# List comprehension:

-- List comprehension are used to create new list from other itrerable.

-[]

-syntax :[expression for item in list]

In [1]:

```
### By using list comprehension we can generate 1 to 20 natural numbers.
l1=[i for i in range(1,21)]
l1
```

Out[1]:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
```

In [5]:

```
###  To genrate 1 to 20 even number in between 1 to 10?
l2=[i for i in range(1,10) if i%2==0]
l2
```

Out[5]:

```
[2, 4, 6, 8]
```

In [6]:

```
### function to create to print odd number.
def odd(n):
    if n%2!=0:
        return n
l3=[i for i in range(1,10) if (odd(i))]
l3
```

Out[6]:

```
[1, 3, 5, 7, 9]
```

In [7]:

```
l2=[i for i in range(1,10) if i%2!=0]
l2
```

Out[7]:

```
[1, 3, 5, 7, 9]
```

In [8]:

```
k=[10,20,13,7,5,19,24]
l4=[i for i in k if(odd(i))]
l4
```

Out[8]:

```
[13, 7, 5, 19]
```

# Dictionary Comprehension

-syntax:{key:value for var in iterable}

In [11]:

```
d={"name":'avinash','empid':12324}
d
```

Out[11]:

```
{'name': 'avinash', 'empid': 12324}
```

In [12]:

```
#To genrate square of  a number in between 1 to 5 using dictionary comprehension
'''input
       n=5
    output:{1:1,2:4,3:9}'''
d={i:i*i for i in range(1,6)}
d
```

Out[12]:

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

In [13]:

```
L=['apple','banana','mango']
d2={ item:len(item) for item in L}
d2
```

Out[13]:

```
{'apple': 5, 'banana': 6, 'mango': 5}
```

# Lambda:

- It is a anonymous function or single line function
  - syntax: lambda variable_name:expression

In [15]:

```
def add(a,b):
    return a+b
add(10,20)
```

Out[15]:

```
30
```

In [16]:

```
p=lambda a,b:a+b
p(10,20)
```

Out[16]:

```
30
```

In [19]:

```
k1=[i for i in range(1,10) if (lambda i:i%2!=0)]
k1
```

Out[19]:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

# Filter:

- filter(function,iterables)

In [21]:

```
li=[2,-4,-1,5,7,-8]
f1=list(filter(lambda i:i<0,li))
f1
```

Out[21]:

```
[-4, -1, -8]
```

# generator:

-yelid

In [24]:

```
def natural(n):
    n=1
    while n!=20:
        yield n
        n=n+1
list(natural(20))
```

Out[24]:

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

# map:

- map(function,iterables)

In [25]:

```
# int a,b:
#scanf("%d%d",&a,&b);
a,b=map(int,input().split())
print(a,b)
```

```
10 20
10 20
```

In [ ]:

```

```