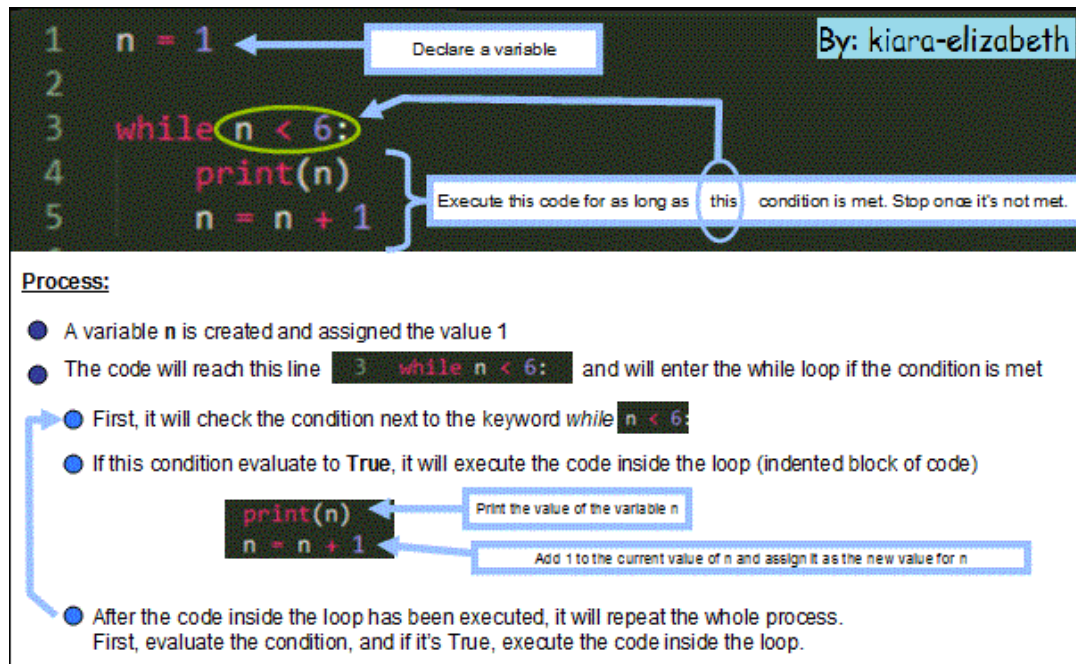


## Control Flow (For loops, While loops and range() ) behind the scenes! Diagrams.

Hi everyone! Control flow is an essential element of programming. You can build very powerful programs with loops, so let's see **how they work behind the scenes**, and how range() works, because you will use it in this course and in your future programming projects. So let's begin!

### Let's start with WHILE LOOPS:

In While loops a block of code will be executed for as long as a condition is met. Every time the block of code is executed, the condition will be checked again, and if it evaluates to True, the code inside the loop runs again, otherwise it exits the loop and continues executing the code below it.




When learning loops, a good strategy is to analyze each step and how values change when the code is executed. In this case, we determine if the code will be executed, the values for the variables involved each time the loop runs and what will be printed on each iteration.

The value of `n` is updated every time the loop executes. When the condition is checked, the new value of `n` will determine if the loop runs again or if it stops.

```

1 n = 1
2
3 while n < 6:
4     print(n)
5     n = n + 1

```



```

>>> n = 1
>>> while n < 6:
...     print(n)
...     n = n + 1
1
2
3
4
5

```

Through the loop	n Before	n < 6	print(n)	n = n + 1	n After
1 <sup>st</sup> time	1	True ✓	1	n = 1 + 1	2
2 <sup>nd</sup> time	2	True ✓	2	n = 2 + 1	3
3 <sup>rd</sup> time	3	True ✓	3	n = 3 + 1	4
4 <sup>th</sup> time	4	True ✓	4	n = 4 + 1	5
5 <sup>th</sup> time	5	True ✓	5	n = 5 + 1	6
6 <sup>th</sup> time	6	False			

**Stop!**

**Notice that n<6 is not n <=6.**

- The condition n<6 doesn't include the value 6.
- If it was n<=6 it would execute the loop one more time

By: kiara-elizabeth

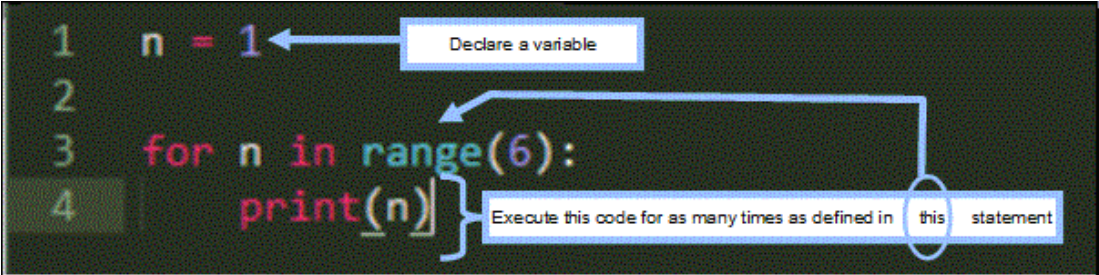
**Now let's study FOR LOOPS:**

For loops use the range() function.

```

1 n = 1
2
3 for n in range(6):
4     print(n)

```



- A variable **n** is created and assigned the value 1
- The code will reach this line `for n in range(6):` and will execute the loop as many times as defined on this line by `range()`.
  - Each time through the loop, when it's finished executing the code inside the loop, it will re-assign a number contained in the sequence of numbers returned by range as the value of n (diagram below)

By: kiara-elizabeth

The range function returns a series of integers that are determined by the numbers we use inside parentheses ( ).

**The general structure of range( ) is ---> range( start, stop, step)**

- The **start** point is the integer from where the sequence will start. (Will be included)
- The **stop** point is the integer at which the sequence will end (**NOT** Included)
- The **step** is the "jump" between one number and the next

You can find [more info on range\(\) at this link](#)

**range(start, stop, step):**

Returns a collection of integers from **start** up to **stop-1**

`range(6) ==> 0, 1, 2, 3, 4, 5`

`range(1, 5) ==> 1, 2, 3, 4` (+1 jumps)

`range(1, 5, 2) ==> 1, 3` (+2 jump)

`range(5, 11, 3) ==> 5, 8` (+3 jump)

start stop step

The third parameter is the **step**, the "jump" from one number to the next. By default, if there are only two parameters inside the parentheses, step is 1.

Adding +3 would give 11, but it's not included since it's the stop parameter

By: kiara-elizabeth

When we execute a for loop with range(), the value of the variable next to the **for** keyword will be update each time the loop runs, it will be assigned a number in the integer sequence returned by range(). The first time through the loop it will assign the first value in the collection of integers, the second time the second value will be assigned to the variable and so on.

The for loop stops when it reaches the end of the collection of integers returned by range().

```

1 n = 1
2
3 for n in range(6):
4     print(n)

```

```

>>> n = 1
>>> for n in range(6):
>>>     print(n)
0
1
2
3
4
5

```

Through the loop	range(6)	n Assigned by range()	print(n)
1 <sup>st</sup> time	0, 1, 2, 3, 4, 5	0	0
2 <sup>nd</sup> time	0, 1, 2, 3, 4, 5	1	1
3 <sup>rd</sup> time	0, 1, 2, 3, 4, 5	2	2
4 <sup>th</sup> time	0, 1, 2, 3, 4, 5	3	3
5 <sup>th</sup> time	0, 1, 2, 3, 4, 5	4	4
6 <sup>th</sup> time	0, 1, 2, 3, 4, 5	5	5

Overwrites the value assigned to the variable on line 1



By: kiara-elizabeth

Hope it helps! If you have any questions or comments on this topic, please share them on the forums!

Estefania (kiara-elizabeth).