PYTHON

The *if* statement

Syntax:

```
if expression:
        statement(s)
```

e.g.

```
a=9
if a>5:
        print "Statement is true"
```

The *else* Statement:

E.g.

```
a=9
if a>5:
    print “Statement is true”
else:
    print “Statement is false”
```

The *elif* Statement

e.g.

```
a, b, c = 20, 10, 30

if a>b and a>c:
    print “a is greater”
elif b>a and b>c:
    print “b is greater”
else:
    print “c is greater”
```

Range() Function

Range(start, end, increment/decrement)

Example :

```
x=range(5)
Print x
```

The *while* Loop

Syntax:

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

e.g.

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

The *for* Loop

Syntax:

for variable **in** sequence:

    statements(s)

e.g.

```
for i in 'Python':
    print 'Current Letter :', i


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

4.

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

5.

1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

6.

1 1 1

2 4 8

3 9 27

4 16 64

Loop controls: break and continue

The *break* Statement:

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

Continue statement cont..

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

Accessing Characters or Strings

h="hello world"

1. h[0]
2. h[-1]
3. h[-3]
4. h[-7:-2]
5. h[-2:]
6. h[:]

h[2]='w' //assigning value.. Will it work or not ?

```
h="hello world"
s=h.upper()
print s              //output ??

s=h.lower()
print s              //output ??


j="this is, my name"
j.split(',')                  //output ??

b=j.split('m')
print b              //output ??

k='hello'
k.replace('l', 't')          //output ??
```

Functions

- Function Definition
- Function Calling

Function cont..

Syntax:

```
def func():
        statements

func()
```

Example:

```
def printme( ):
        print “Hello this is a function
   calling”

printme()
```

Passing functions onto a function

```
def hello():
    print “hi”
def how(f):
    print “bye”
    f()
how(hello)
```

Output:

First how will work
The function hello will be stored in f

“bye”
“hi”

```
def hello():
    print “hi”
def how(f):
    def f1():
        print “bye”
        f()
    return func

b=how(hello)
b()
```

Output:

First b will call how
Then f1 will work and will print “bye”
Then f will be called from there
And “hi” will be printed.

“bye”
“hi”

Armstrong Number

```
Number=input("Enter the number : ")
Temp=number
sum=0

while(temp>0):
    rem=temp%10
    sum+=rem**3
    temp=temp/10

If(number==sum):
    print("Armstrong number")
else:
    print("Not an Armstrong Number")
```

For example : 153

Prime Numbers in Range

```
a=input("Enter starting range : ")
b=input("Enter ending range : ")
c=0
for n in range(a, b+1):
    p=True
    if n>1:
        for i in range(2, n):
            if(n%i==0):
                p=False
    if p==True:
        print n
        c+=1
print "Prime Numbers : "%c
```

Factorial

```
n=input("Enter the Number to find Factorial
: ")
a=1

for i in range(1, n+1):
    a=a*i
print a
```

Difference between range and xrange

The functioning of both are same… but the only difference is in their types.

Range - List

Xrange - xrange

Classes

Syntax:

```
class class_name:
        statement_1
        .
        .
        statement_n
```

Example of a Class and it's Object

```
class hello():
        def add(self, a,b):
                print "sum = ", a+b

H=hello()

H.add(3,2)
```

Class Inheritance

```
class A():
        n=10
        def __init__(self):
                print “Hey..! I’m a base class Constructor”
        def hello(self):
                print “Hello”
        def bye(sef):
                print “bye”
class B(A):
        def __init(self):
                print “Hey..! I’m a sub-class Constructor”
        def how(self):
                print “How are you..? “
b=B()
b.how()
b.hello()
b.bye()
print b.n
```

Data Overriding

```
class A():
        n=10
        def __init__(self):
                print “Hello”
        def hello(self):
                print “How are you..?”

class B(A):
        def __init(self):
                print “Hey”
        def hello(self):
                print “I’m good.“
a=A()
b=B()

a.hello()                       //hello in A()

b.hello()                       //hello in B()        //function overrides.
```

Regular Expression

1. **Search()**
2. **Match()**
3. **Substitute()**

Searching

```
import re

line="this is an example of Regular Expression"

m=re.search(r'e\w+e', line)
print m.group()

m=re.search(r'm\w+e', line)
print m.group()

M=re.search(r'E\w+$', line)          //reverse searching
print M.group()

m=re.search(r'.*', line)                    //to search
entire line
```

Matching

```
import re

line="this is an example of Regular Expression"

m=re.match(r't\w+', line)
print m.group()                    //it will only match the initial
word of the line

m=re.match(r't\w+\s\w+', line)
print m.group()                    //to find the first two words
                                   //  \s is used for spaces

m=re.match(r'T\w+s', line, re.I)                    //to
ignore case-sensitivity
print m.group()
```

Substitute

```
import re

line="this is a match function match and
match"

m1=re.sub(r'match', 'not', line, 2)

if m1:
    print m1
else:
    print 'no match'
```

Thank You