Ch 7: Loops in Python

Sometimes we need to repeat a set of instructions in our program.

eg - print 1 to 100

```
print(1)
print(2)
  |
  |
  |
print(100)
```

→ This is a dumb way.

We use loops to make this easier.

→ for loop
→ while loop

```
      True/False
while <condition>:   ← Syntax
    # Code #
```

The code inside while loop will only be executed if the condition is True. The loop terminates once the condition is False.

```
        → True
while (4>3):
    print("loop") → "loop" will be printed infinite times
```

```
        → False
while (7==8):
    print("loop") → "loop" will not be printed as the condition gets False
```

```
i = 0
while (i<10) :
      i = i+1
      print("this is good", i) → this is good 1
                                 this is good 2

                                this is good 10
```

As soon as the condition becomes False [i=10 & 10≮10] the loop stops executing.

Q → Write a program to print first n natural numbers in reverse order.

- □ break keyword - This instructs a loop to stop.

```
i = 0
while (i<10):
      i += 1
      print(i)
      if (i==6):
          break → 1
                  2
                  3
                  4
                  5
                  6
```

□ continue keyword - Helps you to skip the current instruction.

```
i = 0
while(i<10):
    i+=1
    if(i==6):
        continue
    print(i) → 1
               2
               3
               4
               5 }
               7 } → skipped 6
               8
               9
               10
```

□ pass keyword - This instructs a program to do nothing

```
if(10==1):
    print("Good")
```

└→ "Good" will be printed

```
if(10>=1):
    pass
```

└→ This will do nothing

For loop

for loops can be used in lists, tuples, sets.

```
li = ['Python', 10, True]
for i in li
    print(i) → Python
               10
               True
```

range() function

This is used to generate a sequence of numbers.

range (starting integer, stop)

range (100) → [0,1,2,3,4, - - - - - - - - 99]
- → start is by default 0
- → last number not included

range (1,101) → [1,2,3,4,5 - - - - 98, 99, 100]

- for loop with else

```
for i in [2,4,9,6]:
    print(i)
else:
    print("Task completed!")
```

→ The else is executed only when for loop completes it's process.

If the for loop breaks, else won't be executed.

$$\text{range}\,(1,101,\underset{\text{step size}}{4}) \longrightarrow [1,\underset{+4}{5},\ \underset{+4}{9},\ 13,\ \underset{+4}{17}, \text{------}, 93,\ \underset{+4}{97}]$$