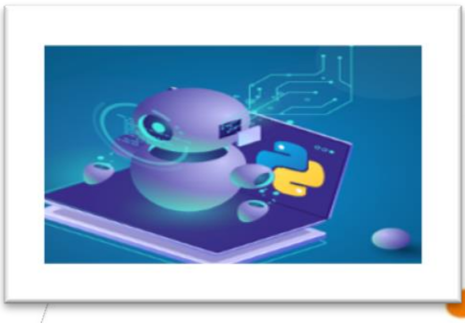
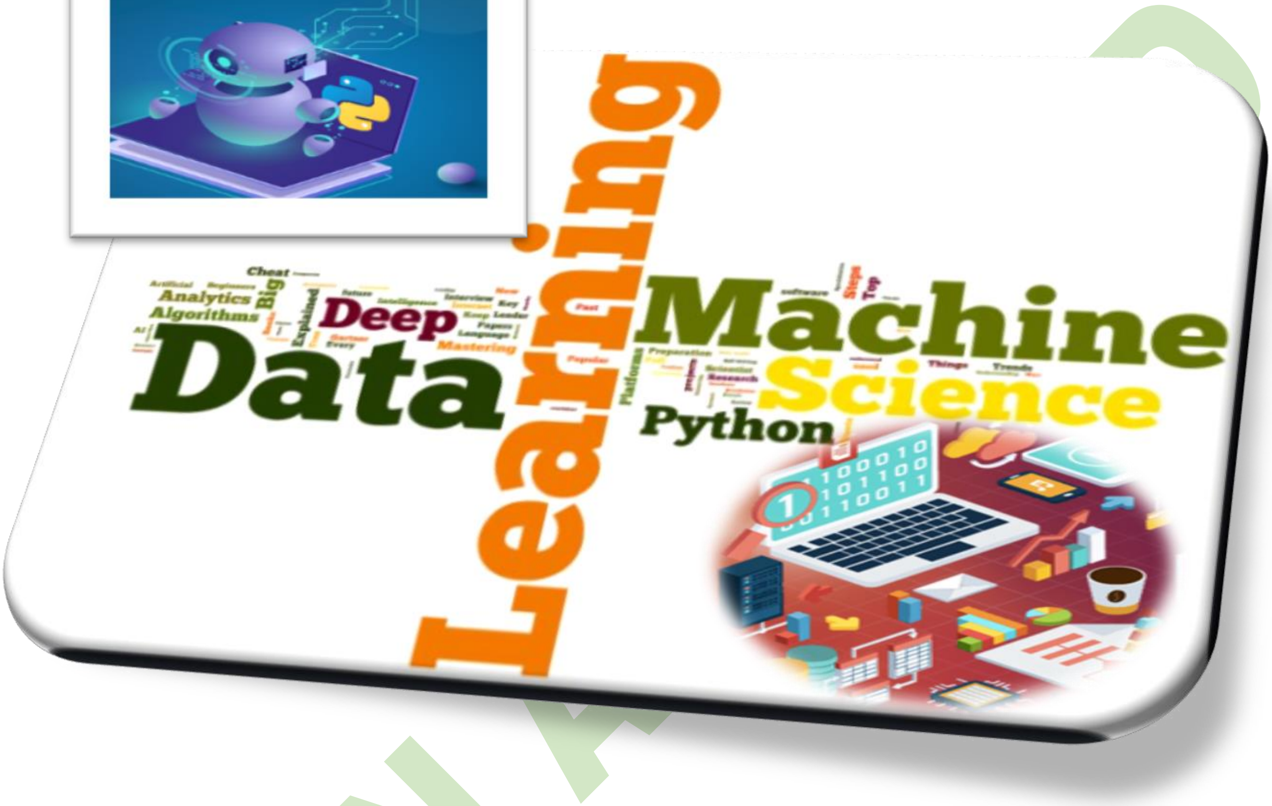
Data science and Machine learning with Python

