# Introduction to Computational Physcis By Python 2.7.x

400 PHY, King Saud University, First semester 1437-38 / 2016-17
Written By Abdulaziz S Alqasem
abdalqasem@ksu.edu.sa

## part 1: Basic programming:

1- Python as a calculater.
2- Variables
3- Basic Data types.
4- Some built in functions. 5- integer (int).
6- float>
7- string (str).
8- List.
9- Tuple.
10- Dictionary.
11- Boolean.
12- Input functions.
13- Conditional statement.
14- Loops: for & While.
15- Functions: lambda & def.
16- Libraries.
17- Input and Output>

In [1]:
```
# ملاحضة أي سطر برمجي أو حتى كتابة تُكتب بعد هذا الرمز (#) لا يقرأه البايثون، مثل هذا ال سطر الذي تقرأه الان
```

### 1- Python as a calculater

In [2]:
```
1 + 1
```

Out[2]: 2

In [3]:
```
3 - 1
```

Out[3]: 2

In [4]:
```
2 * 3
```

Out[4]: 6

```
In [5]: 8 / 2
Out[5]: 4

In [6]: 7 / 2
Out[6]: 3

In [7]: 7.0 / 2
Out[7]: 3.5

In [8]: 3 ** 3
Out[8]: 27

In [9]: pow(3,3)
Out[9]: 27

In [10]: 10 % 3  # يعطي الباقي من القسمة
Out[10]: 1

In [11]: 2 ** 2 + 4 * 5
Out[11]: 24

In [12]: (2 + 3) * 2
Out[12]: 10
```

الحاسب يقوم بحساب المدخلات بالترتيب التالي:
1- الأقواس
2- الأسس
3- الضرب والقسمة
4- الجمع والطرح

## 2- variables المتغيرات

المتغير هو أداة لتخزين البيانات في ذاكرة الحاسب ووسمها برمز معين يمكن لنا من خلاله استرجاع البيانات و القيام بعمليات عليها
شروط كتابة المتغير:
1- أن يبدأ بحرف أو بهذه العلامة _
2- يجب أن لا يحتوي المتغير على مسافة
3- يجب أن لايكون المتغير يشابه أحد الدوال الاساسية للبايثون مثل len

ملاحظة يمكن حذف المتغير من خلال الدالة del

In [13]:
```
a = 2
a * 5
```

Out[13]: 10

In [14]:
```
a = a**2
a
```

Out[14]: 4

In [15]:
```
a + 4
```

Out[15]: 8

In [16]:
```
_b = 8
c = 3 * _b
c
```

Out[16]: 24

In [17]:
```
v,w = 2,4
```

In [18]:
```
w
```

Out[18]: 4

In [19]:
```
del a # هدا الامر يقوم بحذف اي متغير
```

## 3- Basic data types: بعض أنواع البيانت الأساسية

1- Integer: (أي عدد صحيح من دون فاصلة)
1, 2, 5, 7

2- Float: (الاعداد التي لها فواصل)
1.5, 2.0, 10.0001

3- Complex number: (الأعداد المركبة)
2 + 3i

4- String. (نص كتابي )
'Abdulaziz'
"Abdulaziz"

5- List: (قائمة)
[1, 2, 655, 'Ahmad', 'abc100', 3.00002, 4.5]

6- Tuple: (قائمة لا يمكن التعديل عليها)
(1, 4, 'ww', 2.44, '234')

7- Set : (مجموعة)
المجموعة تحذف التكرار في العناصر
{1 ,1 , 1, 3, 5, '2sde'} >> {1, 3, 5, '2sde'}

8- Dictionary: ( قاموس أو فهرس )
{'ali':551234567 , 'Ahmad':553421421 , 'My _home':112345623}
{"a": 22 , 'b': 'R11' , 'c': '100'}

9- Boolean: ( عبارة منطقية )
1 == 1 >> True
1 != 1 >> False

## 4- Some Built in functions: بعض الدوال الاساسية في البايثون

1 - print طباعة
تقوم دالة الطباعة بطباعة المتغير على الشاشة
ملاحضة: عند طباعة أكثر من متغير يجب عليك عندئذ فصل المتغيرات بفاصلة

2- type() النوع
تقوم دالة النوع بإعطائك نوع المتغير فيما اذا كان رقم أو نص أو قائمة ... الخ

3- len() الطول
تقوم دالة الطول بعد العناصر

4- sum() الجمع
تقوم دالة الجمع بجمع العناصر حسابيا

5- max() الأعلى
تقوم دالة الأعلى بإعطائك أعلى قيمة

6- min() الأدنى
تقوم دالة الأدنى بإعطائك أدنى قيمة

7- abs() القيمة المطلقة
تقوم دالة القيمة المطلقة بتحول الرقم السالب إلى موجب

8- round(number, digit after point) التقريب
تقوم دالة التقريب بتحديد عدد الارقام بعد الفاصلة
number أي العدد المراد تقريبه
digit after point أي الرقم بعد الفاصلة

9- \n
print يقوم هذا الرمز بطباعة ما بعده في السطر التالي، لكن يجب ان يكون في سطر الدالة

## 5- Integers (int):

In [20]:
```
a = -3
a * 5
```

Out[20]: -15

In [21]:
```
r = 11
r/2   # عدد غير كسري تقسيم عدد غير كسري يعطي دائما عدد غير كسري
```

Out[21]: 5

In [22]:
```
abs(a)
```

Out[22]: 3

In [23]:
```
b = 4.5
int(b) # التحويل من عدد كسري إلى صحيح
```

Out[23]: 4

In [24]:
```
int('11') # يحول لك النص إلى رقم
```

Out[24]: 11

In [25]:
```
q = 10
q = q + 10
q
```

Out[25]: 20

In [26]:
```
q = 10
q += 10      # q = q + 10
q
```

Out[26]: 20

In [27]:
```
q = 10
q *= 3   # q = q * 10
q
```

Out[27]: 30

## 6- Float:

In [28]:
```
x = 44
float(x) # يعطيك عدد كسري
```

Out[28]: 44.0

In [29]:
```
r = 11.0
r/2
```

Out[29]: 5.5

In [30]:
```
float('83') # يحول لك النص إلى عدد كسري
```

Out[30]: 83.0

In [31]:
```
d = 14.32423
round(d,2) # يقوم بالتقريب: في هذا المثال قرب لك الى عددين
```

Out[31]: 14.32

In [32]:
```
#Complex:
z = complex(3,5)
z
```

Out[32]: (3+5j)

In [33]:
```
z.real # الجزء الحقيقي
```

Out[33]: 3.0

In [34]:
```
z.imag # الجزء التخيلي
```

Out[34]: 5.0

In [35]:
```
z.conjugate() # يعطيك المرافق
```

Out[35]: (3-5j)

## 7- String (str):

In [36]:
```
My_name = 'Abdulaziz'
My_name = "Abdulaziz"
# يمكن كتابة النص بكلا الطريقتين أعلاه
type(My_name)
```

Out[36]: str

In [37]:
```
len(My_name)
```

Out[37]: 9

In [38]:
```
 My_name[0] # يعتبر البايثون الحرف الأول بصفر
```

Out[38]: 'A'

In [39]:
```
My_name[4:] # أي حدد من العنصر الرابع إلى نهاية النص
```

Out[39]: 'laziz'

In [40]:
```
My_name[-1] # أخر عنصر يعتبره البايثون سالب واحد
```

Out[40]: 'z'

In [41]:
```
My_name[2:5] # أي حدد من العنصر الثاني إلى العنصر ما قبل الخامس
```

Out[41]: 'dul'

In [42]:
```
My_name[-4:-1]
```

Out[42]: 'azi'

In [43]:
```
My_name[:-3]
```

Out[43]: 'Abdula'

In [44]:
```
My_name[::-1] # أي أخرج العناصر من اليمين إلى اليسار (مقلوبة) الاتجاه
```

Out[44]: 'zizaludbA'

In [45]:
```
My_name[5:1:-1] # لاحئ ان البداية والنهاية كذلك مقلوبة لأن الاتجاه مقلوب
```

Out[45]: 'alud'

In [46]:
```
My_name[::2] # اي اطبع عنصر ثم اقفز عنصر
```

Out[46]: 'Adlzz'

In [47]:
```
My_Father_Name = 'Saleh'
Full_name = My_name + ' ' + My_Father_Name
Full_name
```

Out[47]: 'Abdulaziz Saleh'

In [48]:
```
print My_name + '\n' + My_Father_Name   # ينقل مابعده من النص إلى السطر التالي
 هذا الرمز ('\n')
```

```
Abdulaziz
Saleh
```

In [49]:
```
First_Letter = My_name[0] + ' . ' + My_Father_Name [0]
First_Letter
```

Out[49]: 'A . S'

In [50]:
```
str(55) # يحول العدد إلى نص
```

Out[50]: '55'

In [51]:
```
'1 + 4'     # هذا نص وليس عملية حسابية
```

Out[51]: '1 + 4'

In [52]:
```
'1' + '4'
```

Out[52]: '14'

In [53]:
```
My_name * 3   # يقوم بمضاعفة النص 3 مرات
```

Out[53]: 'AbdulazizAbdulazizAbdulaziz'

### *Some string Functions*

In [54]:
```
My_name.count('z')   # يعطيك عدد تكرار الحرف في النص
```

Out[54]: 2

In [55]:
```
'abc'.capitalize()  # يقوم بتحويل الحرف الاول فقط الى حرف كبير
```

Out[55]: 'Abc'

In [56]:
```
My_name.find('u')    # بحث عن مكان الحرف في النص، لاحظ أن البايثون يبدأ العد من الصف ر وليس الواحد
```

Out[56]: 3

In [57]:
```
My_name.index('u')
```

Out[57]: 3

In [58]:
```
My_name.replace('aziz','lah')  # يقوم باستبدال الكلمتين
```

Out[58]: 'Abdullah'

In [59]:
```
My_name.split('l')   # يقوم بفصل الكلمة عند هذا الحرف
```

Out[59]: ['Abdu', 'aziz']

In [60]:
```
'Hello, My Name is %s.' %(Full_name)
# يقوم بتعويض الكلمة التي بين القوسين في النص في مكان النسبة المؤية
```

Out[60]: 'Hello, My Name is Abdulaziz Saleh.'

In [61]:
```
'I LOVE %s and %s'  %('Physics', 'Pyhton')
# يقوم بتعويض الكلمتين التي بين القوسين في النص في مكان النسبة المؤية
```

Out[61]: 'I LOVE Physics and Pyhton'

In [62]:
```
'Hello, My Name is {}.'.format(Full_name) # طريقة أخرى
```

Out[62]: 'Hello, My Name is Abdulaziz Saleh.'

## 8- List:

In [63]:
```
my_list = ['a','b','c','1',2,3.0] # القائمة يمكن أن تحتوي نصوص وارقام وأنواع أخرى
my_list
```

Out[63]: ['a', 'b', 'c', '1', 2, 3.0]

In [64]:
```
type(my_list)
```

Out[64]: list

In [65]:
```
len(my_list)
```

Out[65]: 6

In [66]:
```
my_list[0]
```

Out[66]: 'a'

In [67]:
```
del my_list[0] # يقوم بحذف أول عنصر
my_list
```

Out[67]: ['b', 'c', '1', 2, 3.0]

In [68]:
```
my_list[3:]
```

Out[68]: [2, 3.0]

In [69]:
```
my_list[-2]
```

Out[69]: 2

In [70]:
```
my_list[1:3]
```

Out[70]: ['c', '1']

In [71]:
```
my_list[-3:-1]
```

Out[71]: ['1', 2]

In [72]:
```
my_list[::-1]
```

Out[72]: [3.0, 2, '1', 'c', 'b']

In [73]:
```
my_list * 2
```

Out[73]: ['b', 'c', '1', 2, 3.0, 'b', 'c', '1', 2, 3.0]

In [74]:
```
my_list[-2] * my_list[-1]
```

Out[74]: 6.0

**Some list_functions:**

In [75]:
```
my_list.append('New element') # يضيف عنصر جديد في القائمة
my_list
```

Out[75]: ['b', 'c', '1', 2, 3.0, 'New element']

In [76]:
```
number = range(0,20)  # 19 إلى 0 يعطيك قائمة بها أعداد من
number
```

Out[76]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

In [77]:
```
even_number = range(0,20,2) # يعطيك قائمة من الاعداد من 0 إلى 19 بحيث يكون بين كل
عدد وعدد قفزة برقمين
even_number
```

Out[77]: [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]

In [78]:
```
my_list.remove('New element') # يقوم بحذف عنصر من القائمة
my_list
```

Out[78]: ['b', 'c', '1', 2, 3.0]

In [79]:
```
my_list.count('a') # يقوم بعد العنصر المحدد
```

Out[79]: 0

In [80]:
```
my_list.index('1') # يقوم بالبحث عن مكان العنصر في القائمة
```

Out[80]: 2

In [81]:
```
x = ['c','b','z','a']
x.sort() # يقوم بترتيب القائمة أبجدياً
x
```

Out[81]: ['a', 'b', 'c', 'z']

In [82]:
```
sorted (['z','a']) # هذه الدالة ايضا تقوم بترتيب العناصر
```

Out[82]: ['a', 'z']

In [83]:
```
['a','b'] + ['c','d']
```

Out[83]: ['a', 'b', 'c', 'd']

In [84]:
```
num = [1,2,3]
sum( num )
```

Out[84]: 6

In [85]:
```
max(num)
```

Out[85]: 3

In [86]:
```
min(num)
```

Out[86]: 1

## 9- tuple:

In [87]:
```
my_tuple = (1,2,3,'a','a','c')
my_tuple
```

Out[87]: (1, 2, 3, 'a', 'a', 'c')

In [88]:
```
my_tuple.count('a')
```

Out[88]: 2

```
In [89]: my_tuple.index('c')
Out[89]: 5

In [90]: len(my_tuple)
Out[90]: 6

In [91]: my_tuple * 2
Out[91]: (1, 2, 3, 'a', 'a', 'c', 1, 2, 3, 'a', 'a', 'c')

In [92]: my_tuple = my_tuple + my_tuple
         my_tuple
Out[92]: (1, 2, 3, 'a', 'a', 'c', 1, 2, 3, 'a', 'a', 'c')

In [93]: my_tuple + ('x','e')
Out[93]: (1, 2, 3, 'a', 'a', 'c', 1, 2, 3, 'a', 'a', 'c', 'x', 'e')

In [94]: my_tuple[1]
Out[94]: 2

In [95]: my_tuple[3:8]
Out[95]: ('a', 'a', 'c', 1, 2)

In [96]: my_tuple[::-1]
Out[96]: ('c', 'a', 'a', 3, 2, 1, 'c', 'a', 'a', 3, 2, 1)

In [97]: num = (1,2,3)
         sum(num)
Out[97]: 6

In [98]: max (num)
Out[98]: 3

In [99]: min (num)
Out[99]: 1
```

## 10- dictionary:

فهرس = {مفتاح1:قيمة1 , مفتاح2:قيمة2} الفهرس مكون من مفاتيح و قيم

In [100]:
```
my_dic = {'a':10 , 'b':20 , 'c':30}
my_dic
```

Out[100]: {'a': 10, 'b': 20, 'c': 30}

In [101]:
```
type(my_dic)
```

Out[101]: dict

In [102]:
```
len(my_dic)
```

Out[102]: 3

In [103]:
```
my_dic['a'] #  هنا عبارة عن مفتاح والرقم 10 هو قيمة هذا المفتاح 'a'
```

Out[103]: 10

In [104]:
```
my_dic.keys() # يعطيك جميع المفاتيح في الفهرس على شكل قائمة
```

Out[104]: ['a', 'c', 'b']

In [105]:
```
my_dic.values() # يعطيك جميع القيم في الفهرس على شكل قائمة
```

Out[105]: [10, 30, 20]

In [106]:
```
my_dic ['new key'] = 100 # طريقة إضافة عنصر جديد في الفهرس

my_dic
```

Out[106]: {'a': 10, 'b': 20, 'c': 30, 'new key': 100}

## 11- Boolean:

In [107]:
```
1==1    #  أنت تسأل البايثون هنا هل 1 يساوي 1 ؟
```

Out[107]: True

In [108]:
```
1==2
```

Out[108]: False

In [109]:
```
1>2
```

Out[109]: False

In [110]:
```
2<3
```

Out[110]: True

In [111]:
```
1 < 3 > 0 != 2 # هذه العلامة =! تعني لا يساوي
```

Out[111]: True

In [112]:
```
1 != 3
```

Out[112]: True

In [113]:
```
r = 'abcd'
'a' in r   # أنت تسأل البايثون هل هذا العنصر موجود في هذا المتغير
```

Out[113]: True

In [114]:
```
'a' not in r # أنت تسأل البايثون هل هذا العنصر غير موجود في هذا المتغير
```

Out[114]: False

In [115]:
```
1 is 1 # == مثل
```

Out[115]: True

In [116]:
```
2 is not 4  # != مثل
```

Out[116]: True

In [117]:
```
True and True
```

Out[117]: True

In [118]:
```
True and False
```

Out[118]: False

In [119]:
```
False and False
```

Out[119]: False

In [120]:
```
True or True
```

Out[120]: True

In [121]:
```
True or False
```

Out[121]: True

In [122]:
```
False or False
```

Out[122]: False

In [123]:
```
n = 1
n == 2 or n == 1  # 1 يساوي 1 أو 2 أختبار منطقي: هل واحد يساوي
```

Out[123]: True

In [124]:
```
n == 2 and n == 1  # 1 يساوي 1 و 2 أختبار منطقي 1 يساوي
```

Out[124]: False

In [125]:
```
T = 5
T >= 3 and T < 10 and T == 5 or T != 5
```

Out[125]: True

In [126]:
```
T <= 20 or T > 0
```

Out[126]: True

In [127]:
```
L = [1,2,3]
len(L) == 3
```

Out[127]: True

In [128]:
```
L[1] == 2
```

Out[128]: True

## 12- Input Functions: دوال الادخال

1- input( هنا تكتب نص يظهر للمستخدم )
المدخلات أرقام فقط >>> المخرجات أرقام فقط

2- raw_input( هنا تكتب نص يظهر للمستخدم )
المدخلات أرقام أو حروف أو رموز او ...الخ >>> المخرجات نص فقط

ملاحضة: المدخلات التي يكتبها المستخدم تنتقل إلى المتغير

In [129]:
```
x = input('Entar Numbers Only: ')
# x دالة لإدخال الارقام فقط ويمنع ادخال الاحرف. والمدخلات تتنتقل الى المتغير
```

```
Entar Numbers Only: 100
```

In [130]:
```
type(x)
```

Out[130]: int

In [131]:
```
x * 5
```

Out[131]: 500

In [132]:
```
y = raw_input('Enter Numbers or Letters: ')
type(y)
```

```
Enter Numbers or Letters: phys 400
```

Out[132]: str

## 13- Conditional Statments (if,elif,else): الجمل الشرطية

الجملة الشرطية هي جملة برمجية تنفذ الأوامر التي بداخلها عندما يتحقق الشرط
تتكون الجملة الشرطية من الاتي:
1- a- if إذا
b- elif (تعمل اذا الشرط الذي قبلها لم يعمل) اخر إذا
c- else (اذا لم تعمل جملتا (اذا) و (اخر اذا) السابقتين تعمل هذه) آخر

2- Logical Statement جملة منطقية
3- : نقطتين رأسيتين تفيد بأنتهاء الجملة المنطقية
4- indentation (لكي يفهم البايثون ان اي كود مزاح هو ظمن الجملة الشرطية) ازاحة بمقدار أربعة أحرف
5- inner code (الاوامر التي ينفذها بايثون اذا تحقق الشرط) الكود الداخلي

ملاحضة: عندما تنتهي من الجملة الشرطية قم بالنزول الى سطر جديد من دون ازاحة وسيفهم الحاسب انك انتهيت

مثال واقعي للجملة الشرطية:
اذا كان مصباح الغرفة مضاءا:

اتركه مضاءا

اخر إذا كان مصباح الغرفة مقفلا :

قم بتشغيله

اخر :

قم بإصلاح المصباح

قم بغلق باب الغرفة

لاحظ ان جملة قم بغلق باب الغرفة تقوم بها دائما لانها ليست ظمن أي جملة شرطية

In [133]:
```
a = 1
if a == 1:
    print 'a is one'
```

```
a is one
```

In [134]:
```
my_friend = 'Ali'

if my_friend == 'Ahmad':
    print 'Ahmad is from Riyadh'

elif my_friend == 'Ali':
    print 'Ali as my best friend'
    print 'He is from Makkah'

print 'I Like all my Friend'
```

```
Ali as my best friend
He is from Makkah
I Like all my Friend
```

