

```
count=0
while count<100:
    print("Programming is fun")
    count=count+1
```

Generate a list of numbers from 0 to **100**

for loop

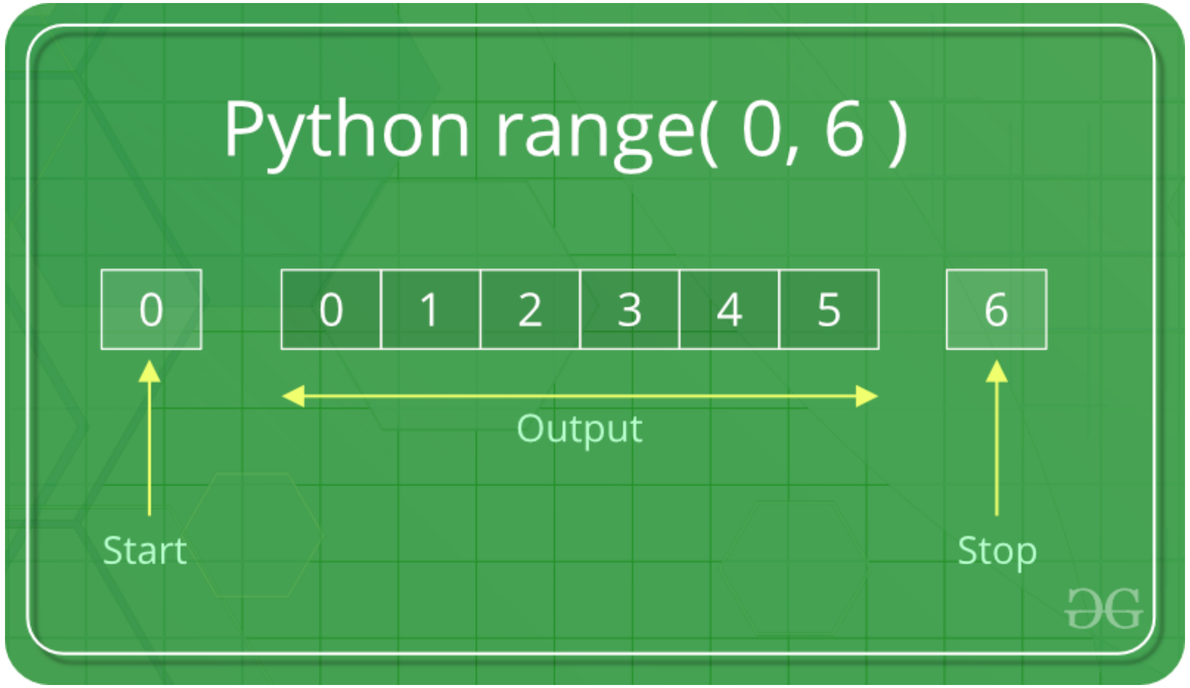
```
for i in range(0,100):
    print("Programming is fun")
```

How to use `range` function in **for** loops? Three Ways

- `range()` generates a series of numbers within a given range.

- Arguments in `()` decides the range
  - `range(start)`
  - `range(stop)`
  - `range(start,stop,end)`

range(start, stop)



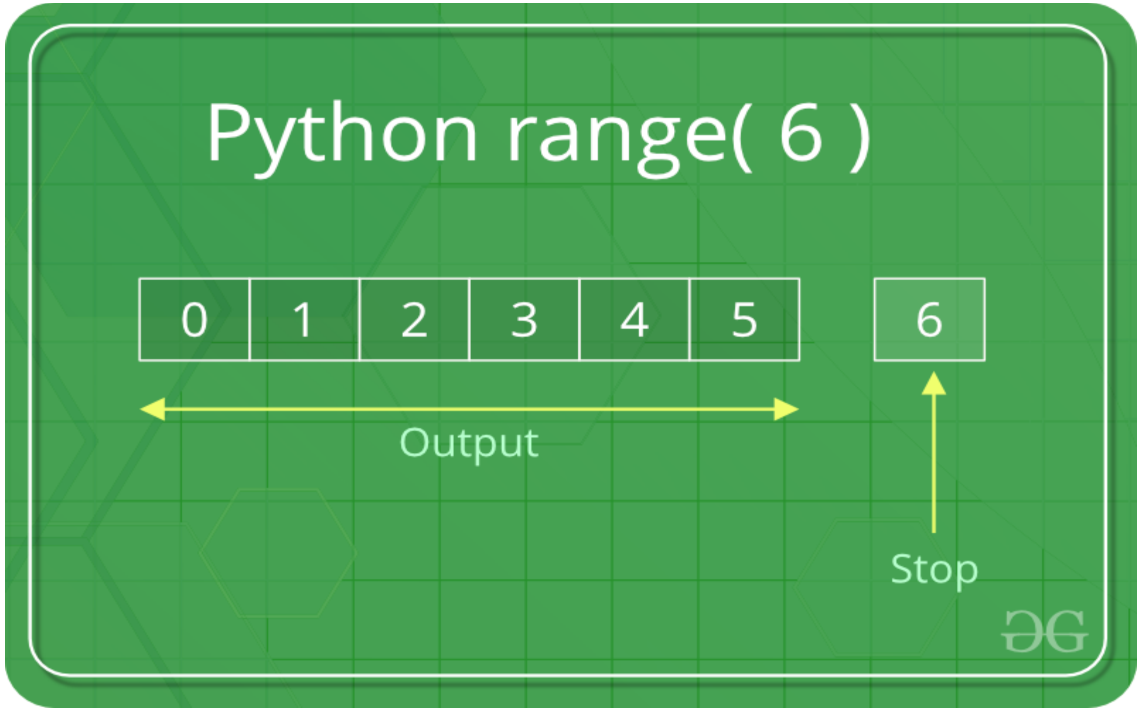
`stop` is exclusive!!

```python
# print number up to 6
for i in range(0, 6):
    print(i, end=" ")
print()

# print number from 3 to 6
for i in range(3, 6):
    print(i, end=" ")
```

Output:

```
0 1 2 3 4 5
3 4 5
```

range(stop)



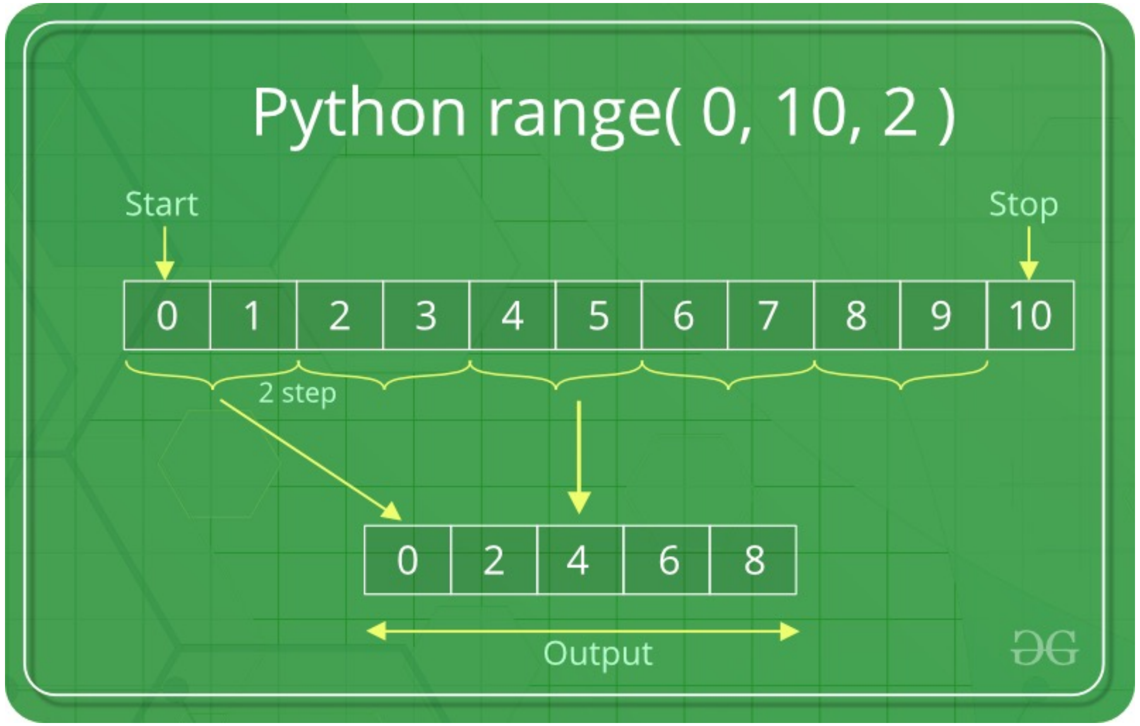
```python
# print number up to 6
for i in range(6):
    print(i, end=" ")
print()

# print number from 3 to 6
for i in range(3):
    print(i, end=" ")
```

- `start` is 0 in default!!
- `stop` is exclusive!!
- Same as `range(0,stop)`

Output:

```
0 1 2 3 4 5
0 1 2
```

range(start,stop,step)



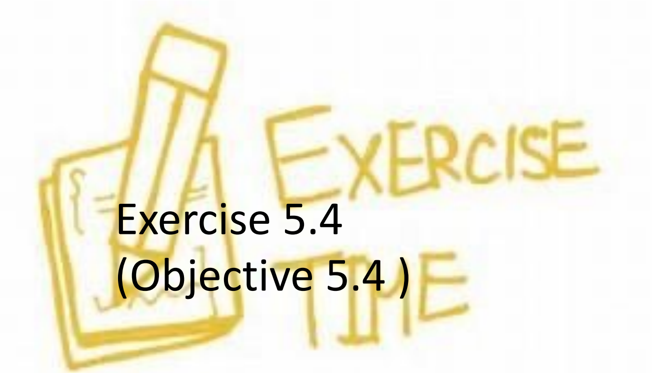
```
# print number from 0 to 10
# incremented by 2
for i in range(0,10,2):
    print(i, end=" ")
print()

# print number from 10 to 5
# decremented by 2
for i in range(10,5,-2):
    print(i, end=" ")
```

- `step` can be negative
- `stop` is exclusive!!

Output:

```
0 2 4 6 8
10 8 6
```

Practice



Exercise 5.4
(Objective 5.4)

Use **break** to Break Out of the Loop

- `break` is a statement
- It terminates the loop containing it.

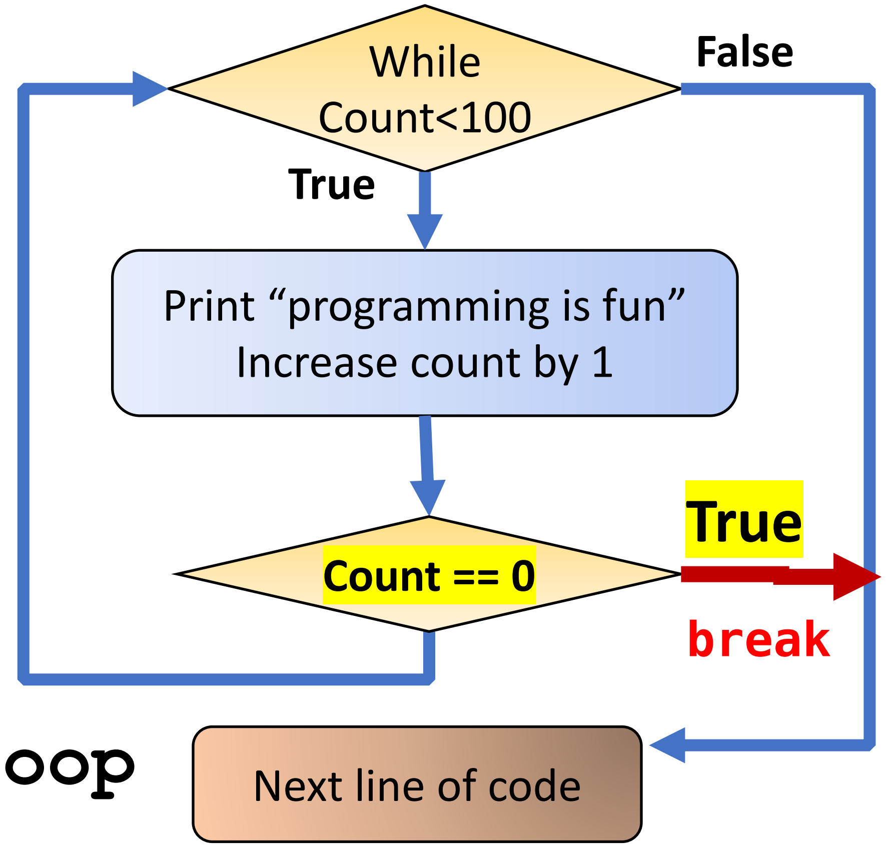
```
count=0

while count<100:
    print("Programming is fun")
    count=count+1
    if count == 2:
        break
```

How many times will “Programming is fun” be printed?

```
1 count=0

2 while count<100:
3     print("Programming is fun")
4     count=count+1
5     if count == 2:
6           break
```

Break out of the loop

“Programming is fun” is printed TWO times!!!



While Count<100

False

True

Print “programming is fun”
Increase count by 1

Count == 0

True

break

Next line of code

count=1 break

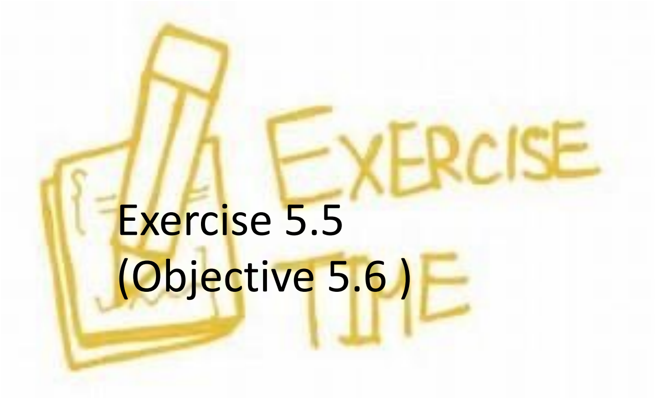
Execution order of lines : 1 2 3 4 5 2 3 4 5 6

count=0 count=2

break example: Guessing number problem

```
import random

#Generate a random number to be guessed
number = random.randint(0, 100)

print("Guess a magic number between 0 and 100")

while True:
    #Prompt the user to guess the number
    guess = int(input("Enter your guess: "))

    if guess == number:
        print("Yes, the number is " + str(number))
        break
    elif guess > number:
        print("Your guess is too high")
    else:
        print("Your guess is too low")
```

When loop stops?



Exercise 5.5
(Objective 5.6)