```
#Nested functions
def f1(): #outer function
    print ("Hello")
    def f2(): #inner function
        print ("world")
    f2()

f1()

def father():
    data=100
    print(data)
    def son():
            print(data)
    son()

father()




def f1(): #outer function
    x = 1 #variable defined in f1 function
    def f2(a): #inner function
        print (a+x) #able to acces the variable of outer function
    f2(2)



def f1(): #outer function
    x = 1 # variable defined in the outer function
    def f2(a): #inner function
       #will create a new variable in the inner function
       #instead of changing the value of x of outer function
        x = 4
        print (a+x)
    print (x) # prints the value of x of outer function
    f2(2)

f1()
```

```
# Output 1 6

def f1():
    a = 5
f1()

#a can't be accessed outside the function f1
print (a) # will give error



a = 5
def f1():
    print (a) # will print 5

f1()




a = 1
def f1():
    a = 5
    print (a) #will print 5

print (a) #will print 1

f1()




a = 1
def f1():
    global a #using global variable a
    a = 5 #global variable is changed to 5
    print (a) #will print 5

f1()

print (a) #will print 5
```

```
# Output  5 5


def f1(): #outer function
    a = 1
    def f2(): #inner function
        print (a) #will print 1
    f2()
f1()




def f1(): #outer function
    a = 1
    def f2(): #outer function
        a = 2
        print (a) #prints 2
    f2()
    print (a) #prints 1
f1()


def father():
    data=100
    print(data)
    print(x)
    def son():
        global g
        g=2000
          x=20
          print(data)
    son()
print(g)
father()


def f1(): #outer function
    a = 1
    def f2(): #outer function
```

```
        nonlocal a
        a = 2
        print (a) #prints 2
    f2()
    print (a) #prints 2
f1()
# Output  2 2
```