while loop

In [2]:
```
n=int(input('enter an input'))
i=0
while i<=n:
    print(i,end=' ')
    i=i+1 #i +=1
```

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

In [6]:
```
# reverse of an number 123 - 321

rev=0
while n>0:
    r= n%10
n=int(input('enter value'))    rev=rev*10+r
    n=n//10
print(rev)
```

```
enter value12345
54321
```

In [12]:
```
# palindrome
n=int(input('enter value'))
m=n
rev=0
while n>0:
    rev= rev*10+n%10
    n=n//10
if m==rev:
    print(m,'is palindrome')
else:
    print('not palindrome')
```

```
enter value232
232 is palindrome
```

functions

- A function is a group of statements to do a specific task
- function breaks a code into small modules to look more organised
- code re-usability
- types of functonc
  - built-in functions
  - user defined fun

list of builtins

In [14]:
```
dir(__builtins__)
```

Out[14]:
```
['ArithmeticError',
 'AssertionError',
 'AttributeError',
 'BaseException',
 'BlockingIOError',
 'BrokenPipeError',
 'BufferError',
 'BytesWarning',
 'ChildProcessError',
 'ConnectionAbortedError',
 'ConnectionError',
 'ConnectionRefusedError',
 'ConnectionResetError',
 'DeprecationWarning',
 'EOFError',
 'Ellipsis',
 'EnvironmentError',
 'Exception',
 'False',
```

user defined functions

syntax in c

```
functions fname(){
    cond/stmts to excute
}

### syntax in python
def fname():
    cond/stmts
    return
fname()
```

- advantages
- making of large codes into small codes
- reuse of code in a function by calling its fname

In [16]:
```
a=4
b=10
sum([a,b])
```

Out[16]:
```
14
```

In [17]:
```
a=[1,2,3,4,5]
max(a)
```

Out[17]: 5

In [18]:
```
min(a)
```

Out[18]: 1

In [19]:
```
len(a)
```

Out[19]: 5

In [20]:
```
a='badrinath'
len(a)
```

Out[20]: 9

In [21]:
```
a='naveen'
min(a)
```

Out[21]: 'a'

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

Out[22]: 'v'

types of arguments in functions

- required arguments
- keywoard
- default arguments
- variables length

In [26]:
```
def add(a,b):
    c=a+b
    return c
a=int(input('enter a value'))
b=int(input('enter b value'))
add(a,b)
```

```
enter a value7
enter b value3
```

Out[26]: 10

In [27]:
```
# keyword arguments
def key(str):
    print(str)
key(str=123)
```

```
123
```

In [28]:
```
def keywoard(name,clz):
    print('name:',name)
    print('clz:',clz)
keywoard(clz='Aits',name='abc')
```

```
name: abc
clz: Aits
```

In [33]:
```
# default arguments
def default(l,r=1):
    print(1,r)
default(l='11',r='a')
default(l='13')
```

```
1 a
1 1
```

In [39]:
```
# n odd numbers using functions
n=int(input('enter value'))
def odd(n):
    for i in range(1,n+1):
        if i%2 !=0:
            print(i,end=' ')
    return
odd(n)
```

```
enter value5
1 3 5
```

prime or not

In [48]:
```
n=int(input('enter a number'))
def prime(n):
    c=0
    for i in range(1,n+1):
        if n%i==0:
           c=c+1
    if c==2:
        print(n,'is prime')
    else:
        print('not a prime')
prime(n)
```

```
enter a number19
19 is prime
```

In [ ]:

In [ ]: