In [ ]:
```
# Create basic function
```

one argument

In [1]:
```
def add(x):
    return(x+10)

add(20)
```

Out[1]: 30

In [2]:
```
def add(x):
    summ=x+10
    return(summ)

add(20)
```

Out[2]: 30

In [ ]:
```
def add(x):
    return(x+10)

add(20)


# how many arguemts are present? : x
# what you are returning?: x+10

# format: lambda <argument_name>:<output>
```

In [3]:
```
add=lambda x:x+10
add(20)
```

Out[3]: 30

In [ ]:
```
def square(x):
    return(x*x)
```

In [4]:
```
square=lambda a:a*a
square(5)
```

Out[4]: 25

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

In [5]:
```
cube=lambda x:x*x*x

cube(30)
```

Out[5]: 27000

two arguments

In [ ]:
```
def add(a,b):
    return(a+b)

add(20,30)


# how many arguemts are present? : a,b
# what you are returning?: a+b

# format: lambda <arg1>,<arg2>:<output>
```

In [6]:
```
add=lambda a,b:a+b
add(20,30)
```

Out[6]: 50

In [7]:
```
# implement average of 3 numbers using lambda
avrg=lambda a,b,c:(a+b+c)/3
avrg(5,5,5)
```

Out[7]: 5.0

In [10]:
```
# implemet avergae make c as default parameter
avg=lambda a,b,c=30: round((a+b+c)/3,2)
avg(20,30)
```

Out[10]: 26.67

In [11]:
```
round((a+b+c)/3,2)
```

Out[11]: 26.67

if-else

In [12]:
```
# Create a function for finding greater number between two numbers
def greater(n1,n2):
    if n1>n2:
        return(n1)
    else:
        return(n2)

greater(100,200)
```

Out[12]: 200

In [13]:
```
l1=[]
def greater(n1,n2):
    if n1>n2:
        l1.append(n1)
    else:
        l1.append(n2)

greater(100,200)
```

In [14]:
```
l1=[<if_output> <if_con> else <else_op> <loop>]
```

Out[14]:
```
[200]
```

In [ ]:
```
def greater(n1,n2):
    if n1>n2:
        return(n1)
    else:
        return(n2)

greater(100,200)

# format :
# lambda <arg1>,<arg2>: <if_output> <if_con> else <else_op>
```

In [15]:
```
greater=lambda a,b: (a if a>b else b)
greater(8,3)
```

Out[15]:
```
8
```

- lambda function is nothing but create a function
- one argument
- multiple arguments
- if else conditions
- if else conditions same like list compehenshion

In [2]:
```
list1=['hyd','mumbai','chennai']

#output: ['Hyd','Mumbai','Chennai']

# M-1: use append method

list2=[]

for i in list1:
    list2.append(i.capitalize())

print(list2)

# M-2: use list comprhenshion
[i.capitalize() for i in list1]

# M-3: make a lambda function
```

```
['Hyd', 'Mumbai', 'Chennai']
```

Out[2]:
```
['Hyd', 'Mumbai', 'Chennai']
```

In [ ]:
```
lambda <arguments>: <output>
```

In [ ]:
```
# whenever you use iterations

# iterator: some thing can be iterbale/ you can print using for loop

# list ,string, tuple, dictionary
```

In [ ]:
```
lambda <arguments>: <output>,<iterator>

[i.capitalize() for i in list1]
```

In [3]:
```
list1=['hyd','chennai','mumbai']
lambda i:i.capitalize(),list1
```

Out[3]:
```
(<function __main__.<lambda>(i)>, ['hyd', 'chennai', 'mumbai'])
```

In [ ]:
```
lambda <arg>:<output>
```

In [ ]:
```
- next thing is map input and output
```

In [4]:
```
list1=['hyd','chennai','mumbai']
map(lambda i:i.capitalize(),list1)
```

Out[4]:
```
<map at 0x1979c06cfd0>
```

In [ ]:
```
- store the output
```

In [5]:
```
list(map(lambda i:i.capitalize(),list1))
```

Out[5]:
```
['Hyd', 'Chennai', 'Mumbai']
```

- first make a lambda function
- second add your iterator
- map both function and iterator
- save the result in a list

In [ ]:
```
list1=['hyd','chennai','mumbai']
lambda i:i.capitalize(),list1
map(lambda i:i.capitalize(),list1)
list(map(lambda i:i.capitalize(),list1))
```

In [8]:
```
list1=[1,2,3,4,5]
# [1,4,9,16,25]


list(map(lambda i:i*i,list1))

for i in map(lambda i:i*i,list1):
    print(i)
```

```
1
4
9
16
25
```

In [10]:
```
list1=[1,2,3]
list2=[11,22,33]

# [12,24,36]

for i,j in zip(list1,list2):
    print(i+j)
```

```
12
24
36
```

In [15]:
```
str(map(lambda i,j:i+j, list1,list2))
```

Out[15]:
```
'<map object at 0x000001979C06CC40>'
```

In [16]:
```
map(lambda i,j:i+j, list1,list2)
```

Out[16]:
```
<map at 0x1979c06dcc0>
```

In [18]:
```
str([])
```

Out[18]:
```
'[]'
```

In [ ]:
```
i.capitalize()
i*i
i+j
```

In [20]:
```
list1=['h#d','mum#bai','chennai']
#['h#d','mum#bai']

list1=['h#d','mum#bai','chennai']
list2=[]
for i in list1:
    if '#' in i:
        list2.append(i)
print(list2)


[i for i in list1 if '#' in i]
```

```
['h#d', 'mum#bai']
```

Out[20]:
```
['h#d', 'mum#bai']
```

In [25]:
```
#lambda <argument>:<condition>,<iterator>
```

Out[25]:
```
(<function __main__.<lambda>(i)>, ['h#d', 'mum#bai', 'chennai'])
```

In [28]:
```
list1=['h#d','mum#bai','chennai']
list(map(lambda i: '#' in i,list1))

# condition mapping to list of items
```

Out[28]:
```
[True, True, False]
```

In [31]:
```
list1=['h#d','mum#bai','chennai']
list(filter(lambda i: '#' in i,list1))
```

Out[31]:
```
['h#d', 'mum#bai']
```

In [30]:
```
list1=['h#d','mum#bai','chennai']
'#' in 'h#d'
'#' in 'mum#bai'
```

Out[30]:
```
True
```

In [ ]: