In [1]:

```
# functions
```

In [ ]:

```
def function(name):
    fun_body
fun_calling
```

In [11]:

```
x = "Bansal sir"
def simpleinterest(p,t,r=2,name="Avanish"):
    global z
    z = 100
    res = (p*r*t)/100

    return res
    print(x)

a = 1500
b = 5
#c = 3
print(simpleinterest(a,b))
print(x)
```

```
150.0
Bansal sir
```

In [12]:

```
#recursion
```

In [13]:

```
def get_fact(num):
    if num==0:
        return 1
    elif num==1:
        return 1
    else:
        return num*get_fact(num-1)
print(get_fact(5))
```

```
120
```

In [14]:

```
#Lambda
```

In [ ]:

```
#var = lambda arguments:single expression
```

In [15]:

```
res = lambda p,r,t:(p*r*t)/100
print(res(1500,2,3))
```

```
90.0
```

In [ ]:

```
# map(function, *sequence)
# filter(function, *sequence)
```

In [16]:

```
li = [62,87,25,18,29,10]
print(list(map(lambda x:x%2==0, li)))
```

```
[True, False, False, True, False, True]
```

In [17]:

```
li = [62,87,25,18,29,10]
print(list(filter(lambda x:x%2==0, li)))
```

```
[62, 18, 10]
```

In [18]:

```
li = [62,87,25,18,29,10]
li2 = [56,87,12,55,54,78]
print(list(map(lambda x,y:x+y, li,li2)))
```

```
[118, 174, 37, 73, 83, 88]
```

In [21]:

```
li = [62,87,25,18,29,10]
li2 = [56,87,12,55,54,78]
print(list(map(lambda x,y:x%y==0, li,li2)))
```

```
[False, True, False, False, False, False]
```

In [22]:

```
x = (eval(i) for i in input("Enter spome nums:").split(' ')]
x
```

```
Enter spome nums:56 78 56
```

Out[22]:

```
[56, 78, 56]
```

In [23]:

```
x = 67,87,45,100
print(x)
print(type(x))
```

```
(67, 87, 45, 100)
<class 'tuple'>
```

In [28]:

```
class Student:
    y = "SAM"
    print(y)
    def st_details(self, name):
        x = "MCA"
        print(name)
        print(x)
        print(ob.y)

class teacher:
    print("Good Morning")
ob = Student()
ob.st_details("Avanish")
print(Student.y)
```

```
SAM
Good Morning
Avanish
MCA
SAM
SAM
```

In [31]:

```
class Student:
    y = "SAM"
    print(y)
    def __init__(self):
        pass
    def __init__(self, a,b):
        self.name = a
        self.age = b
    def st_details(self, name):
        x = "MCA"
        print(name)
        print(x)
        print(ob.y)
        print("Age is:",self.age)
obj = Student("Avi",26)
obj.st_details("Rohit")
print(obj.name)
```

```
SAM
Rohit
MCA
SAM
Age is: 26
Avi
```

In [9]:

```
class Student:
    y = "SAM"
    print(y)
    def __init__(self, a,b):
        self.__name = a
        self.__age = b
    def __stdetails(self, name):
        x = "MCA"
        print(self.__name)
        print(x)
        print(obj.y)
        print("Age is:",self.__age)
obj = Student("Avi",26)
obj._Student__stdetails(("Rohit"))
print(obj._Student__name)
```

```
SAM
Avi
MCA
SAM
Age is: 26
Avi
```

In [ ]:

```
#inheritance
```

In [15]:

```
class Student:
    y = "SAM"
    print(y)
    def __init__(self, a,b):
        self.p = a
        self.q = b
    def st_details(self, name):
        x = "MCA"
        print(name)
        print(x)
        print(ob.y)
        print(self.p)

class teacher(Student):
    def __init__(self,a,b,c):
        self.r = c
        super().__init__(a,b)

    print("Good Morning")
    def hello(self):
        print("Good Morning")
        print(self.p)
        print(self.r)
ob = teacher(100,200,300)
ob.st_details("Arohi")
ob.hello()
```

```
SAM
Good Morning
Arohi
MCA
SAM
100
Good Morning
100
300
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: