On loops

support

Tópicos

- Iteração: instruções `while`, `for`, `break`

Repetition

- Repetition is natural
- Maths
  - Succession,
  - Option to recursive definition (often referred as iterative)
  - Option != only option…
- When something is naturally repetitive
  - Want to read many numbers
  - Some calculations
  - Output complex and big set of information

Mathematical succession

Initial condition

The rule

$a_1 = 2$ y $a_n = 3a_{n-1}$

The times= the n you select

$$a_2 = 3a_1 = 3x2 = 6$$

$$a_3 = 3a_2 = 3x6 = 18$$

$$a_4 = 3a_3 = 3x18 = 54$$

Mathematical succession

Initial condition

The rule

$$a_1 = 2 \text{ y } a_n = 3a_{n-1}$$

$$a_2 = 3a_1 = 3x2 = 6$$

$$a_3 = 3a_2 = 3x6 = 18$$

$$a_4 = 3a_3 = 3x18 = 54$$

The times= the n ou select

```
aN= … the initial condition
i=0  # how many times
while I<N :  # end when I reachs N
    aN1= … aN  …   # the rule
    i=i+1  # next index
    aN = aN1  # the new becames the old
```

Mathematical succession

Initial condition

The rule

$a_1 = 2$ y $a_n = 3a_{n-1}$

$a_2 = 3a_1 = 3x2 = 6$

$a_3 = 3a_2 = 3x6 = 18$

$a_4 = 3a_3 = 3x18 = 54$

The times= the n ou select

```
aN= … the initial condition
i =0  # how many times
while I<n :  # end when I reachs n
    aN1= 3 * aN # the rule
    i=i+1  # next index
    aN = aN1  # the new becames the old
```

Mathematical succession

Initial condition

The rule

$a_1 = 2$ y $a_n = 3a_{n-1}$

$a_2 = 3a_1 = 3x2 = 6$

$a_3 = 3a_2 = 3x6 = 18$

$a_4 = 3a_3 = 3x18 = 54$

The times= the n ou select

```
aN= … the initial condition
for i in range( n) # end when I reachs N
#  while I<N :
    aN1= 3 * aN # the rule
    # i=i+1  # next index
    aN = aN1  # the new becames the old
```

Option to recursive definition

Factorial Formula by **Recurrence Relation**

$$n! = \begin{cases} 1 & if\ n = 0, \\ (n-1)! * n & if\ n > 0 \end{cases}$$

```
def fact( n ) :
   if  n==0 :
        return 1
   return  fact( n-1 )*n
```

Based on how mathematics describes the function

DON'T NEED LOOPS

Factorial formula by **Product**

$$n! = \prod_{k=1}^{n} k$$

Do not forget to include the n

```
def fact( n ) :
   f=1
   for i in range(n +1 ) :
       f=f*i
   return  f
```

Based on product definition

Option to recursive definition

$$F_n = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F_{n-1} + F_{n-2} & n > 1 \end{cases}$$

Just look to the solution and try to understand it…

```
def fib( n ) :
    if  n<=1 :
        return n
    return fib( n-1 )+ fib(n-2)
```

DON'T NEED LOOPS

```
def fib( n ) :
    if  n<=1 :
        return n
    n1=1
    n2=1
    for i in range(2,n+1 ) :
        f= n1 + n2
        n2=n1
        n1=f
    return  f
```

Do not forget to include the n

Why loops? Repeat something

Note: this is not a undeniable truth but can help when starting to use loops

Know how many time?

```
fruits = ['banana', 'apple',  'mango']
for fruit in fruits:        # Second Example
   print 'Current fruit :', fruit

print "Good bye!"
```

```
# printing first 20
# whole number
for i in range(20):
    print(i, end = " ")
```

```
fruits = ['banana', 'apple',  'mango']
for index in range(len(fruits)):
   print 'Current fruit :', fruits[index]

print "Good bye!"
```

When you know and for everything else

```
count = 0
while (count < 9):
   print 'The count is:', count
   count = count + 1

print "Good bye!"
```

```
a=int(input(“a?”))
while a!=0 :
    a=int( input(“a?”))
```

Repetion, loops

while loop

Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.

for loop

Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.

nested loops

You can use one or more loop inside any another while, for or do..while loop.

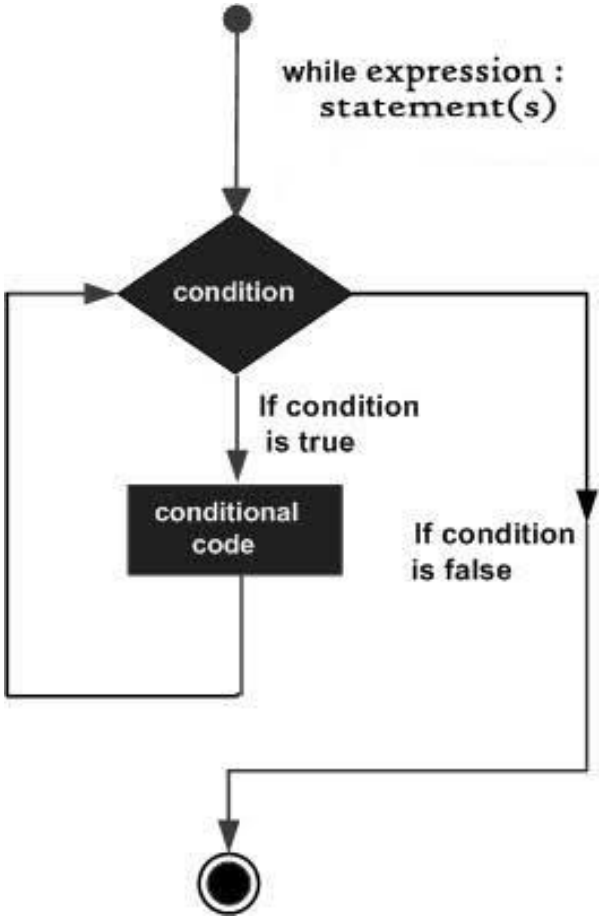
https://www.tutorialspoint.com/python/python_loops.htm
https://www.w3schools.com/python/python_while_loops.asp
https://realpython.com/python-while-loop/
https://www.tutorialspoint.com/python/python_while_loop.htm

while

```
count = 0
while (count < 9):
   print 'The count is:', count
   count = count + 1

print "Good bye!"
```

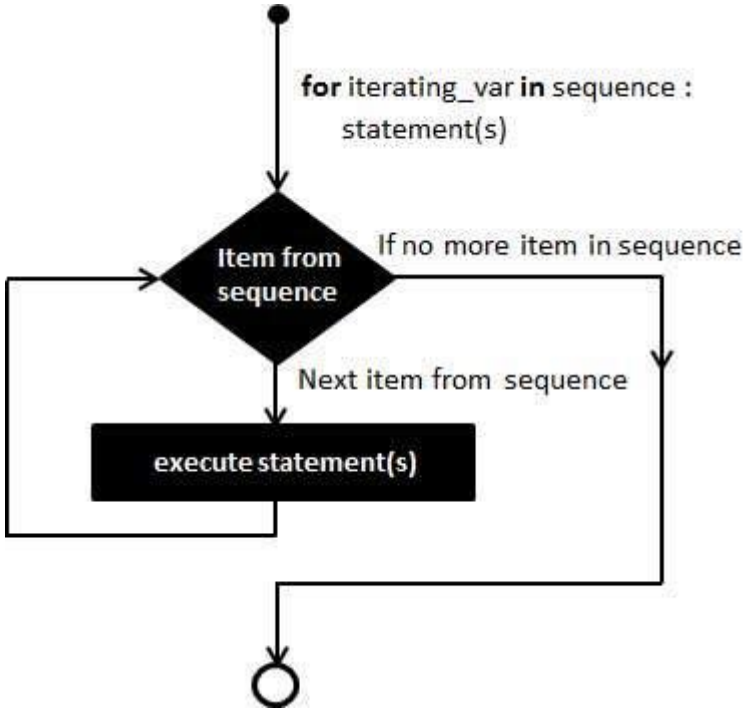
```
The count is: 0
The count is: 1
The count is: 2
The count is: 3
The count is: 4
The count is: 5
The count is: 6
The count is: 7
The count is: 8
Good bye!
```

```
while <expr>:
    <statement>
    <statement>
    break
    <statement>
    <statement>
    continue
    <statement>
    <statement>

<statement>
```



For: collection

```
fruits = ['banana', 'apple',  'mango']
for fruit in fruits:        # Second Example
   print 'Current fruit :', fruit

print "Good bye!"
```



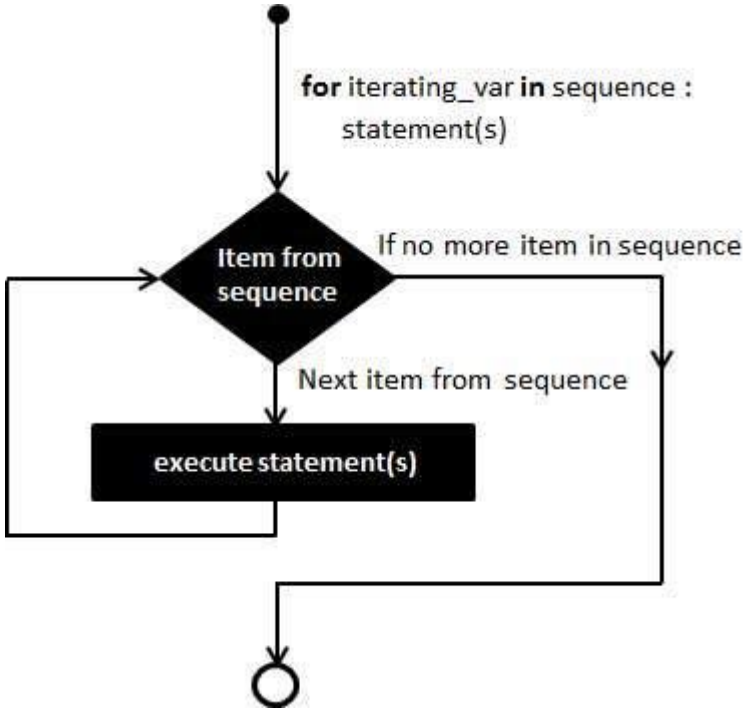

```
Current fruit : banana
Current fruit : apple
Current fruit : mango
Good bye!
```

https://realpython.com/python-for-loop/

For: ranges

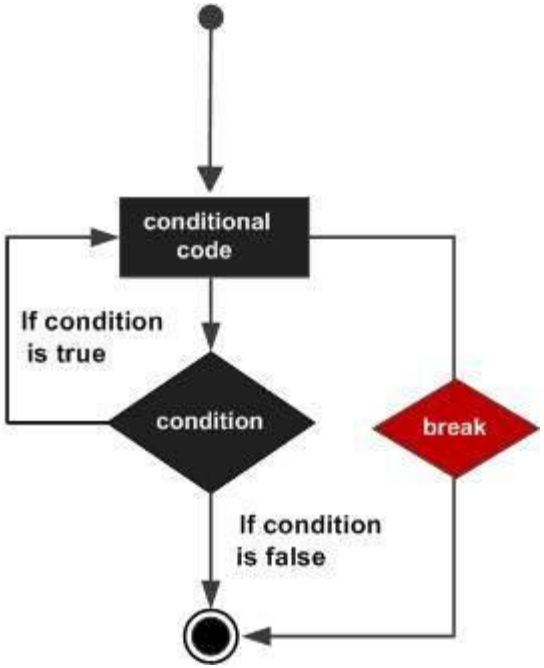
```
fruits = ['banana', 'apple',  'mango']
for index in range(len(fruits)):
   print 'Current fruit :', fruits[index]

print "Good bye!"
```

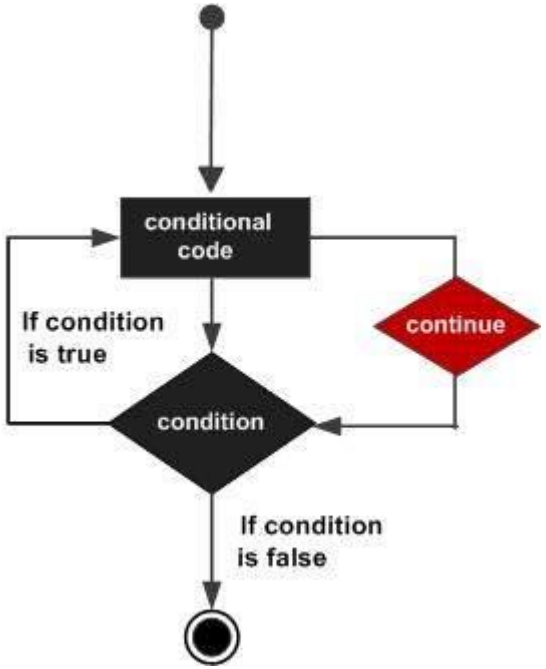
```
Current fruit : banana
Current fruit : apple
Current fruit : mango
Good bye!
```



https://realpython.com/python-for-loop/

Break, continue






```
for letter in 'Python':     # First Example
   if letter == 'h':
      break
   print 'Current Letter :', letter
```

```
for letter in 'Python':     # First Example
   if letter == 'h':
      continue
   print 'Current Letter :', letter
```

https://www.tutorialspoint.com/python/python_continue_statement.htm
https://www.tutorialspoint.com/python/python_break_statement.htm

Nested loops

```
for iterating_var in sequence:
   for iterating_var in sequence:
      statements(s)
   statements(s)
```

```
while expression:
   while expression:
      statement(s)
   statement(s)
```

https://www.tutorialspoint.com/python3/python_nested_loops.htm
https://www.tutorialspoint.com/python/python_nested_loops.htm
https://study.com/academy/lesson/nested-loops-in-python-definition-examples.html

Nested loops: What does the following code does?

```
import sys
for i in range(1,11):
    for j in range(1,11):
        k = i*j
        print (k, end=' ')
    print()
```

```
for a in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]:
    for b in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]:
        print("{} x {} = {}".format(a, b, a*b))
```

Nested loops: What does the following code does?

```
import sys
for i in range(1,11):
    for j in range(1,11):
        k = i*j
        print (k, end=' ')
    print()
```

Nested loops: What does the following code does?

```
i = 2
while(i < 100):
    j = 2
    while(j <= (i/j)):
        if not(i%j): break
        j = j + 1
    if (j > i/j) : print i, " is prime"
    i = i + 1
```

Reading and… stopping

- Reading value
  - "input" from user

```
Input("a?")
```

- Reading values is a repetitive task

```
While … something…
    Input("a?")
```

- Usually is non eternal
  - Need to stop on some condition

```
While … something…
    Input("a?")
```

Some examples

Repeating this block

```
A=int( input("a?"))
If a==0 :   exit()
A=int( input("a?"))
If a==0 :   exit()
A=int( input("a?"))
If a==0 :   exit()
A=int( input("a?"))
If a==0 :   exit()
```

```
a=int(input("a?"))
while a!=0 :
    a=int( input("a?"))
```

Some examples

```
A=int( input("a?"))
If a==0 :   exit()
A=int( input("a?"))
If a==0 :   exit()
A=int( input("a?"))
If a==0 :   exit()
A=int( input("a?"))
If a==0 :   exit()
```

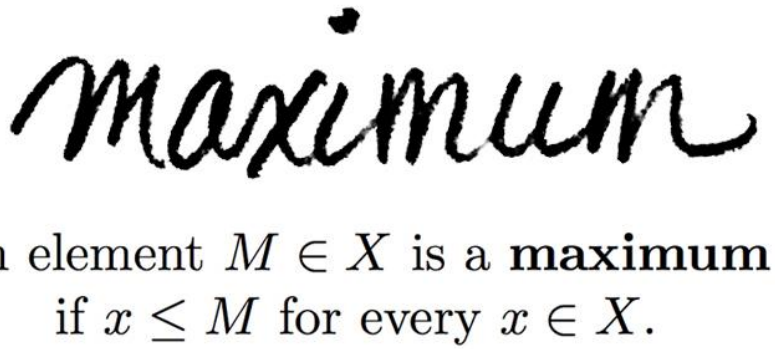
```
a=int(input("a?"))
while a!=0 :
    a=int( input("a?"))
```

Read values (the "a") until a 0 appears
Or
While values != 0 are read …

Some examples: maximum


```
A=int( input("a?"))
Mx=a
If a==0 :   exit()
If  mx<a :
        mx=a
A=int( input("a?"))
If a==0 :   exit()
If  mx<a :
        mx=a
A=int( input("a?"))
If a==0 :   exit()
If  mx<a :
        mx=a
A=int( input("a?"))
If a==0 :   exit()
If  mx<a :
        mx=a
```

Some examples



Repeating this block

```
A=int( input("a?"))
Mx=a
If a==0 :   exit()
If  mx<a :
      mx=a
A=int( input("a?"))
If a==0 :   exit()
If  mx<a :
      mx=a
A=int( input("a?"))
If a==0 :   exit()
If  mx<a :
      mx=a
A=int( input("a?"))
If a==0 :   exit()
If  mx<a :
      mx=a
```

```
a=int(input("a?"))
mx=a
while a!=0 :
    If a>mx :
        mx = a
    a=int( input("a?"))
```

Read values (the "a") until a 0 appears
and finds maximum
Or
While values != 0 are read …
and finds maximum

Some examples

Repeating this block

```
str= input("a?")
If str!="" :      exit()
A= input("a?")
If str!="" :      exit()
A= input("a?")
If str!="" :      exit()
A= input("a?")
If str!="" :      exit()
```

```
str=input("a?")
while str!="" :
    str= input("a?")
```

Some examples

str= input(“a?”)

If str!=“” :	**exit()**
A= input(“a?”)	
If str!=“” :	**exit()**
A= input(“a?”)	
If str!=“” :	**exit()**
A= input(“a?”)	
If str!=“” :	**exit()**

```
str=input(“a?”)
while str!=“” :
    str= input(“a?”)
```

Read string until nothing is entered (empty string “”)
Or
Read string while something not empty (!=“”) is read …

What does the following code does?

```
a=int( input("a?"))
while a>= 0 :
    a=int(input("a?"))
```

```
a=int( input("a?"))
while a!= 0 :
    a=int(input("a?"))
```

```
str=input("a?")
while str!="" :
    str= input("a?")
```

Condition on input
As a number
As a string
…

```
a=int( input("a?"))
while a%2== 0 :
    a=int(input("a?"))
```

```
str=input("a?")
while str!="end" :
    str= input("a?")
```

What does the following code does?

Read values while they >=0

```
a=int( input("a?"))
while a>= 0 :
    a=int(input("a?"))
```

Read values while they are !=0

```
a=int( input("a?"))
while a!= 0 :
    a=int(input("a?"))
```

Read string while they not empty

```
str=input("a?")
while str!="" :
    str= input("a?")
```

Condition on input
As a number
As a string
…

```
a=int( input("a?"))
while a%2== 0 :
    a=int(input("a?"))
```

Read values while they are even

```
str=input("a?")
while str!="end" :
    str= input("a?")
```

Read strings until "end" string is read

Reading values for something

```
a=int(input("a?"))
while a!=0 :
    a=int( input("a?"))
```

The task

```
a=int(input("a?"))
mx=a
while a!=0 :
    If a>mx :
        mx = a
    a=int( input("a?"))
```

```
a=int(input("a?"))
mn=a
while a!=0 :
    If a<mn :
        mn = a
    a=int( input("a?"))
```

```
a=int(input("a?"))
sum=0
while a>= 0 :
    sum=sum+a
    a=int(input("a?"))
```

Reading values for something

```
a=int(input("a?"))
while a!=0 :
    a=int( input("a?"))
```

The task

Read values while they are !=0 and finds the minimum

```
a=int(input("a?"))
mx=a
while a!=0 :
```

Read values while they are !=0 and finds the maximum

```
    a=int( input("a?"))
```

```
a=int(input("a?"))
mn=a
while a!=0 :
    If a<mn :
        mn = a
    a=int( input("a?"))
```

```
a=int(input("a?"))
sum=0
while a>= 0 :
    sum=sum+a
    a=int(input("a?"))
```

Read values while they are >=0 and finds their sum

What does the following code does?

The condition

The task

```
a=int(input("a?"))
sum=0
c=0
while a>= 0 :
    sum=sum+a
    c=c+1
    a=int(input("a?"))
avg=sum / c
```

Read values while they are >=0
and calculates the average of positive values

```
str=input("a?")
sum=0
c=0
while str!="" :
   a=int(str)
    sum=sum+a
    c=c+1
    str= input("a?")
avg=sum / c
```

Reads values an empty string is read
and calculates the average of all values

What does the following code does?

The condition

The task

```
a=int(input("a?"))
sum=0
c=0
while a>= 0 :
    sum=sum+a
    c=c+1
    a=int(input("a?"))
avg=sum / c
```

```
a=int(input("a?"))
sum=0
c=0
while True :
    if a==0 :
        break
    sum=sum+a
    c=c+1
    a=int(input("a?"))
avg=sum / c
```

Breaks and continue

just to allow the entry in the while
i.e. a value to turn the condition True

```
a=1
sum=0
c=0
while a!=0 :
    a=int( input("a?"))
    if a<0 :
        continue
    sum=sum+a
    c=c+1
avg=sum / c
```

The condition

The task

```
a=int(input("a?"))
sum=0
c=0
while True :
    if a==0 :
        break
    sum=sum+a
    c=c+1
    a=int(input("a?"))
avg=sum / c
```

Breaks and continue

Read values while they are !=0
and calculates the average of positive values

```
a=1
sum=0
c=0
while a!=0 :
    a=int( input("a?"))
    if a<0 :
        continue
    sum=sum+a
    c=c+1
avg=sum / c
```

The condition

The task

```
a=int(input("a?"))
sum=0
c=0
while True :
    if a==0 :
        break
    sum=sum+a
    c=c+1
    a=int(input("a?"))
avg=sum / c
```

Read values while they are !=0
and calculates the average of all values

What does the following code does?

The condition

The task

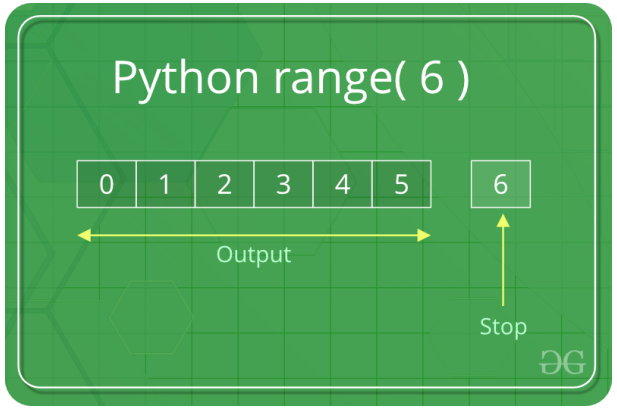
```
a=1
sum=0
c=0
while a!=0 :
    a=int( input("a?"))
    if a<0 :
        continue
    sum=sum+a
    c=c+1
avg=sum / c
```

```
str=input("a?")
sum=0
c=0
while str!="" :
    a=int(str)
    sum=sum+a
    c=c+1
    str= input("a?")
avg=sum / c
```

Question: what with the "avg=sum/c" in ALL previous examples? Fix it so the code works properly

range

Some help: range

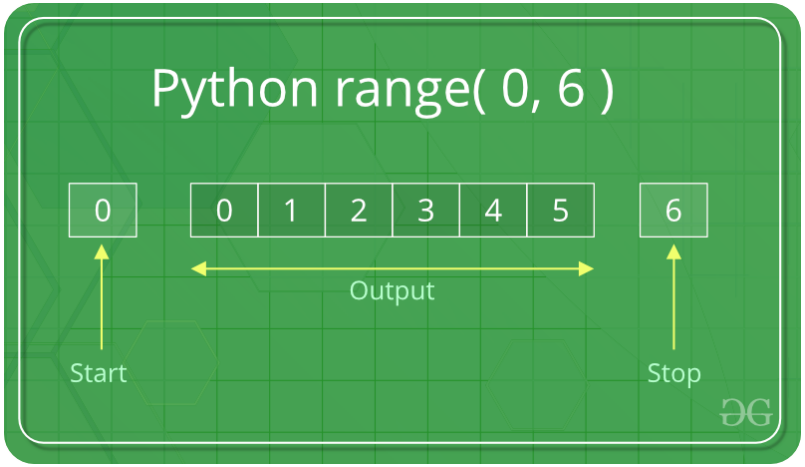



```
for i in range(10):
    print(i, end =" ")
print()

# printing first 20
# whole number
for i in range(20):
    print(i, end = " ")
```

```
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

https://www.geeksforgeeks.org/python-range-function/

Some help: range

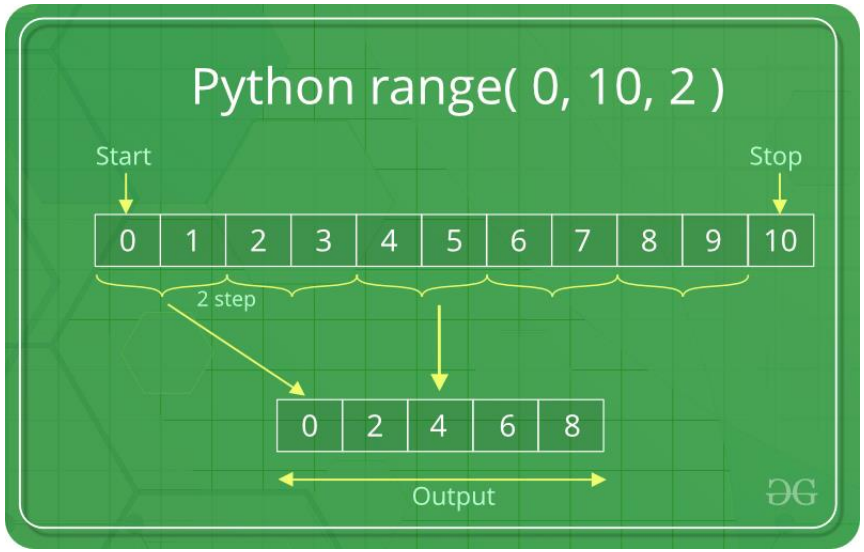



```
for i in range(1, 20):
    print(i, end =" ")
print()

# printing a natural
# number from 5 t0 20
for i in range(5, 20):
    print(i, end =" ")
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

https://www.geeksforgeeks.org/python-range-function/

Some help: range




```
0 3 6 9 12 15 18 21 24 27
0 5 10 15 20 25 30 35 40 45
```

```
for i in range(0, 30, 3):
    print(i, end = " ")
print()

# using range to print number
# divisible by 5
for  i in range(0, 50, 5):
     print(i, end = " ")
```

https://www.geeksforgeeks.org/python-range-function/

Some examples

```
>>> # One parameter
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
>>> # Two parameters
>>> for i in range(3, 6):
...     print(i)
...
3
4
5
>>> # Three parameters
>>> for i in range(4, 10, 2):
...     print(i)
...
4
6
8
>>> # Going backwards
>>> for i in range(0, -10, -2):
...     print(i)
...
0
-2
-4
-6
-8
```

```
>>> my_list = ['one', 'two', 'three', 'four', 'five']
>>> my_list_len = len(my_list)
>>> for i in range(0, my_list_len):
...     print(my_list[i])
...
one
two
three
four
five
```

https://www.pythoncentral.io/pythons-range-function-explained/

Challenge: define your own range function

- Range(end)
- Range(start, end)
- Range(start,end, step)

- Note: just print the numbers to the screen

Return the numbers: Just a help

```
L=[]
for i in range(0,10) :
   L.append( i )
```

```
list = []           ## Start as the empty list
list.append('a')    ## Use append() to add elements
list.append('b')
```

The end