functions

In [ ]:
```
# definition of function

def sayhi():
    print("hiiii hello everyone welcome to my session ")
```

In [ ]:
```
# calling a function
sayhi()
```

In [ ]:
```
# generalised function with user input
def addition(n1,n2):
    a = n1+n2
    return a
```

In [ ]:
```
# calling a function

result = addition(20,250)
print(result)
```

In [ ]:
```
# function parameter order

def areaofcircle(radius,pi):
    return (pi*radius**2)
```

In [ ]:
```
# calling a function  using proper order
areaofcircle(10,3.14)
```

In [ ]:
```
# calling a function  using wrong order
areaofcircle(3.14,10)
```

In [ ]:
```
# using default arguements

def areacircle(radius,pi = 3.14):
    return (pi*radius**2)
```

In [ ]:
```
# calling
areacircle(10)
```

In [ ]:
```
# parameters vs arguments :
# parameters are used while defining function
# arguments are used while calling function
```

In [ ]:
```
# odd_even

def iseven(num):
    if num%2==0:
        return True
    else:
        return False
```

In [ ]:
```
iseven(10)
```

In [ ]:
```
# num : parameter
# 10 : argument
```

In [ ]:
```
# docstring : add comments / documentations to your function
# it is written in triple quotes

def iseven(num):
    """Return true if number is even else return false."""
    if num%2==0:
        return True
    else:
        return False
```

In [ ]:
```
# calling doc string from defined function

iseven.__doc__
```

In [ ]:
```

```

In [ ]:
```
# variable lenght of arguments
```

In [ ]:
```
def add(*args):
    return sum(args)
```

In [ ]:
```
add(1,3,5,10)
```

In [ ]:
```
def getmax(*args):
    return max(args)
```

In [ ]:
```
getmax(1,7,89,45)
```

In [ ]:
```
# global Vs local variable
```

In [ ]:
```python
var = 1 # global varibale

def ad(*nums):
    # total = local variable
    total = 0
    for i in nums:
        total = total + i
    return total
```

In [ ]:
```python
ad(1,2,3,4,5,6)
```

In [ ]:
```python
print(var) # global
```

In [ ]:
```python
print(total) # local
```

In [ ]:
```python

```

lambda

In [ ]:
```python
# find cube of number using normal def(function)

def cube(n):
    return (n*n*n)
```

In [ ]:
```python
cube(2)
```

In [ ]:
```python
# find cube of number using lamda
g = lambda x:x*x*x
```

In [ ]:
```python
g(3)
```

In [ ]:
```python
# square rot of number using lambda
(lambda a:a**0.5)(9)
```

In [ ]:
```python

```

In [ ]:
```python
def agecal(yob):
    return (2022-yob)
```

In [ ]:
```python
agecal(1990)
```

In [ ]:
```python

```

map()

In [ ]:
```
# give new alist of numbers, return new list with squares of ea
# normal method
nums = [1,2,3,4,5,6,7,8,9]
sq_lst = []

for i in nums:
    a = i**2
    sq_lst.append(a)

print(sq_lst)
```

In [ ]:
```
# map() : takes function and list as an input

nums = [1,2,3,4,5,6,7,8,9]
sq = list(map(lambda x:x**2,nums))
```

In [ ]:
```
sq
```

In [ ]:
```

```

reduce()

In [3]:
```
#reduce :it reduces input list using our logic , it takes funct

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

import functools
addition = functools.reduce((lambda x,y:x+y),nums)
print(addition)
```

```
45
```

filter()

In [12]:
```
# filter : it filters from list using defined condition

nums = [1,2,3,4,5,6,7,8,9]
evens = list(filter((lambda x:x%2==0),nums))
print(evens)
```

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

In [ ]:
```

```

In [13]:
```
names = ['ramesh','suresh','smith','john','nasar']
names_up = list(map(str.upper,names))
print(names_up)
```

```
['RAMESH', 'SURESH', 'SMITH', 'JOHN', 'NASAR']
```

In [ ]:
```
1
```

In [ ]:
```
1
```