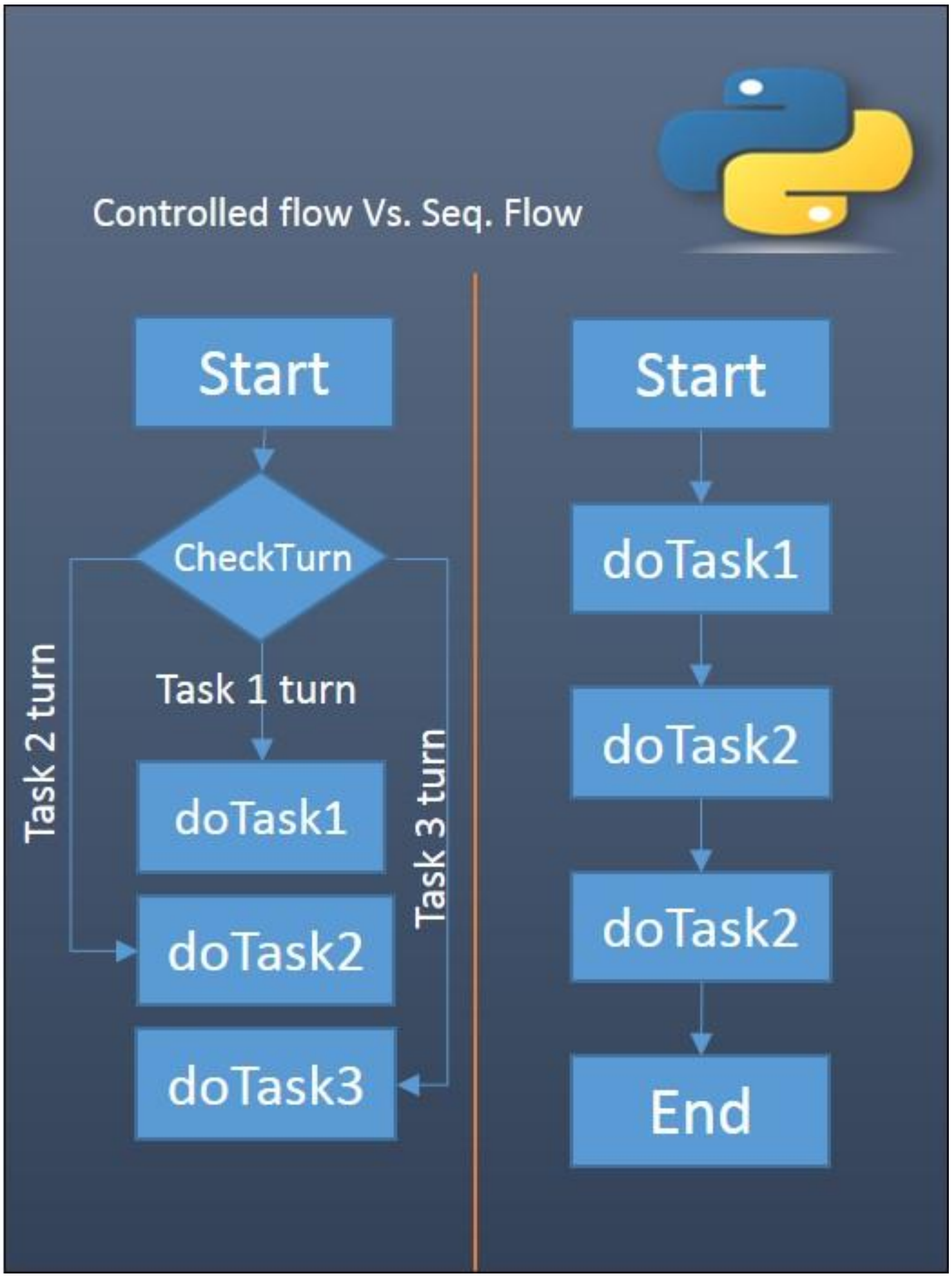




Designed by Abdur Rahman Joy - MCSD, MCPC, MCSE, MCTS, OCJP, Sr. Technical Trainer for VFX at IDB BISW (Scholarship program), and C#.net, R, Scala, Kotlin, JAVA, Android/IOS/Windows Mobile Apps, SQL server, Azure, Oracle, SharePoint Development, AWS , CEH, KALI Linux, Python, Data Science, Machine Learning ,Software Testing, Graphics, Multimedia and Game Developer at Joy Infosys and other premises like BITM, SkillsJob, PNTL, Leads Training and New Horizon inc , Cell #: +880-1712587348, email: jspaonline@gmail.com. Web URL: http://www.joyinfosys.com/me.

Control Flow:

The first word is **control** that simply means controlling. We don't want the default behavior, we want different one. We are getting the different behavior by controlling some aspects of the behavior. Now it comes to **flow,** Flow is just a way or sequence of program execution. By default every statement of program is executed one by one in an order they appear in a program code. When we combine the above two words we get **control flow,** That simply means controlling the flow of program execution to get desire behavior or result. Using control flow we are controlling the statement execution, Now program will no longer be executing in sequence, the execution is controlled by control tools. To understand it let's take few examples, In a bank management program we don't want to allow the retrieve function to work if the money in an account is zero. In that case we need to skip the retrieve program code and that is control flow.



Designed by Abdur Rahman Joy - MCSD, MCPC, MCSE, MCTS, OCJP, Sr. Technical Trainer for VFX at IDB BISW (Scholarship program), and C#.net, R, Scala, Kotlin, JAVA, Android/IOS/Windows Mobile Apps, SQL server, Azure, Oracle, SharePoint Development, AWS , CEH, KALI Linux, Python, Data Science, Machine Learning ,Software Testing, Graphics, Multimedia and Game Developer at Joy Infosys and other premises like BITM, SkillsJob, PNTL, Leads Training and New Horizon inc , Cell #: +880-1712587348, email: jspaonline@gmail.com. Web URL: http://www.joyinfosys.com/me.

Control flow tools in python

Python provide various tools for flow control. Some of them are if , if .. elif .. else, if..else, while ,for , switch, pass, range, break, else, continue, function etc. In this article I'll be covering only if-else and loops.

If – If this is the case **then** do this

This control statement indicate that *if something happens then do this* . It's a good way of handling some short conditions. An if block can be followed by zero or any number of else block.

Syntex:

```
if (condition):
   statements…
```

Demo:

a = 5 b = 10

```
if a < b:

          print("a is smaller than b")
```

Demo

a = 5 b = 10

```
if a < b and a > 0:

          print("a is smaller than b but greater than 0")
```

Note: Use of colon (":") in python is same as we use brackets in java or c++. Python uses colon and indentation for defining the scope or code block. So if you are getting an error like the following picture then correct your code indentation.

If … else

It's like **if** have money **then** spend **else** wait for salary. I hope it's clear from the previous line what this statement means. It's like if the conditions of if block is evaluated to true then execute that block else execute the else block. The else block is optional and one if can have any number of else blocks.

Syntax:

```
if (condition):
   statements…
else:
   default statements…
```

Demo:

```
a = 10 b = 5

if a < b:

        print("a is smaller than b")

else:

        print("a is greater than b")
```

If … elif … else

Basically that statement can replace switch statement. You can put any number of elif blocks after if and else block is optional.

Syntax:

```
if (option1 condition):
   option1 statements…
elif(option2 condition):
   option2 statements…
elif(option3 condition):
   option3 statements…
else:
   default option statements…
```

Demo:

```
a = 10
if a == 0:
        print("Zero")
elif a > 0:
        print("Positive number")
else:
        print("Negative number")
```

For statement

It is used for looping over a sequence. Python doesn't supports old for loop or c-style for loop. In traditional style for loop we have one variable which iterates over a sequence and we can change the value of sequence and variable as well but in modern for loop we have iteration variable that iterates over a fixed sequence. We can not change the sequence as well as iteration variable during iteration.

Syntax:

```
for iterator_name in iterating_sequence:
    …statements…
```

Demo:

```
a = ['cat', 'window', 'defenestrate']
for x in a:

    print x, len(x)
```

If you want to modify the sequence then we need to create the copy of the original sequence before doing this.

Using range

Sometimes it is required that we just want to iterate over number sequence like 1,2,3,4,… To solve this purpose python provides range function which generate the arithmetic progression with number of terms equal to the parameter passed in it. We have 3 variations of range() function. One take only

Syntax:

```
1. for iterator_name in range(10):
    …statements…
2. for iterator_name in range(start,end):
    …statements…

3. for iterator_name in range(start,stop,increment):
    …statements…
```

Demo:

```
for i in range(1, 5):

    print(i)
else:
    print('The for loop is over')
```

Break, Continue in For loop

Break is used for terminating the loop abnormally. i.e that even the sequence is not completed but loop is exited.

Continue is used for continuing to next iteration of loop without doing anything inside the loop.

Else is introduced in python and it is placed in loop without if. It will execute only if the loop is terminated without break.

Note: there are two more else statement, one is for **if** that I already explained and other is with try. The difference between try else block and loop else is try else block executes when no code is executed and loop else executes when no break is executed.

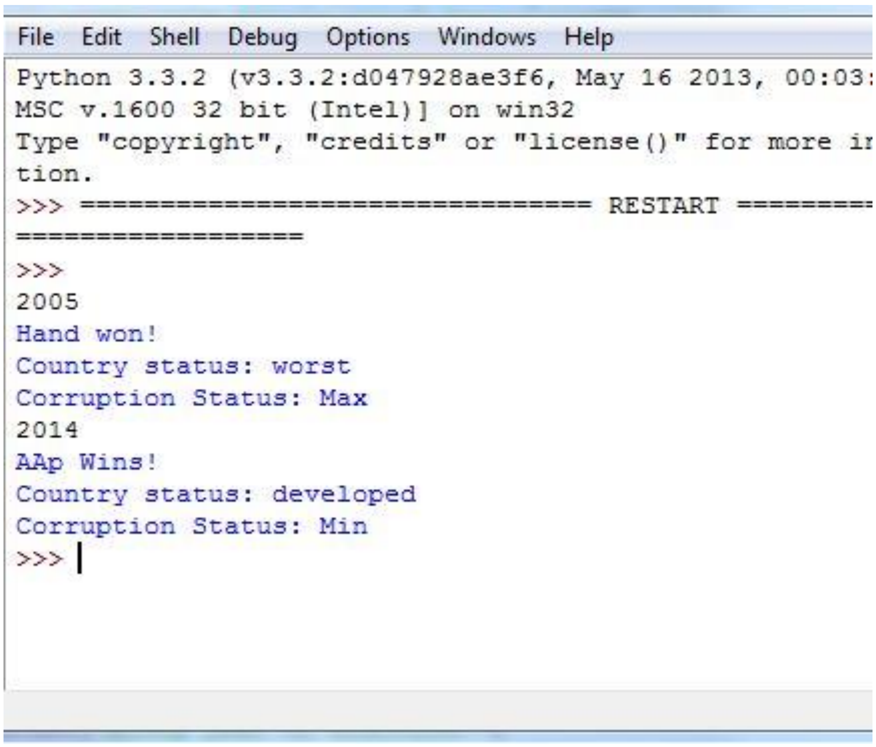
```
for i in range(10):
    if (i==5):
        continue
    print(i)
print('\n')

for i in range(10):
    if (i==5):
        break

            print (i)
```

Example

```
politicleParties=['AAP','Elephent','Hand','Rest']
electionYear=["2014","2009","2005","2001"]
countryStatus=["worst","developing","developed"]
corruptionStatus=["Max","Normal","Min"]
for party in politicleParties:
    year=input()
    if year in electionYear:
        if year == "2014":
            print("AAp Wins!")
            print("Country status: "+countryStatus[2])
            print("Corruption Status: "+corruptionStatus[2])
            break
        elif year == "2009":
            print("Hand Won :(")
            print("Country status: "+countryStatus[0])
            print("Corruption Status: "+corruptionStatus[0])
            continue
        elif year == "2005":
            print("Hand won!")
            print("Country status: "+countryStatus[0])
            print("Corruption Status: "+corruptionStatus[0])
        elif year == "2001":
            print("Rest Won!")
    else:
        print("Wrong year of election!")
else:
    print("The above loop was just for demonstration purpose!")
```



```
Python 3.3.2 (v3.3.2:d047928ae3f6, May 16 2013, 00:03:
MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more ir
tion.
>>> ================================ RESTART =========
==================
>>>
2005
Hand won!
Country status: worst
Corruption Status: Max
2014
AAp Wins!
Country status: developed
Corruption Status: Min
>>> |
```

while loop

As in other programming languages the while loop is an entry control iterative looping statement. The difference lies in the fact that the inner statements are written with an indentation of one 'tab' space. A simple program is illustrated here and the output also is shown.

```
x = 5
while x > 0:
print (x)
#all the print statement must be in parenthesis for version 3.4.0 x = x - 1 #the
algebra need not be done within the parenthesis

x = 5 y = x
while y > 0:
      print (y) y = y - 1
else:
      print (x)
```

Demo:

```
while True:
    s = input('Enter something : ')
    if s == 'quit':
        break
    print('Length of the string is', len(s))
print('Done')
```

Demo:

```
while True:
    s = input('Enter something : ')
    if s == 'quit':
        break
    if len(s) < 3:
        print('Too small')
        continue
    print('Input is of sufficient length')
```