Week 05 - Some ideas and exercises

Mathematical sucession

Initial condition

The rule

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

The times= the n ou select

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

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

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

Mathematical sucession

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 sucession

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
```

Python 3.6
(known limitations)

```
1  i =0  # how many times
2  n=5
3  aN= 2 # the initial condition
4  while i<n :  # end when I reachs n
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

Frames Objects

Global frame

i	0
n	5
aN	2

http://tiny.cc/k8g0tz

Python 3.6
(known limitations)

```
1  i =0  # how many times
2  n=5
3  aN= 2 # the initial condition
4  while i<n :  # end when I reachs n
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

```
termo a(1) é 2
```

Frames Objects

Global frame

i	0
n	5
aN	2

Python 3.6
(known limitations)

```
1  i =0  # how many times
2  n=5
3  aN= 2 # the initial condition
4  while i<n :  # end when I reachs n
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

```
termo a(1) é 2
```

Frames Objects

Global frame

i	1
n	5
aN	6
aN1	6

Python 3.6
(known limitations)

```
1  i =0  # how many times
2  n=5
3  aN= 2 # the initial condition
4  while i<n :  # end when I reachs n
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

```
termo a(1) é 2
termo a(2) é 6
termo a(3) é 18
termo a(4) é 54
termo a(5) é 162
```

Frames Objects

Global frame	
i	5
n	5
aN	162
aN1	486

Python 3.6
(known limitations)

```
1  i =0  # how many times
2  n=5
3  aN= 2 # the initial condition
4  while i<n :  # end when I reachs n
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

```
termo a(1) é 2
termo a(2) é 6
termo a(3) é 18
termo a(4) é 54
termo a(5) é 162
```

Frames Objects

Global frame

i	5
n	5
aN	486
aN1	486

Python 3.6
(known limitations)

```
1  i =0  # how many times
2  n=5
3  aN= 2 # the initial condition
4  while i<n :  # end when I reachs n
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

```
termo a(2) é 6
termo a(3) é 18
termo a(4) é 54
termo a(5) é 162
the end
```

Frames Objects

Global frame	
i	5
n	5
aN	486
aN1	486

Week 05 – Some notes on exercises

Mathematical sucession

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
```

Python 3.6
(known limitations)

```
1  n=5
2  aN= 2 # the initial condition
3  for i in range( n) : # end when i reachs n (0,1,2 , n-1 )
4  #  while I<N :
5    print("termo a({}) é {}".format((i+1),aN))
6    aN1= 3 * aN # the rule
7    # i=i+1  # next index
8    aN = aN1  # the new becames the old
9  print("the end")
```

Edit this code

line that just executed
next line to execute

Print output (drag lower right corner to resize)

Frames Objects

http://tiny.cc/y8g0tz

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?"))
mx=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

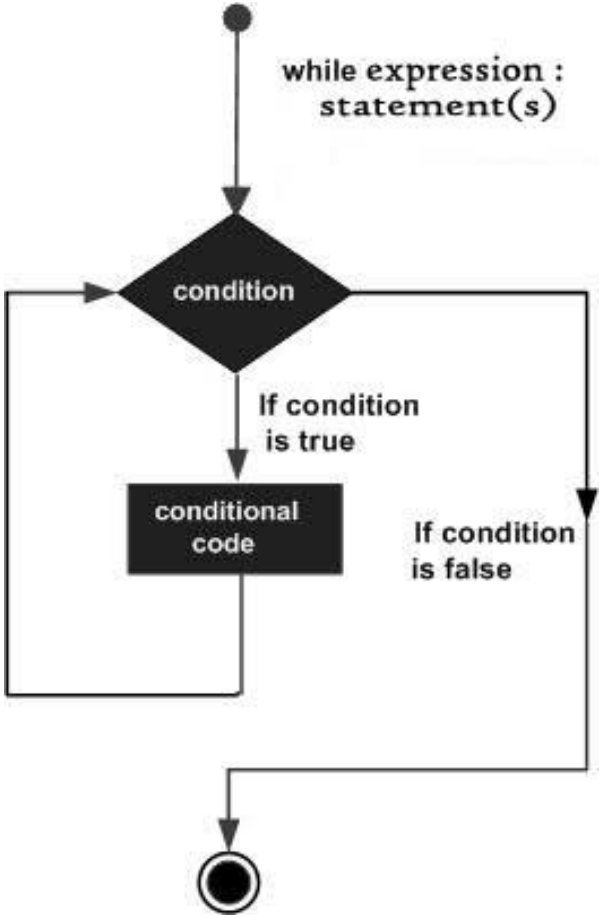
while

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

print "Good bye!"
```

http://tiny.cc/59g0tz

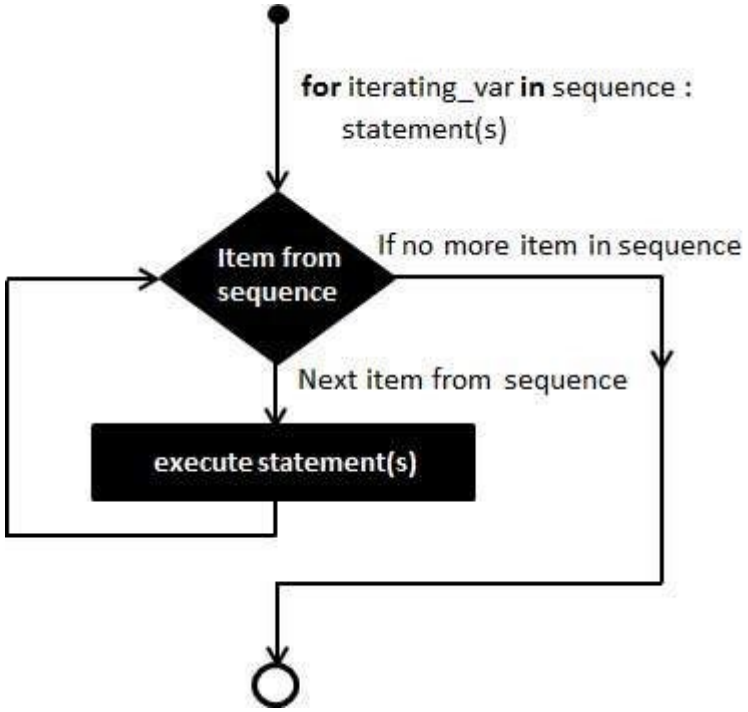
```
The count is: 0
The count is: 1
The count is: 2
The count is: 3
The count is: 4
The count is: 5
The count is: 6
The count is: 7
The count is: 8
Good bye!
```

```
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!"
```



http://tiny.cc/i9g0tz

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

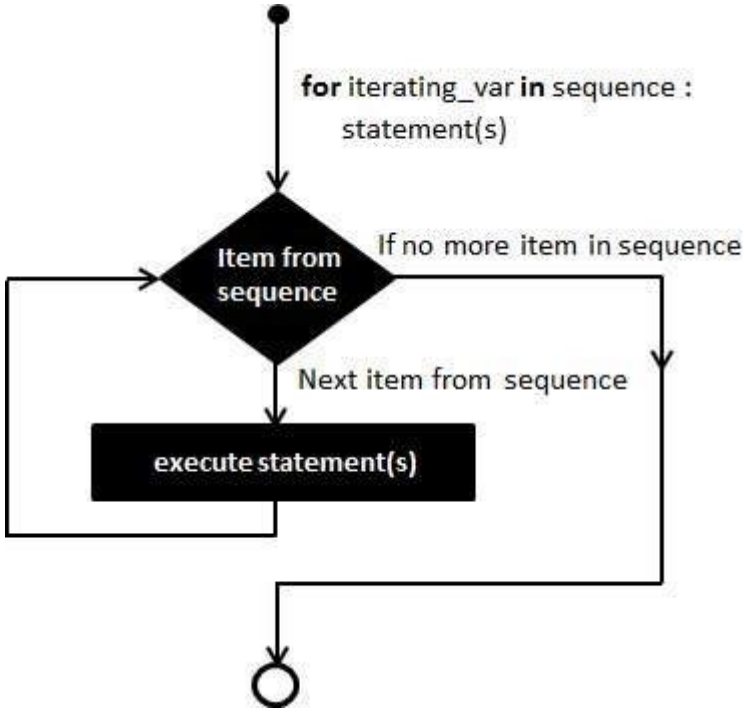
https://realpython.com/python-for-loop/

For: ranges

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

print "Good bye!"
```