In [135]:
```
time_now = raw_input('What is the time Now: ') # تطلب من المستخدم ان يدخل الوق
ت
AM_OR_PM = raw_input('AM or PM: ') # تسأل المستخدم اذا كان الوقت صباحا او مساءا

time_now = int(time_now) # تحول الرقم المدخل من نص إلى رقم

if AM_OR_PM == 'PM' or AM_OR_PM == 'pm' : # اذا كان الوقت مساءا
    time_now += 12  # اضف 12 لكي تقوم بالتحويل الى نظام 24 ساعة


if  6 < time_now < 9 :
    print 'Give me two Eags sandwich and a cop of tea.'
    one_sandwich = 3
    tea = 2
    Money_paid = 2 * one_sandwich + tea

elif 12 < time_now <15 :
    print 'Give some rice And one orange juice.'
    rice = 12
    juice = 3
    Money_paid = rice + juice

elif 19 < time_now < 24:
    print 'Give me three piece of pizza and two bottle of water.'
    pizza = 15
    water = 1
    Money_paid =3 * pizza + 2 * water

else:
    print 'Thank you, I am not hungry.'
    Money_paid = 0

print 'You paid %s SR' % Money_paid
```

```
What is the time Now: 10
AM or PM: pm
Give me three piece of pizza and two bottle of water.
You paid 47 SR
```

# 14 - Loops الحلقات

## 1- For Loop حلقة لكل

تقوم هذه الحلقة بالمرور على كل العناصر في (قائمة، فهرس، نص، ...الخ) ثم
تجري على كل عنصر عمليات يحددها المُبرمج

تتكون هذه الحلقة من الاتي:
1- for لكل
2- variable/varibles متغير/متغيرات بحيث يكون العنصر المأخوذ مساوي لها
3- in في
4- Data that is consists of elements بيانات مكونة من عناصر
5- : نقطتين رأسيتين تبين نهاية الأمر
6- indentation ازاحة بمقدار اربعة حروف لكي يفهم الحاسب انها ظمن امر الحلقة
7- inner code الكود الداخلي الذي فيه اوامر يقوم الحاسب من خلالها باجراء نفس العمليات على كل عنصر

مثال على حلقة لكل:
لكل ورقة في الكتاب:

أكتب اسمك
اكتب رقمك الجامعي

قم بإغلاق الكتالب

لاحظ في هذا المثال بأنك قمت بالكتابة في كل ورقة حتى انتهى الكتاب
وبعد أن انتهيت من الكتابة في الكتاب قمت بإغلاقه

In [136]:
```
for x in [1,2,3]:
    print x
# هنا انتهت الحلقة
print 'Done'

# يقوم المتغير اكس بعمل دورات أو حلقات بحيث في كل دورة يأخذ قيمة
#واحدة فقط ثم يتغير إلى قيمة أخرى في الدورة التالية وهكذا حتى تنتهي القائمة

# ملاحضة مهمة، يجب أن تضع مسافة اربعة احرف للكود الذي بداخل الحلقات
# اذا انتهيت من الكود المراد تضمينه في الحلقات توقف عن ترك مسافة حتى يفهم الحاسب انك
انتهيت
```

```
1
2
3
Done
```

In [137]:
```
for letter in ('a','b','c'):
    print letter
```

```
a
b
c
```

In [138]:
```
for i in range(1,5):
    x = i**2
    print i,x
```

```
1 1
2 4
3 9
4 16
```

In [139]:
```
lis = [] #انشأنا قائمة فارغة
count = 0
my_name = 'Abdulaziz'

for i in my_name:
    print i,      # الفاصلة تجعل الحرف التالي يأتي يمين الحرف الاول، احذفها وانظر ماذا يح
صل
    lis.append(i) # يضيف الحرف الى القائمة
    count += 1  # يقوم هذا المتغير بالزيادة بمقدار واحد في كل دورة

print '\nMy_list is:', lis # اطبع القائمة التي اظفنا اليها الأحرف

print 'Number of loops is: ', count # طبع عدد المرات الي قمت فيها بالدوران في الحلق
ة
```

```
A b d u l a z i z
My_list is: ['A', 'b', 'd', 'u', 'l', 'a', 'z', 'i', 'z']
Number of loops is:  9
```

## 2- While Loop حلقة مادام

حلقة مادام هي حلقة تعمل مادام الشرط متحققا ومتى ما زال الشرط تنتهي الحلقة
حلقة مادام مكونة من الاتي:
1- While مادام
2- Condition الشرط
3- : نقطتين رأسيتين توضع عند انتهاء الشرط
4- Indentation ازاحة مسافة اربعة حروف
5- Inner cood الكود الداخلي التابع ل حلقة مادام

مثال على حلقة مادام:
مادام الشاي مرا:

اضف ملعقة صغيرة من السكر

اضف فنجال من الحليب
قل بسم الله
اشرب الشاهي حليب
قل الحمد الله

لاحظ في هذه الحلقة، الشرط هو مادام الشاي مرا، فاذا كان هذا صحيح فأضف ملعقة صغيرة من السكر ثم اختبر الشرط
فاذا لا زال الشرط متحققا اضف ملعقة اخرى صغيرة من السكر
ثم اختبر الشرط .. حتى ينتفي الشرط ويكون الشاي غير مُر، عندئذ اخرج من الحلقة
وقم بإضافة فنجال من الحليب ثم سم الله واشرب الشاهي حليب بالعافية ثم قل الحمد لله

In [140]:
```
x = 10
while x > 1: # مادام الشرط متحققا قم بعمل الكود الذي بداخلي
    print x,
    x -= 1 #   اطرح واحد من قيمة اكس في كل دورة
print 'Done'
# ملاحضة مهمة، يجب أن تضع مسافة اربعة احرف للكود الذي بداخل الحلقات
# ذا انتهيت من الكود المراد تضمينه في الحلقة توقف عن ترك مسافة حتى يفهم الحاسب انك ان تهيت
```

```
10 9 8 7 6 5 4 3 2 Done
```

# 15- Functions كتابة الدوال

## 1- lambda (small Function):

لامدا هي دالة يتم تعريفها بحيث اذا تم ادخال بيانات محددة فيها تقوم باداء عمليات برمجية عليها
يحددها المبرمج ثم تعطي مخرجات
لامدا تتكون من:
1- lambda دالة لامدا
2- variavle/variables متغير/اومتغيرات مفصولة ب فاصلة
3- : نقطتين رأسيتين تفيد انتهاء تعريف المتغيرات
4- functions العملية المراد تطبيقها على المتغيرات

مثال:
الريال = لامدا الدولار: الدولار * 3.75
الريال(10) >> 37.5

```
In [141]: y = lambda x : x + 10 # y تم عمل دالة وتعيينها للمتغير
          # x لاحظ اننا اولا عرفنا المتغير ثم وضعنا نقطتين رأسيتين ثم كتبنا المعادلة المراد اجرائها
          y(10) # هنا أنت تطلب من البايثون بأن يعوض الرقم 10 في اكس ثم يعطيك النتيجة

Out[141]: 20

In [142]: y = lambda s: s[::-1]
          y('abc')

Out[142]: 'cba'
```

## 2-Definition (def): تعريف دالة

تعريف الدالة هو عبارة عن صناعة تعريف كامل لدالة لها مدخلات يحصل لها عمليات برمجيه تحولها إلى مخرجات
ومفهومها اشمل واوسع واقوى واكثر استخداما من دالة لامدا
خطوات تعريف الدالة يكون كالاتي:
1- def تعريف
2- function name اسم للدالة المراد تعريفها
3- (a,b,c,...) المتغيرات للدالة يمكن أن يكون متغير واحد أو أكثر أو بدون متغير
4- : نقطتين رأسيتين تفيد بانتهاء سطر التعريف
5- indentation ازاحة لمسافة اربعة احرف لكل سطر موجود ظمن الدالة
6- inner code الاوامر التي تقوم بها الدالة لمعالجة المدخلات
7- return ارجع
يقوم بتحديد المخرجات، يمكن عدم استخدامها

مثال:

```
عرف دالة الريال(الدولار) :
    الريال = الدولار * 3.75
    ارجع الريال
الريال(10) >> 37.5
```

In [143]:
```
def y(x):
    return x + 10
y(10)
```

Out[143]: 20

In [144]:
```
def add_2_numbers(a,b):
    return a + b

add_2_numbers(10,43)
```

Out[144]: 53

In [145]:
```
def convert_str_to_int(str):
    int_number = int(str)
    return int_number

convert_str_to_int('22')
```

Out[145]: 22

In [146]:
```
r = convert_str_to_int
r('100')
```

Out[146]: 100

In [147]:
```
def v(v0,a,t): # لسرعة النهائية
    vf= v0 + a * t
    return vf
v(10,5,20)
```

Out[147]: 110

### Break: اقطع

اقطع يقوم ب قطع الحلقات والخروج منها. كما في المثال التالي

In [148]:
```
for i in range(10):
    if i == 5:
        break # 5 لاحظ ان الحلقة توقفت بعد ان تم قطعها هنا عند الرقم
    print i,
```

```
0 1 2 3 4
```

## 16- Libraries مكتبات

المكتبة هي عبارة عن ملف به أكواد جاهزة قام بها مجموعة من المبرمجين، يمكن لك استدعائها والاستفادة منه
يوجد مجموعة كبيرة من المكتاب المفيدة جدا للتطبيقات العلمية وكل ما عليك هو استدعائها واستخدامها
أمثلة على بعض المكتبات
1- math
sin, cos, factorial, exp يوجد فيها الدوال الرياضيات الاساسية مثل
2- numpy
3- scipy
3- pandas
4- matplotlib
5- sympy
6- os
7- time
8- datetime

يوجد عدة طرق لاستدعاء المكتبات:
1- import x قم باستدعاء المكتبة اكس
2- import x as s قم باستدعاء المكتبة اكس وارمز لها بالرمز اس
3- from x import y من المكتبة اكس استدعي الجزئية واي
4- from x import * قم باستدعاء كامل المكتبة اكس
ملاحضة: اكس تعني اي مكتبة و واي تعني اي جزئية من المكتبة و اس تعني اي رمز اختصار

In [149]:
```
import math   # math استدعاء لمكتبة
math.pi # استدعينا هنا قيمة الرقم باي
```

Out[149]: 3.141592653589793

In [150]:
```
import math as m
m.pi
```

Out[150]: 3.141592653589793

In [151]:
```
from math import pi
pi
```

Out[151]: 3.141592653589793

In [152]:
```
from math import * # لا يُنصح باستخدامه الا في حالات خاصة
pi
```

Out[152]: 3.141592653589793

In [153]:
```
import math
sin = math.sin # اختصار
sin(10)
```

Out[153]: -0.5440211108893698

## 17- Input and Output: الإدخال والإخراج

الإدخال والإخراج، هو لإدخال البيانات من خارج البايثون أو إخراجها.
يمكننا عمل ذلك باستخدام طرق مختلفة منها الملف النصي .txt

خطوات إدخال أو اخراج اي ملف:

1- نفتح الملف الذي نريد أن نكتب فيه
```
f = open(path,file mode)
f :   هو متغير يمثل الملف الذي تم فتحه أو انشاءه
path : هو موقع الملف في ذاكرة الحاسب، يجب أن يكون على شكل نص
file mode: نوع الملف، حيث أنه يوجد أنواع كثيرة، يجب أن يكون على شكل نص
    'r': يستخدم لقراءة الملف فقط
    'w': يستخدم للكتابة على الملف فقط، واذا قمت بالكتابة على الملف مرة اخرى فسوف يحذف المحتوى القديم
    'r+': يقوم بالقراءة على الملف والكتابة ايضا
    'w+': يقوم بالكتابة والقراءة كذلك وله خصائص اكثر من r+
    'a+': يقوم بالكتابة من دون حذف المحتوى السابق ويقوم كذلك بالقراءة
```

2- نقوم بكتابة الأمر فيما اذا كنا نريد أن نقرأ أو نكتب
```
f.read()  للقراءة
f.write()  للكتابة
```

3- تقوم بإغلاق الملف
```
f.close()
```

In [154]:
```
path = 'C:/Users/User/Desktop/test.txt' # ضع مكان الملف الذي تريد فتحه هنا
f = open(path,'a+') # 'r','r+','w',w+' جرب مع
f.write('I Study Physics')
f.close()
```

In [155]:
```
f = open('C:/Users/User/Desktop/test.txt','a+') # 'r','r+','w',w+' جرب مع
data = f.read()
f.close()

print data  # لاحظ انه قرأ الملف الذي كتبته
```

```
I Study Physics
```

In [156]:
```
f = open('C:/Users/User/Desktop/test.txt','w+') # 'r','r+','w',w+' جرب مع

data = ''
for i in range(20):
    data+= str(i) + str(i**2).rjust(5)
    data+= '\n'

f.write(data)
f.close()
```

In [ ]: