interation loops

- 2types in python
  - for

```
#syntax in c
for (i=0<con;i++/i--)
{
   stmts(conditions)
}
# syntax in python
for i in range(n):
   stmts/conditions
```

  - while

```
while con:
stmts
or
initialisation
condition
   inc/dec
```

In [3]:
```
dir(__builtins__)
```

Out[3]:
```
['ArithmeticError',
 'AssertionError',
 'AttributeError',
 'BaseException',
 'BlockingIOError',
 'BrokenPipeError',
 'BufferError',
 'BytesWarning',
 'ChildProcessError',
 'ConnectionAbortedError',
 'ConnectionError',
 'ConnectionRefusedError',
 'ConnectionResetError',
 'DeprecationWarning',
 'EOFError',
 'Ellipsis',
 'EnvironmentError',
 'Exception',
 'False',
 'FileExistsError',
 'FileNotFoundError',
 'FloatingPointError',
 'FutureWarning',
 'GeneratorExit',
 'IOError',
 'ImportError',
 'ImportWarning',
 'IndentationError',
 'IndexError',
 'InterruptedError',
 'IsADirectoryError',
 'KeyError',
 'KeyboardInterrupt',
 'LookupError',
 'MemoryError',
 'ModuleNotFoundError',
 'NameError',
 'None',
 'NotADirectoryError',
 'NotImplemented',
 'NotImplementedError',
 'OSError',
 'OverflowError',
 'PendingDeprecationWarning',
 'PermissionError',
 'ProcessLookupError',
 'RecursionError',
 'ReferenceError',
 'ResourceWarning',
 'RuntimeError',
 'RuntimeWarning',
 'StopAsyncIteration',
 'StopIteration',
 'SyntaxError',
 'SyntaxWarning',
 'SystemError',
 'SystemExit',
```

```
 'TabError',
 'TimeoutError',
 'True',
 'TypeError',
 'UnboundLocalError',
 'UnicodeDecodeError',
 'UnicodeEncodeError',
 'UnicodeError',
 'UnicodeTranslateError',
 'UnicodeWarning',
 'UserWarning',
 'ValueError',
 'Warning',
 'WindowsError',
 'ZeroDivisionError',
 '__IPYTHON__',
 '__build_class__',
 '__debug__',
 '__doc__',
 '__import__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__',
 'abs',
 'all',
 'any',
 'ascii',
 'bin',
 'bool',
 'breakpoint',
 'bytearray',
 'bytes',
 'callable',
 'chr',
 'classmethod',
 'compile',
 'complex',
 'copyright',
 'credits',
 'delattr',
 'dict',
 'dir',
 'display',
 'divmod',
 'enumerate',
 'eval',
 'exec',
 'filter',
 'float',
 'format',
 'frozenset',
 'get_ipython',
 'getattr',
 'globals',
 'hasattr',
 'hash',
```

```
 'help',
 'hex',
 'id',
 'input',
 'int',
 'isinstance',
 'issubclass',
 'iter',
 'len',
 'license',
 'list',
 'locals',
 'map',
 'max',
 'memoryview',
 'min',
 'next',
 'object',
 'oct',
 'open',
 'ord',
 'pow',
 'print',
 'property',
 'range',
 'repr',
 'reversed',
 'round',
 'set',
 'setattr',
 'slice',
 'sorted',
 'staticmethod',
 'str',
 'sum',
 'super',
 'tuple',
 'type',
 'vars',
 'zip']
```

In [4]:
```
range(10)
```

Out[4]:
```
range(0, 10)
```

In [6]:
```
range(5,10)
```

Out[6]:
```
range(5, 10)
```

for loop

In [1]:
```
n=10
##for i in range(1,n):
for i in range(1,n+1):
    ##print(i,end=(","))
    print(i,end=(" "))
```

```
1 2 3 4 5 6 7 8 9 10
```

while loop

In [10]:
```
n=int(input('enter the input'))
i=0
while i<=n:
    print(i,)
    i=i+1
```

```
enter the input10
0
1
2
3
4
5
6
7
8
9
10
```

In [13]:
```
# reverse of a number 123   321
n=int(input('enter the value'))
rev=0
while n>0:
    #r=n%10
    #rev=rev*10+r
    rev=rev*10+n%10
    n=n//10
print(rev)
```

```
enter the value123456789
987654321
```

In [16]:
```
#given number is palindrome or not
n=int(input('enter the value'))
m=n
rev=0
while n>0:
    #r=n%10
    #rev=rev*10+r
    rev=rev*10+n%10
    n=n//10
if m==rev:
    print(m,'is palindrome')
else:
    print(m,'not palindrome')
```

```
enter the value123
123 not palindrome
```