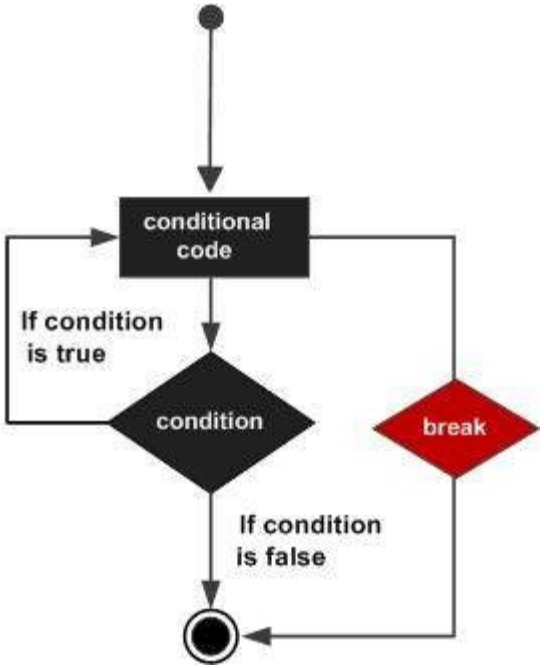
http://tiny.cc/q9g0tz

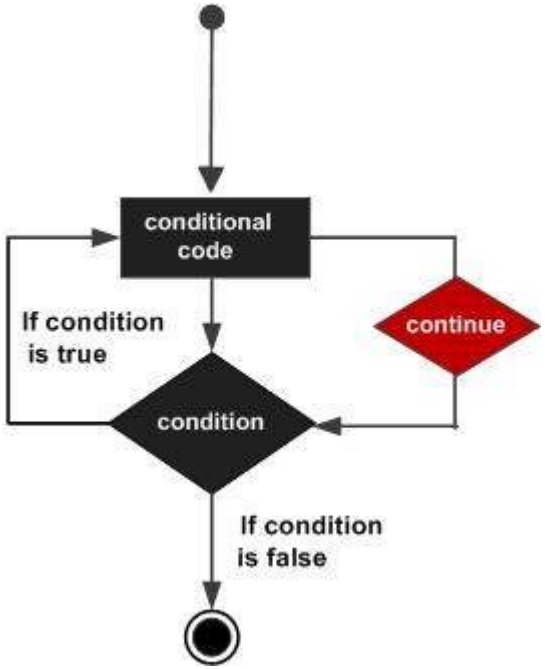
```
Current fruit : banana
Current fruit : apple
Current fruit : mango
Good bye!
```



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

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

```
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?"))
```

http://tiny.cc/y9g0tz

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()
```

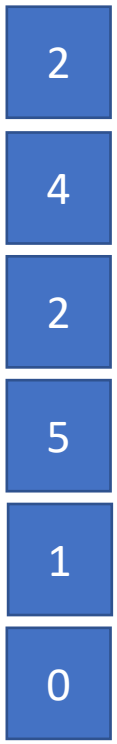
```
a=int(input("a?"))
while a!=0 :
    a=int( input("a?"))
```

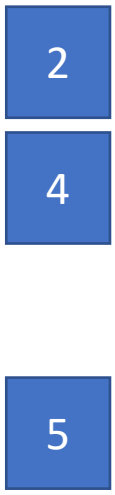
Read values (the "a") until a 0 appears

Or

While values != 0 are read …

Some examples: maximum





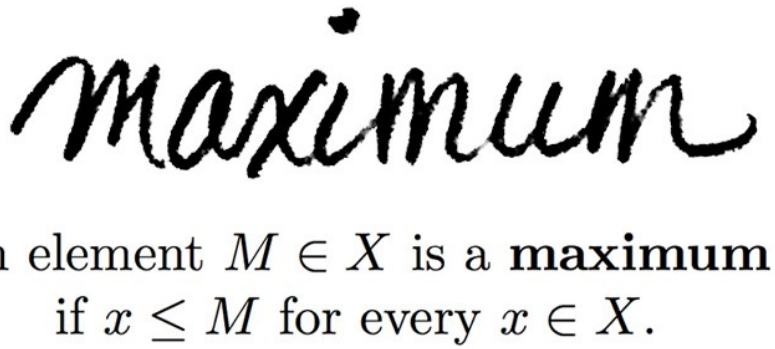
mx<a ? true

mx<a ? true

A==0? True

```
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




```
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?"))
```

http://tiny.cc/7ag0tz

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

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?”)
```

http://tiny.cc/eag0tz

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

```
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?"))
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
```

```
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
```

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

Breaks and continue

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
```

```
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

http://tiny.cc/xag0tz

```
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
```

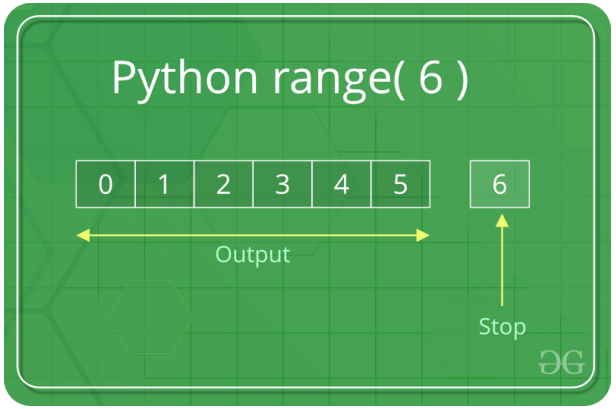
http://tiny.cc/8bg0tz

```
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

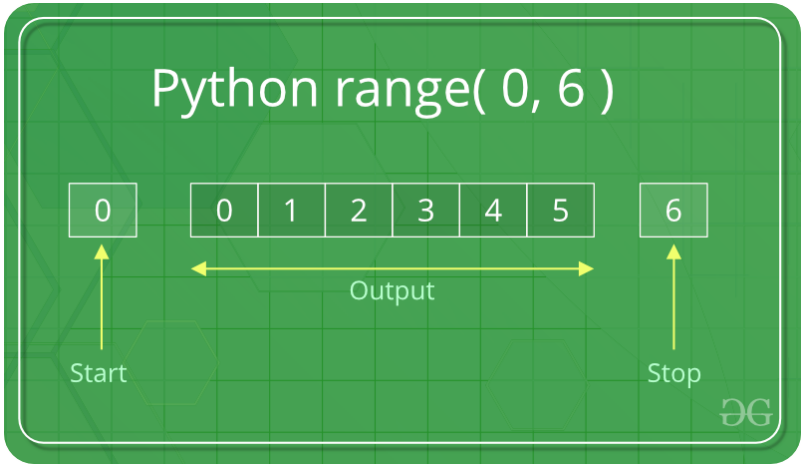



```
for i in range(10):
    print(i, end =" ")
print()

# 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

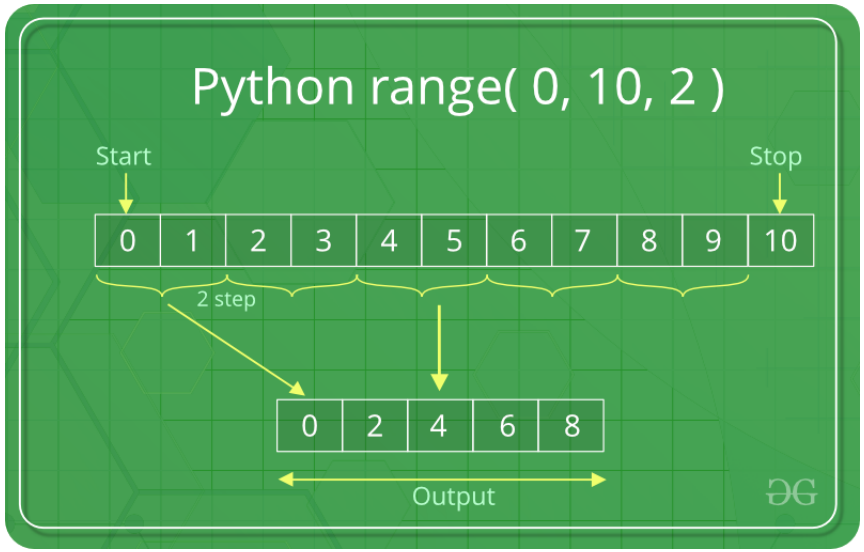



```
for i in range(1, 20):
    print(i, end =" ")
print()

# 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/

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
```

http://tiny.cc/kbg0tz

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()
```

http://tiny.cc/tbg0tz

```
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))
```

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/

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')
```

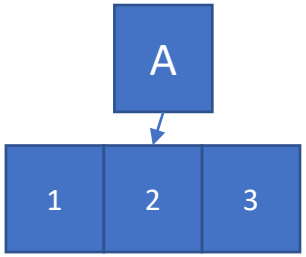
Exercises …

What does the following code does?

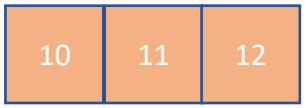
```
import sys
for i in range(1,11):
    for j in range(1,11):
        k = i*j
        print (k, end=' ')
    print()
```

Plot: define arrange…

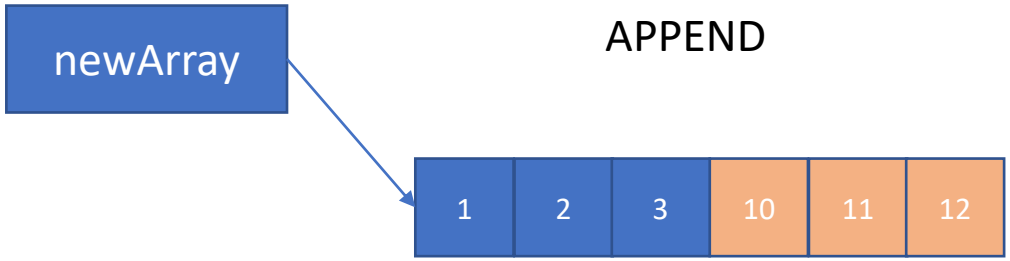
- Help
  - Loop for numpy.arrange / arrays




```
import numpy
a = numpy.array([1, 2, 3])
newArray = numpy.append (a, [10, 11, 12])
print(newArray)
```




```
[ 1  2  3 10 11 12]
```



https://likegeeks.com/numpy-array-tutorial/

4. Considere a sequência real $(¿¿0, U_1, ...)$ onde o primeiro termo é $U_0 = 100$ e os seguintes são dados por $U_n = 1.01\, U_{n-1} - 1.01$. O programa `sequenceUn.py` gera os primeiros 20 termos dessa sequência. Modifique o programa para mostrar todos os termos, enquanto forem positivos. Note que terá que usar uma instrução `while`. No fim, o programa deve dizer quantos termos gerou.

5. Escreva uma função `factorial(n)` que calcule o fatorial de n, definido por $n! = 1 \times 2 \times 3 \times \cdots \times n$. Teste a função com diversos valores de n.

9. A sequência de Fibonacci é uma sequência de inteiros na qual cada elemento é igual à soma dos dois anteriores: 0, 1, 1, 2, 3, 5, 8, 13, …, ou seja, cada termo obtém-se como $F_n = F_{n-1} + F_{n-2}$. Os primeiros valores são definidos como $F_0 = 0$ e $F_1 = 1$. Escreva uma função `Fibonacci(n)` para calcular o n-ésimo número de Fibonacci. *Sugestão: em cada iteração atualize e guarde os dois últimos valores da sequência.*

10. Escreva uma função `isPrime(n)` que devolva `True` se o número n é primo e `False`, caso contrário. Um número N é primo se não tiver divisores além de 1 e de N. *Sugestão: tente dividir o número por 2, por 3, etc. Se encontrar um divisor exato, então o número não é primo.* Teste a função fazendo um programa que percorre todos os números entre 1 e 100 e indique para cada um se é primo ou não.

6. O jogo HiLo consiste em tentar adivinhar um número (inteiro) entre 1 e 100. No início, o programa escolhe um número aleatoriamente. Depois, o utilizador introduz um número e o programa indica se é demasiado alto (High), ou demasiado baixo (Low). Isto é repetido até o utilizador acertar no número. O jogo acaba indicando quantas tentativas foram feitas. O programa `hilo.py` já tem um instrução para gerar um número aleatório com a função `randrange` do módulo `random`. Complete o programa para fazer o resto do jogo.

Some ideas

- Print all leap years between to given years
- Ler valores até algum ser 0
  - Imprimir máximo, minimo

The END