Lambda function

- lambda function is aslo a function
- It is a kind of single line function
- we already seen list comprehension(single line)
- lambda function gives a felxibility to stop writing many lines
- on iterative excuetion it will asave the time

In [2]:
```
def summ(num):
    return (num+10)

summ(20)
```

Out[2]: 30

$pattern - 1$

- function name
- variable name
- return output

In [ ]:
```
# <function_name>= lambda <variable_name>:<return_output>
```

In [5]:
```
summ=lambda num:num+10
summ(20)
```

Out[5]: 30

In [6]:
```
def cube(x):
    return(x*x*x)
cube(10)
```

Out[6]: 1000

In [9]:
```
cube= lambda x : x*x*x
cube(10)
```

Out[9]: 1000

$pattern - 2$

Two arguments

In [ ]:
```
<function_name>=lambda <arg1>,<arg2>:<output>
```

In [10]:
```
def add(n1,n2):
    return(n1+n2)
```

```
add(10,20)
```

Out[10]: 30

In [12]:
```
add= lambda n1,n2:n1+n2
add(10,20)
```

Out[12]: 30

In [18]:
```
average= lambda n1,n2,n3:(n1+n2+n3)/3
average(10,200,3000)
```

Out[18]: 1070.0

In [20]:
```
average= lambda n1,n2,n3:round((n1+n2+n3)/3,2)
average(1021,200,3000)
```

Out[20]: 1407.0

$pattern - 3default arguments$

In [22]:
```
avg=lambda n1,n2,n3=100:round((n1+n2+n3)/3,2)
avg(10.5,200)
```

Out[22]: 103.5

In [21]:
```
average= lambda n1=10,n2=100,n3=20.5:round((n1+n2+n3)/3,2)
average()
```

Out[21]: 43.5

$pattern - 4$ **if**

In [ ]:
```
# <fun_name>=lambda <arg>: <if_output> <if-condition> <else> <else_output>
```

In [26]:
```
def greet(n1):
    if n1>10:
        return('hello')
    else:
        return('hai')
greet(20)
```

Out[26]: 'hello'

In [35]:
```
greet= lambda n1 : 'hello' if n1>10  else 'hai'
greet(20)
```

Out[35]: 'hello'

In [38]:
```
# greatest number with lambda function
greatest= lambda n1=10,n2=20: f'{n1} is big' if n1>n2  else f'{n2} is big'
greatest()
```

Out[38]: '20 is big'

$pattern - 5$

using list

In [ ]:
```
list1=['hyd','blr','chennai']
# op=list2=['Hyd','Blr','Chennai']

'hi how are you' ==== > 'Hi How Are You'   # title
                 ===== > 'Hi how are you'  # capitalize
                 ====== > 'HI HOW ARE YOU'  # upper
                 ====== > 'hi how are you'  # lower
                 ====== > '***hi how are you**' #centre
```

In [39]:
```
list1=['hyd','blr','chennai']
list2=[]
for i in list1:
    list2.append(i.capitalize())
list2
```

Out[39]:
```
['Hyd', 'Blr', 'Chennai']
```

In [ ]:
```
- what is the variable name ==> i

- waht s the return output  ==> i.capitalize

- from where it is coming  ==>list1


lambda <variable_name> : <output>,<iteratable>

- strings and list are iterable
```

In [50]:
```
lambda i : i.capitalize(),list1
```

Out[50]:
```
(<function __main__.<lambda>(i)>, ['hyd', 'blr', 'chennai'])
```

map

In [52]:
```
list(map(lambda i : i.capitalize(),list1))
```

Out[52]:
```
['Hyd', 'Blr', 'Chennai']
```

In [ ]:
```
# step-1 create a normal lambda function
   # lambda<arg>:<output>
#- step-2: add iterable
   #  lambda<arg>:<ouput>,<list>
# step-3: map the both function and list
   # map(lambda<arg>:<output>,<list>)
# step-4:save the result in list or tuple
   # list(map(lambda<arg>:<ouput>,<list>))
```

In [54]:
```
list(map(lambda i : i.capitalize(),list1))
```

Out[54]:
```
['Hyd', 'Blr', 'Chennai']
```

In [55]:
```
lambda i: i.title()
lambda i: i.title(),list1
map(lambda i: i.title(),list1)
list(map(lambda i: i.title(),list1))
```

Out[55]: ['Hyd', 'Blr', 'Chennai']

In [56]:
```
lambda i : i.upper()
lambda i : i.upper(),list1
map(lambda i : i.upper(),list1)
list(map(lambda i : i.upper(),list1))
```

Out[56]: ['HYD', 'BLR', 'CHENNAI']

In [58]:
```
lambda i : i.lower()
lambda i : i.lower(),list1
map(lambda i : i.lower(),list1)
list(map(lambda i : i.lower(),list1))
```

Out[58]: ['hyd', 'blr', 'chennai']

In [67]:
```
list1 = ['hyd','blr','chennai']
print(list(map(lambda i: i.capitalize(), list1)))
print(list(map(lambda i: i.upper(), list1)))
print(list(map(lambda i: i.lower(), list1)))
print(list(map(lambda i: i.center(10,"*"), list1)))
```

```
['Hyd', 'Blr', 'Chennai']
['HYD', 'BLR', 'CHENNAI']
['hyd', 'blr', 'chennai']
['***hyd****', '***blr****', '*chennai**']
```

In [2]:
```
list1=['hyd','blr','chennai']
lambda i : i.center(10,'*')
lambda i : i.center(10,'*'),list1
map(lambda i: i.center(10,'*'),list1)
list(map(lambda i: i.center(10,"*"),list1))
```

Out[2]: ['***hyd****', '***blr****', '*chennai**']

In [64]:
```
list1=['hyd','blr','che#nnai','mumb#ai']
# list2=['che#nnai','mumb#ai']
for i in list1:
    if '#' in i:
        list2.append(i)
list2
```

Out[64]: ['Hyd', 'Blr', 'Chennai', 'che#nnai', 'mumb#ai']

In [65]:
```
lambda i: '#' in i,list1
list(map(lambda i: '#' in i,list1))
```

Out[65]: [False, False, True, True]

filter method

In [66]:
```
list(filter(lambda i : '#' in i ,list1))
```

Out[66]: ['che#nnai', 'mumb#ai']

In [69]:
```
list1=['hyd','chennai','blr','mumbai']
list2=['hyd','blr'] # len(i)<3
```

```
list(filter(lambda i: len(i)<=3,list1))
```

Out[69]:
```
['hyd', 'blr']
```

In [ ]:
```

```