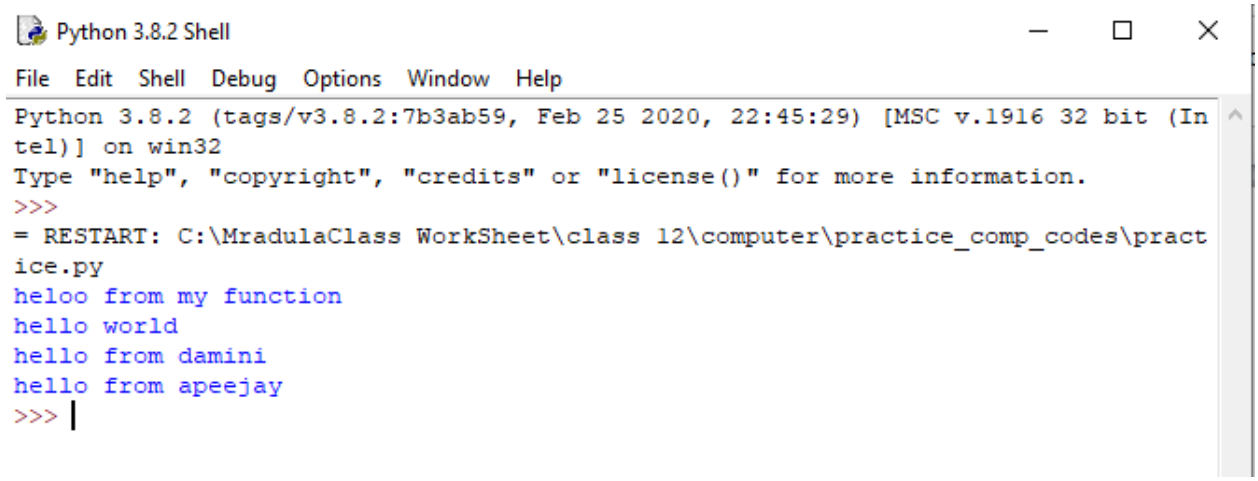
functions codes

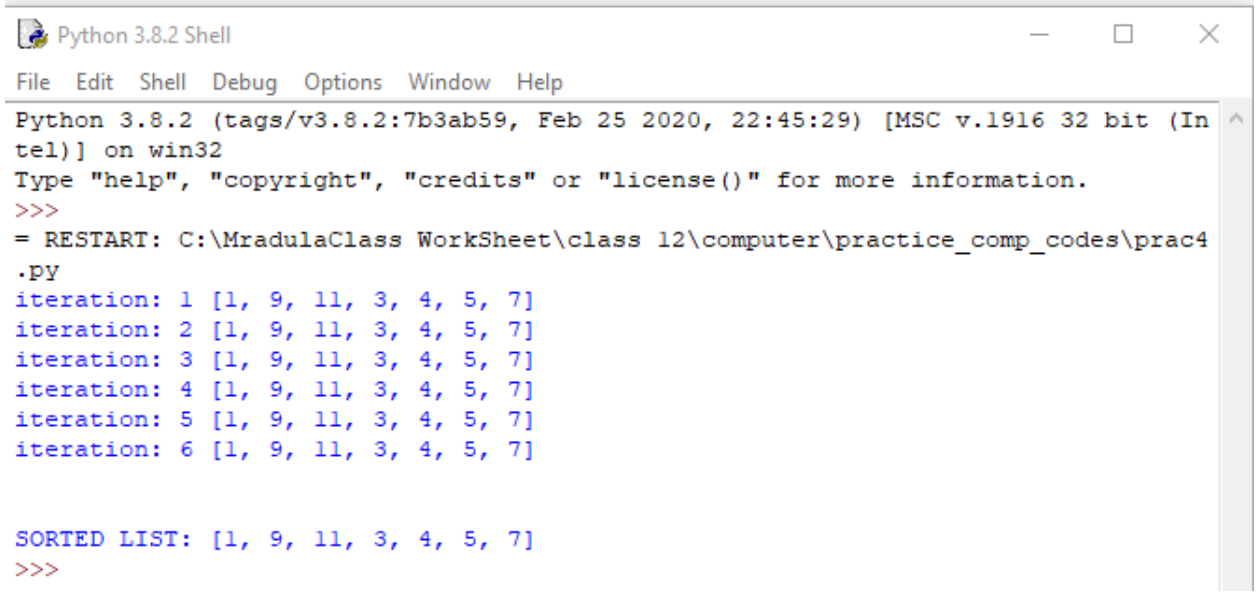
1. create a user defined function hello()

```
def hello():
    print("heloo from my function")
hello()

def heloo(s):
    print("hello from "+s)
print("hello world")
heloo("damini")
a="apeejay"
heloo(a)
```



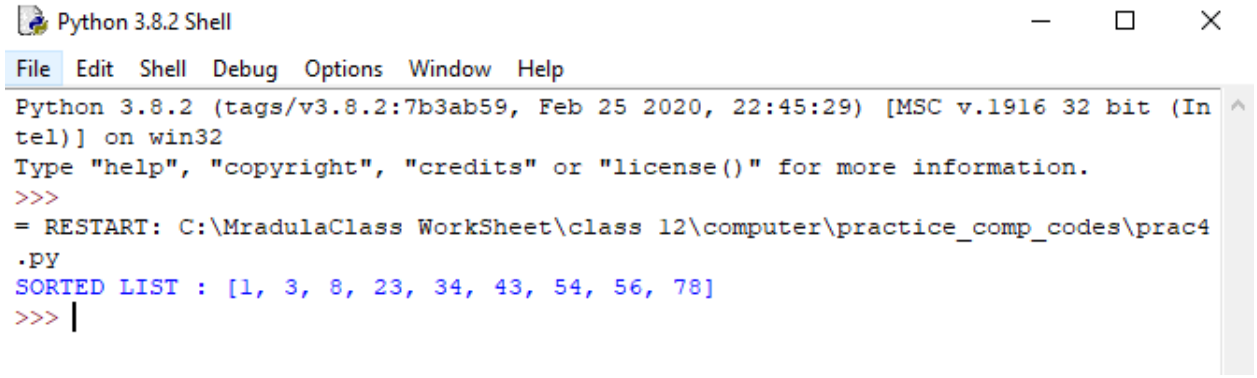
2. bubble sort

```
def bsort(a):
    for j in range(len(a)-1):
        swap=false
        for i in range(len(a)-j-1):
            if a[i]>a[i+1]:
                a[i],a[i+1]=a[i+1],a[i]
                swap=true
            print("iteration:",j+1,a)
            if swap==false:
                break;
z=[1,9,11,3,4,5,7]
bsort(z)
print("\n\nsorted list:",z)
```



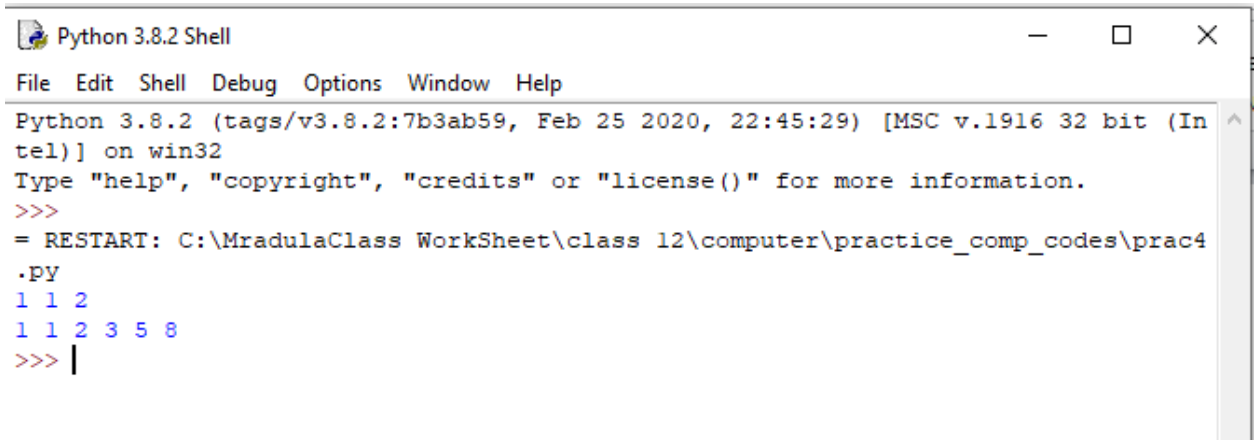
3. insertion sort

```
def isort(a):
    for i in a:
        j=a.index(i)
        while j>0:
            if a[j-1]>a[j]:
                a[j-1],a[j]=a[j],a[j-1]
            else:
                break
            j=j-1
z=[1,3,8,54,34,23,78,43,56]
isort(z)
print("sorted list :",z)
```



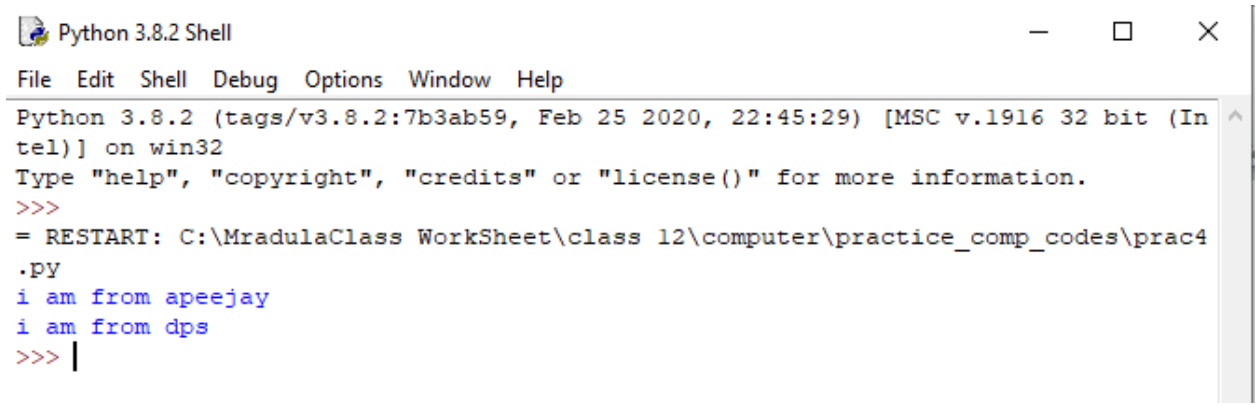
4. fibonacci series

```
def fib(n):
    a,b=0,1
    while b<n:
        print(b,end=' ')
        a,b=b,a+b
    print()

fib(3)
fib(9)
```



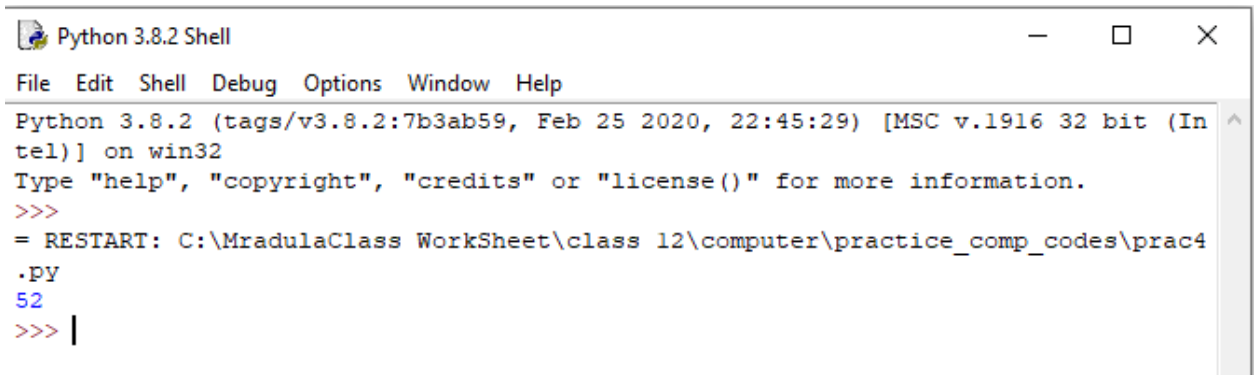
5. default argument

```
def fun(a='apeejay'):  #default argument
    print('i am from '+a)

fun()
fun("dps")
```



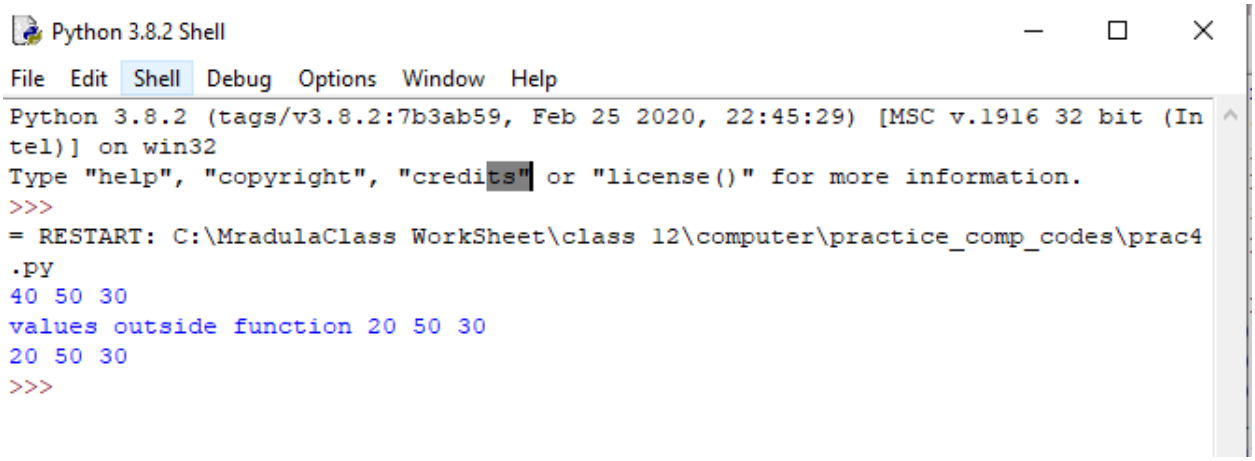
6. functions as building blocks

```
def mpow(b,p):
    y=b**p
    return y
def squ(x):
    a=mpow(x,2)
    return a
n=5
result=squ(n)+mpow(3,3)
print(result)
```



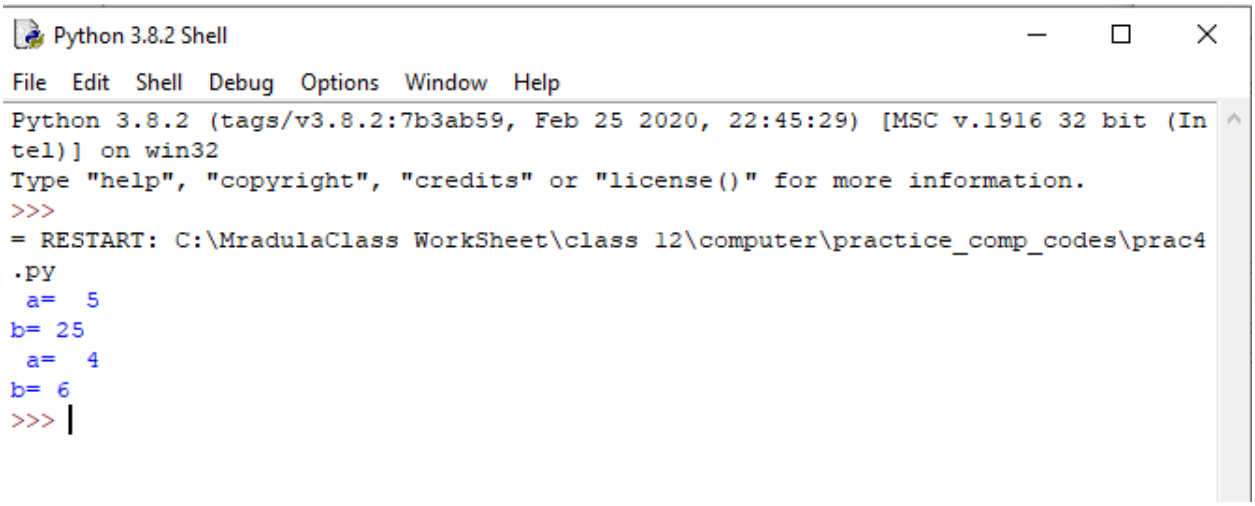
7. scope and lifetime variables

```
def modify():
    global b;
    x=10
    b=b+10
    x=x+c
    print(x,b,c)
def read():
    print(x,b,c)
x=20
b=40
c=30
modify()
print("values outside function",x,b,c)
read()
```



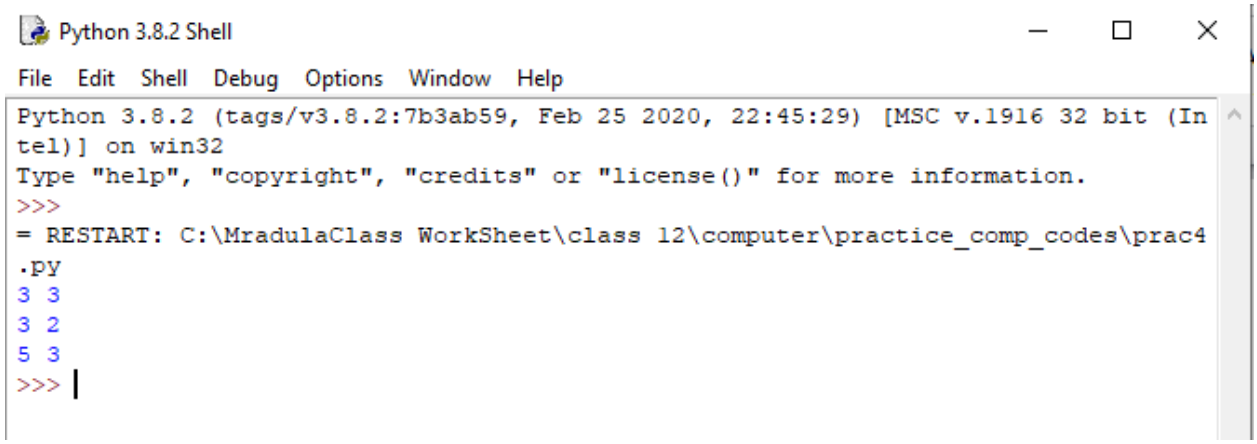
8. predict the output:

```
def calc(num):
    if num >= 5 :
        num = num * num
        return num
    else:
        num= num +2
        return num
def main(a):
    b= calc( a)
    print(" a= ", a)
    print("b=",b)
main(5)
main(4)
```



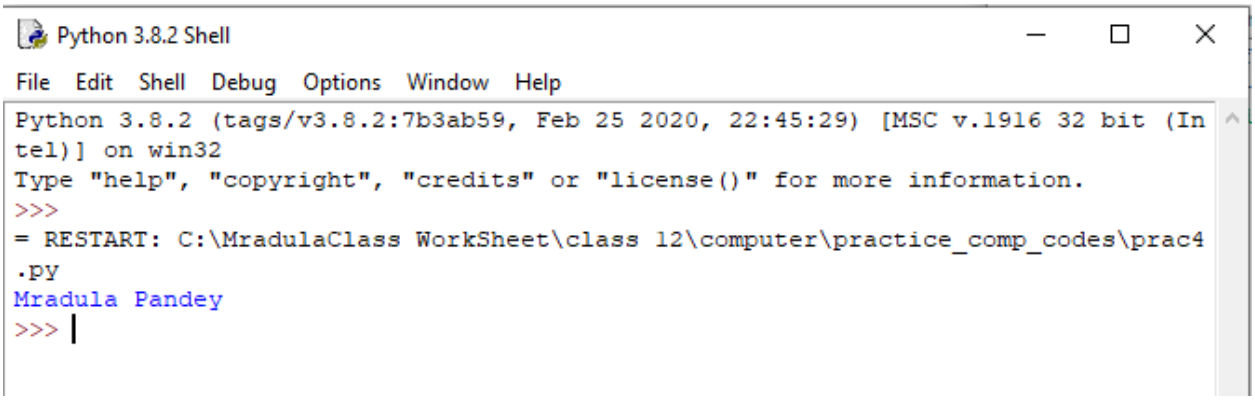
9. predict the output:

```
def magic(n1=1, n2=2):
    n1=n1+n2
    n2+=1
    print(n1,n2)
magic()
magic(2,1)
magic(3)
```



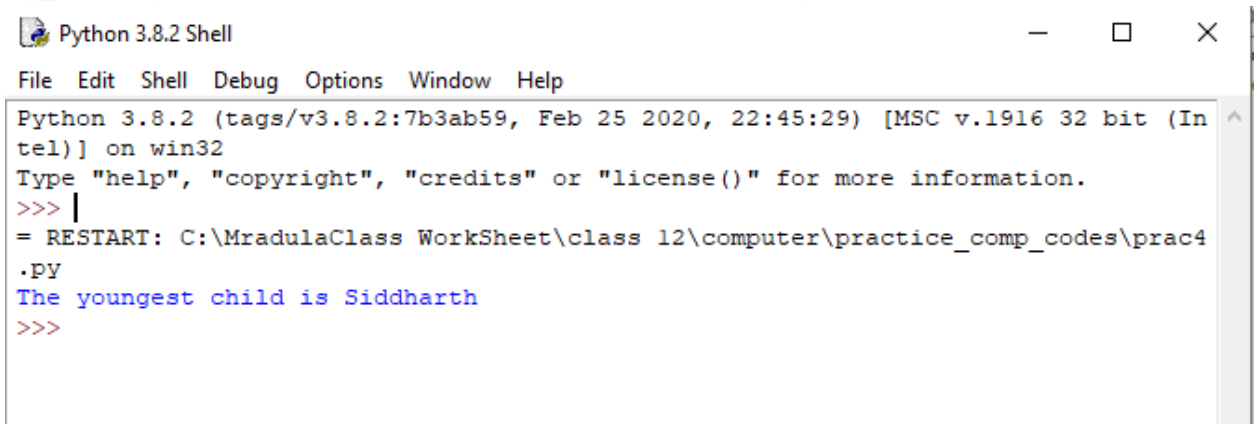
10. positional argument

```
def my_function(fname, lname):
    print(fname + " " + lname)
my_function("mradula", "pandey")
```



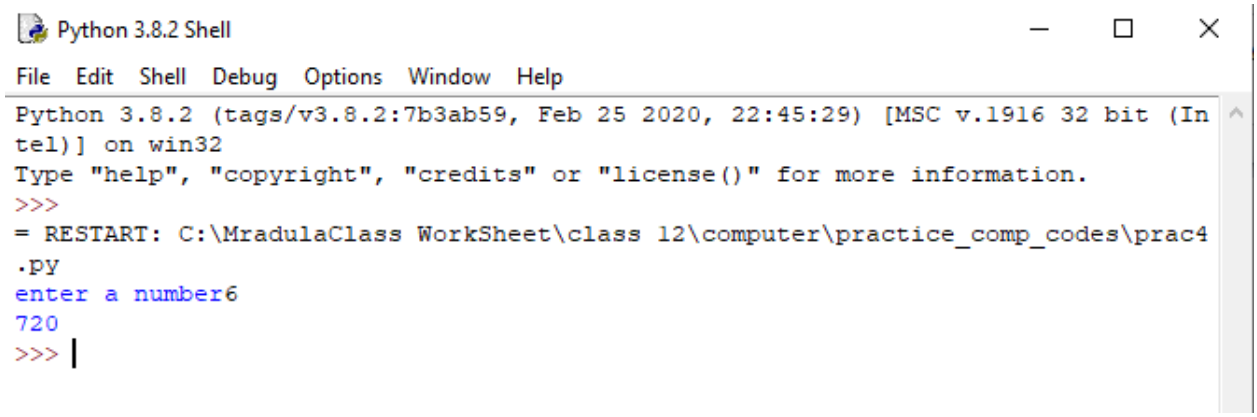
11. keyword argument

```
def my_function(child3, child2, child1):
    print("the youngest child is " + child3)
my_function(child1 = "aman", child2 = "anirudh", child3 ="siddharth")
```



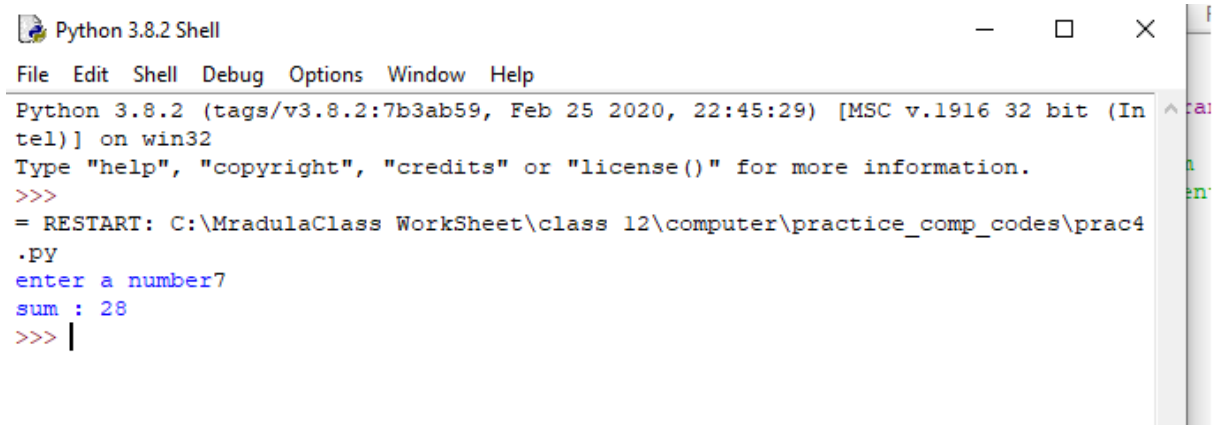
12. to display the factorial of a number

```
def fact(n):
    t=1
    for i in range(1,n+1):
        t*=i
    print(t)
y=int(input("enter a number"))
fact(y)
```



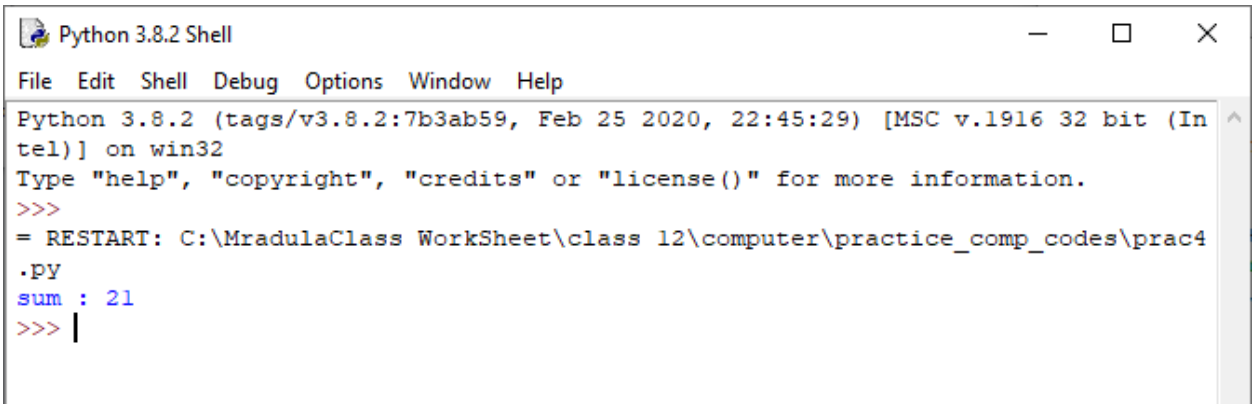
13. to display sum of number between 1 to n.

```
def sumn(n):
    s=0
    for i in range(1,n+1):
        s+=i
    print("sum :",s)
w=int(input("enter a number"))
sumn(w)
```



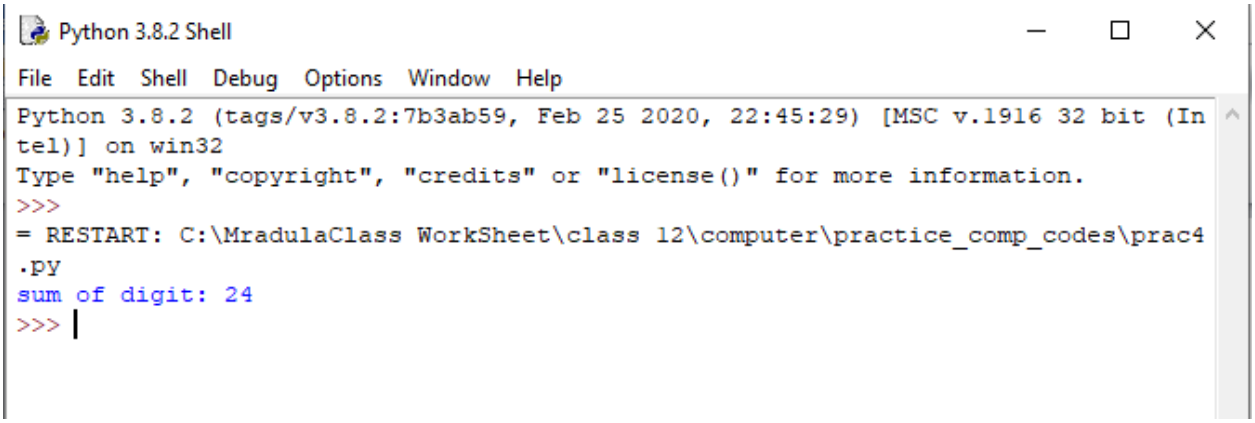
14. to display the sum of elements in a list.

```
def suml(n):
    s=0
    if n==[]:
        s=0
    else:
        for i in n:
            s+=i
    print("sum :",s)
suml([1,2,3,4,5,6])
```



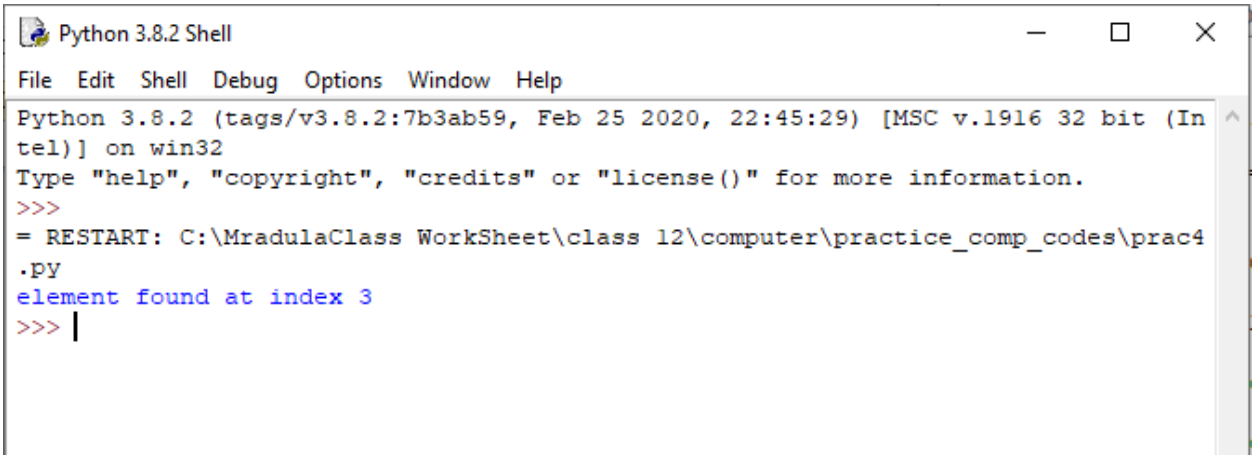
15. to display the sum of digits.

```
def sumd(n):
    s=0
    for i in str(n):
        s+=int(i)
    print("sum of digit:",s)
sumd(123387)
```



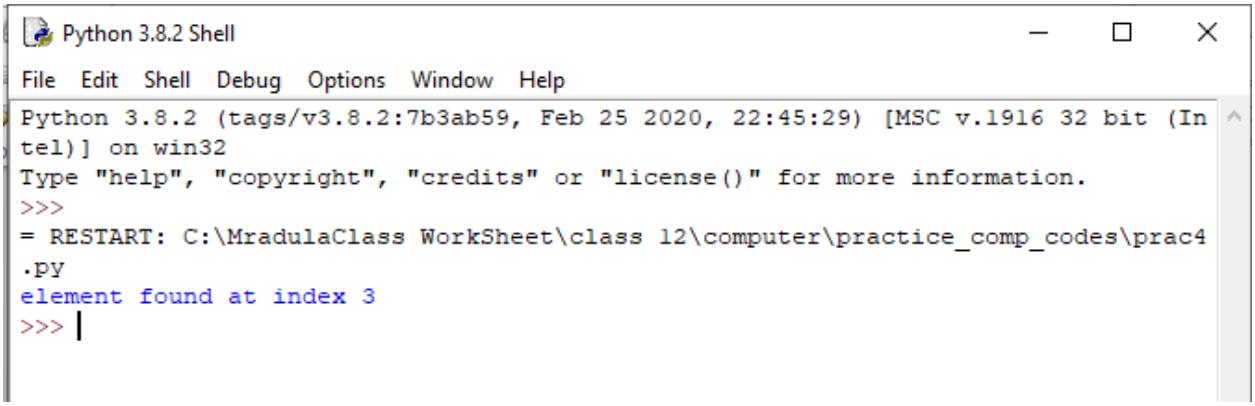
16. linear search

```
def lsearch(plist,e):
    position=-1
    for i in range(0,len(plist)):
        if (plist[i] == e):
            position=i
            break;
    return position
mylist=[1,2,3,4,10,40,60,70,100]
element=4
pos=lsearch(mylist,element)
if pos != -1:
    print("element found at index %d"%pos)
else:
    print("element not found in list collection")
```



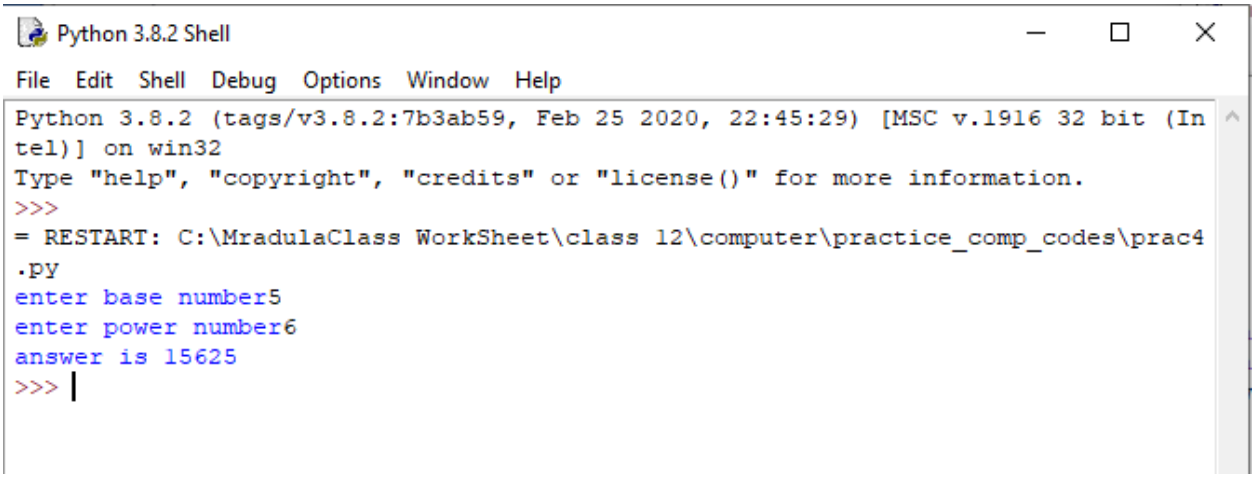
17. binary search

```
def bsearch(plist,x):
    low=found=0
    high=len(plist)
    pos=-1
    while low<=high:
        mid=(low+high)//2
        if plist[mid]==x:
            found=1
            pos=mid
            break
        elif plist[mid]<x:
            low=mid+1
        else:
            high=mid-1
    return(pos)
mylist=[1,2,3,4,10,40,60,70,100]
element=4
pos=bsearch(mylist,element)
if pos != -1:
    print("element found at index %d"%pos)
else:
    print("element not found in the list")
```



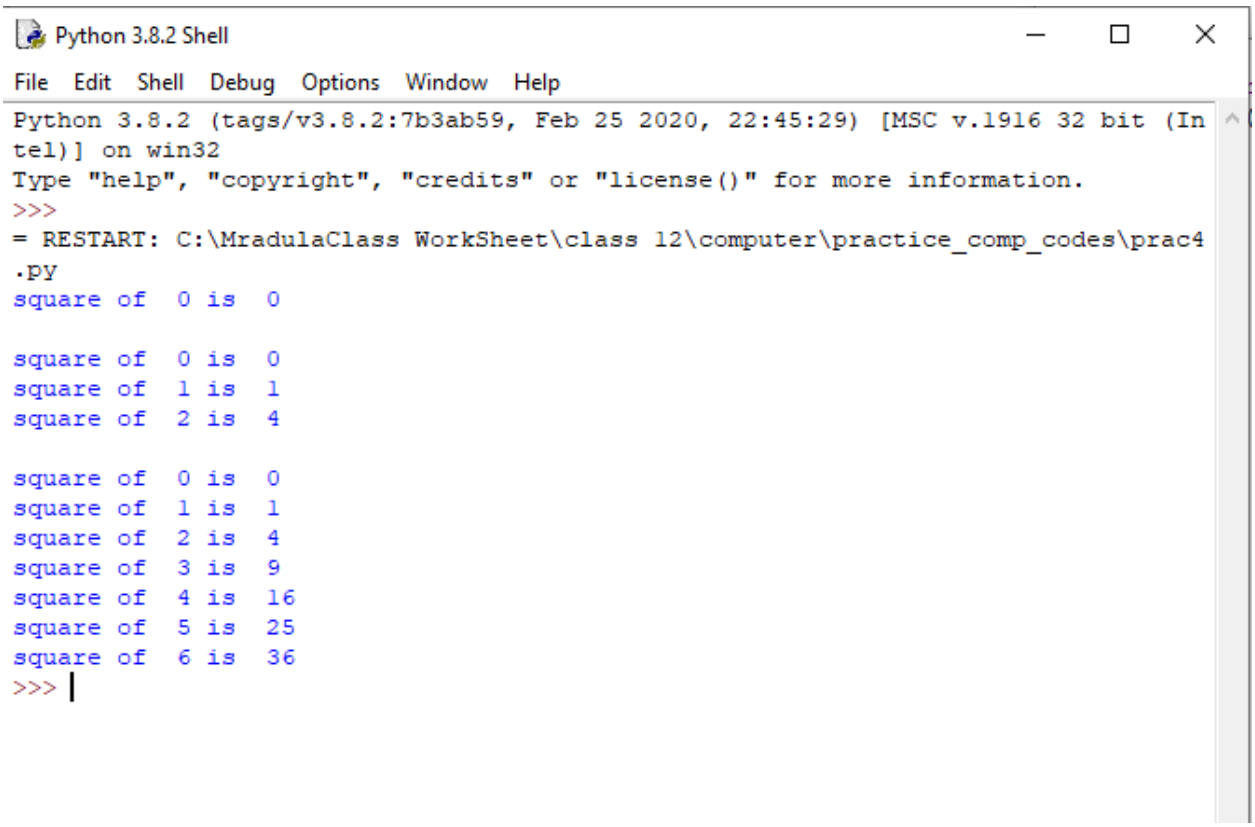
18. to display x^y .

```
def ppow(x,y):
    t=1
    if y==0:
        print("answer is 1")
    else:
        for i in range(y):
            t*=x
        print("answer is",t)
m=int(input("enter base number"))
n=int(input("enter power number"))
ppow(m,n)
```



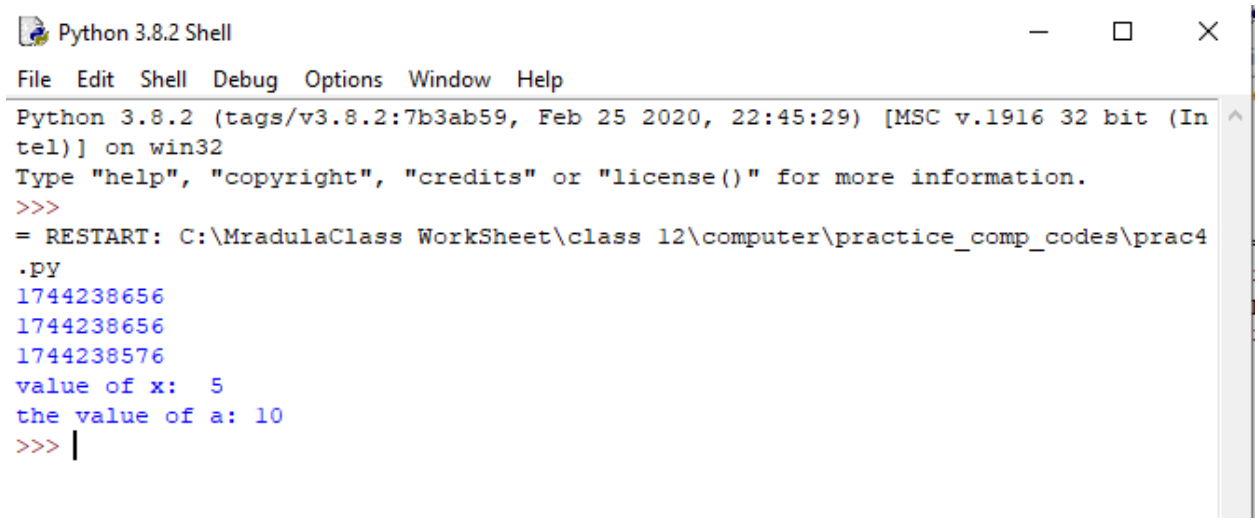
19. wap to compute and print the square of n numbers

```
def sq(n):
    for i in range(n+1):
        print("square of " ,i,"is ",i**2)
sq(0)
print()
sq(2)
print()
sq(6)
```



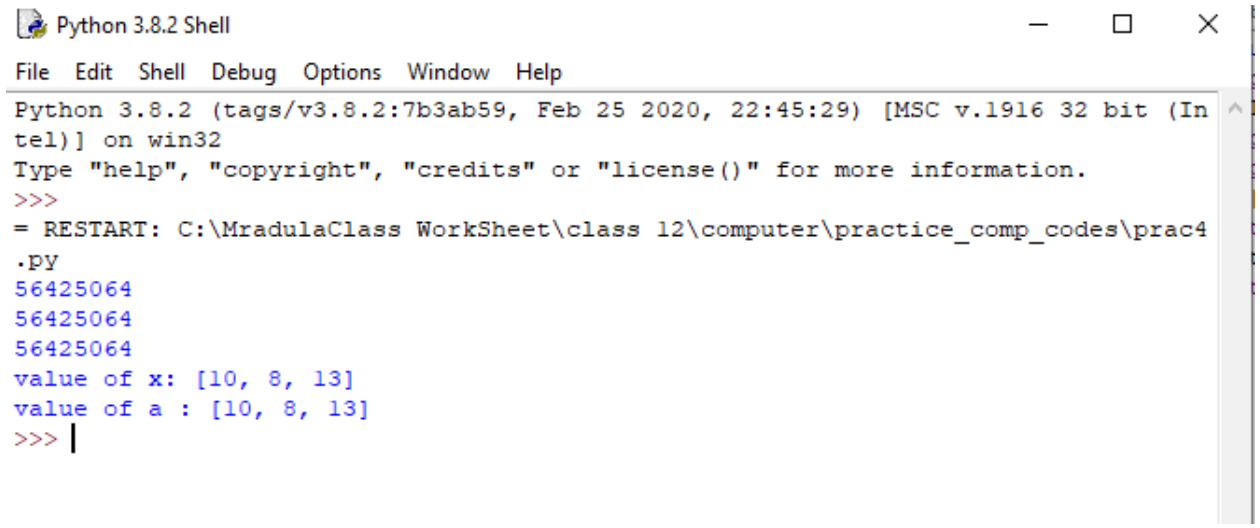
20. update method

```
def update(x):
    print(id(x))
    x=5
    print(id(x))
    print("value of x: ",x)
a=10
print(id(a))
update(a)
print("the value of a:",a)
```



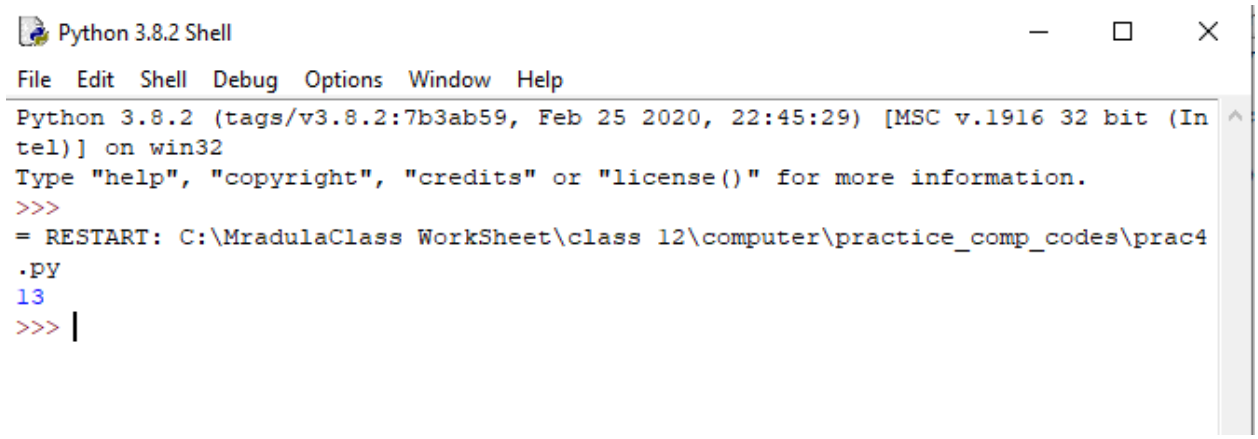
21. passing list to functions

```
def update(lst):
    print(id(lst))
    lst[1]=8
    print(id(lst))
    print("value of x:",lst)
lst=[10,12,13]
print(id(lst))
update(lst)
print("value of a :",lst)
```



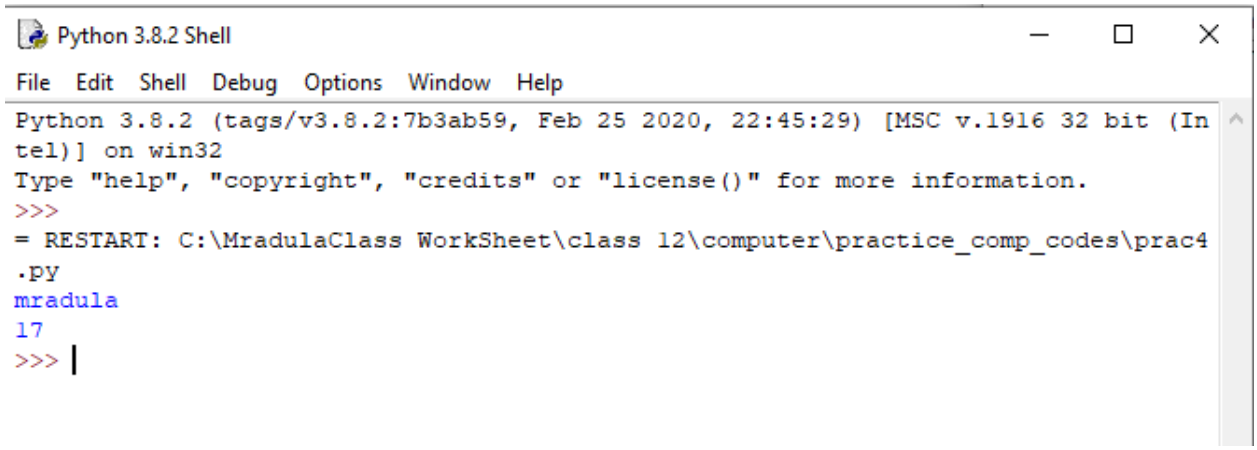
22. postitional argument

```
def add(x,y):
    c=x+y
    print(c)
    return c
add(y=4,x=9)
```



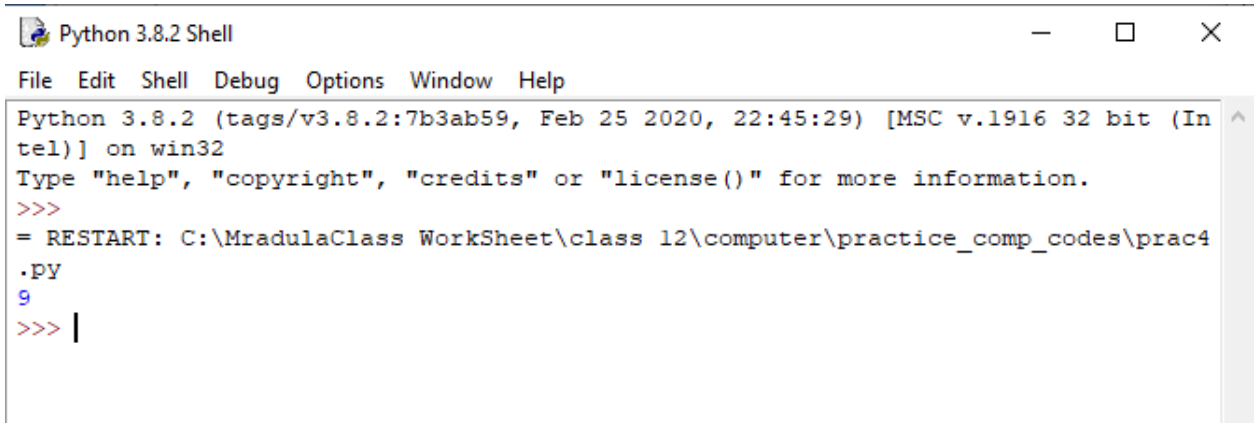
23. default argument

```
def fb(name,age=17):
    print(name)
    print(age)
fb("mradula")
```



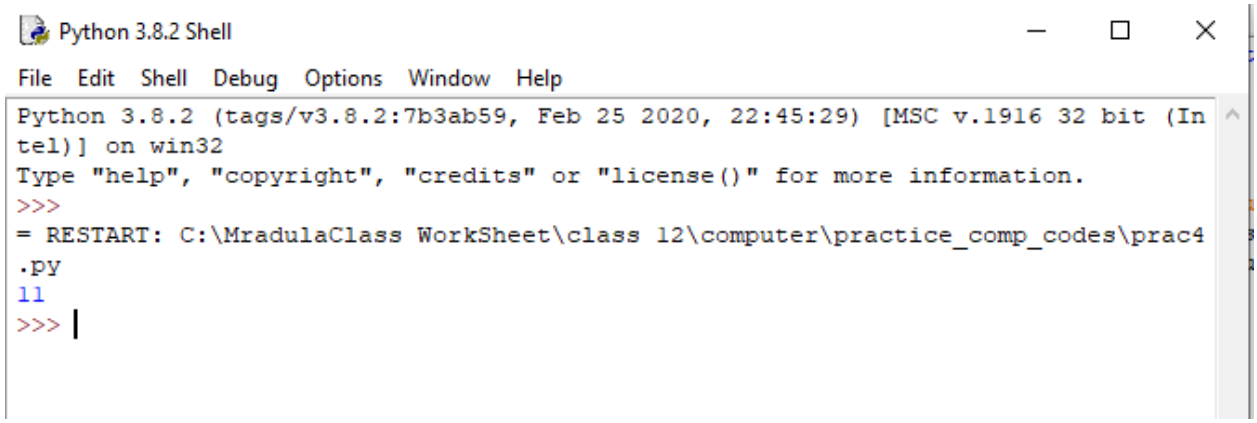
24. variable length argument

```
def add(*b):
    c=0
    for i in b:
        c+=1
    print(c)
    return c
add(2,3,4,43,34,56,65,765,7532)
```



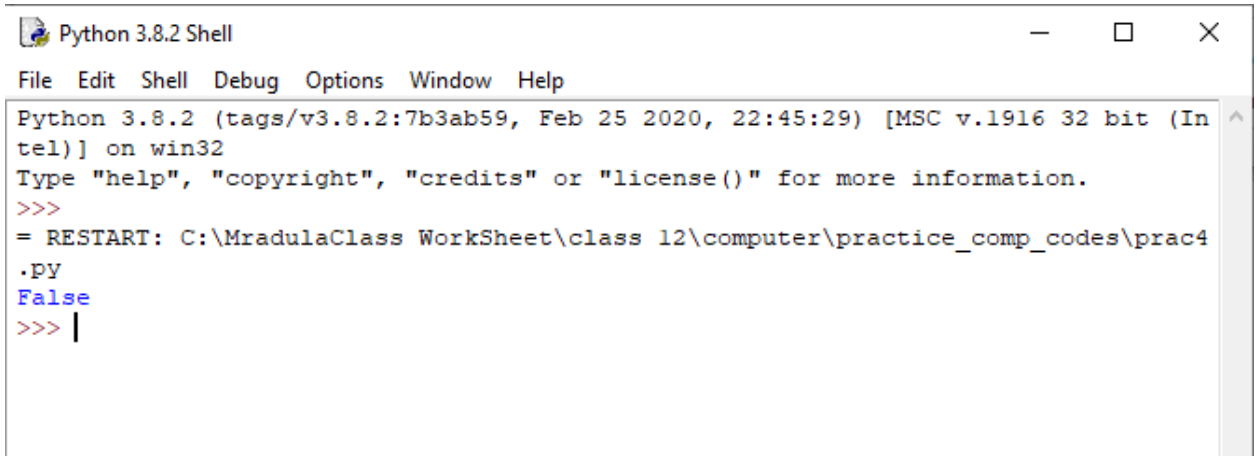
25. a python program to access a function inside a function

```
def test(a):
    def add(b):
        nonlocal a
        a+=1
        return a+b
    return add
func=test(5)
print(func(5))
```



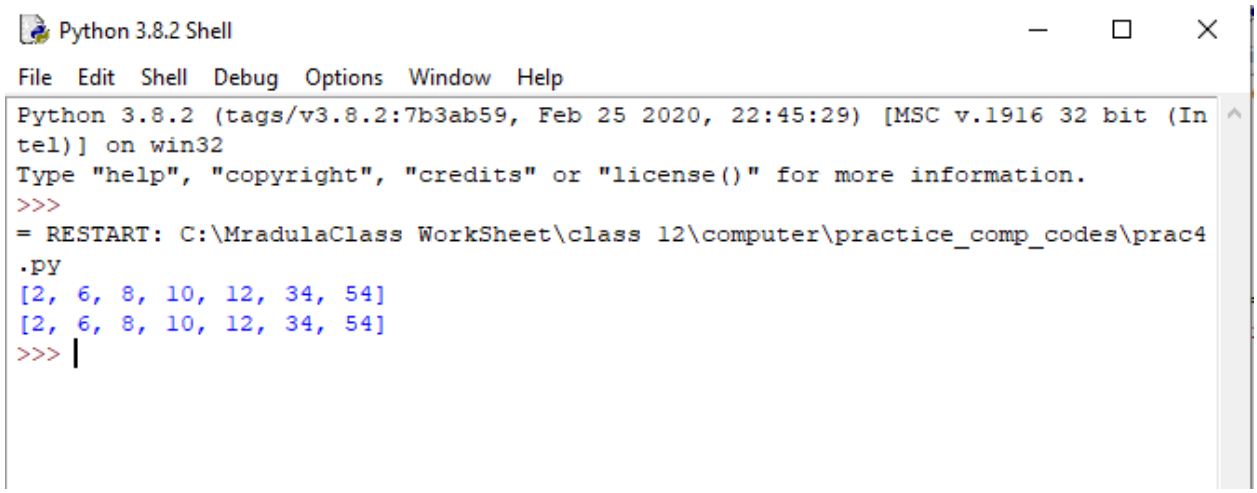
26. to check whether a number is perfect or not

```
def perfect(n):
    sum=0
    for x in range(1,n):
        if n%x==0:
            sum+=x
        return sum==n
print(perfect(6))
```



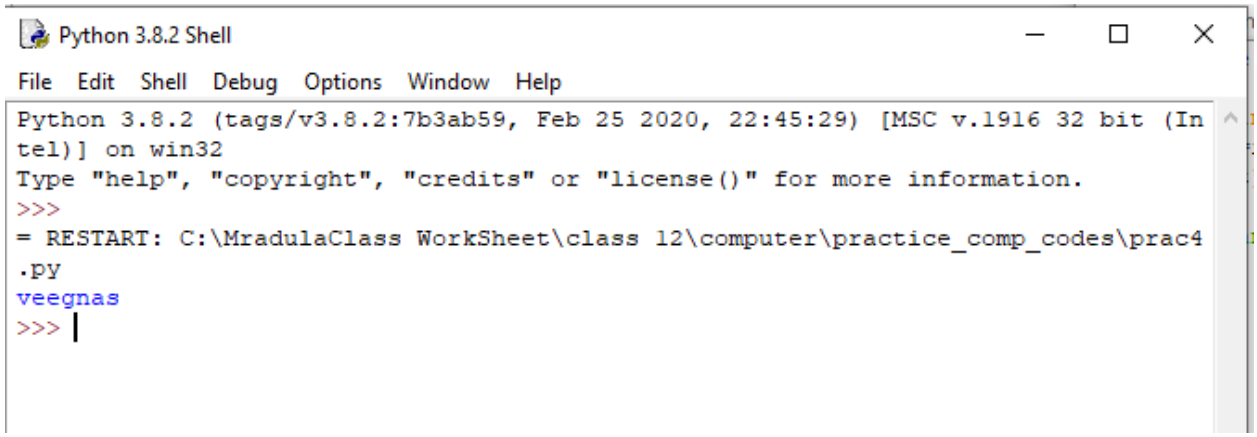
27. a python program to print the even numbers from a given list.

```
def even(n):
    enum=[]
    for i in n:
        if i%2==0:
            enum.append(i)
    print(enum)
    return enum
l=[1,2,5,6,8,10,12,34,54,55,77]
print(even(l))
```



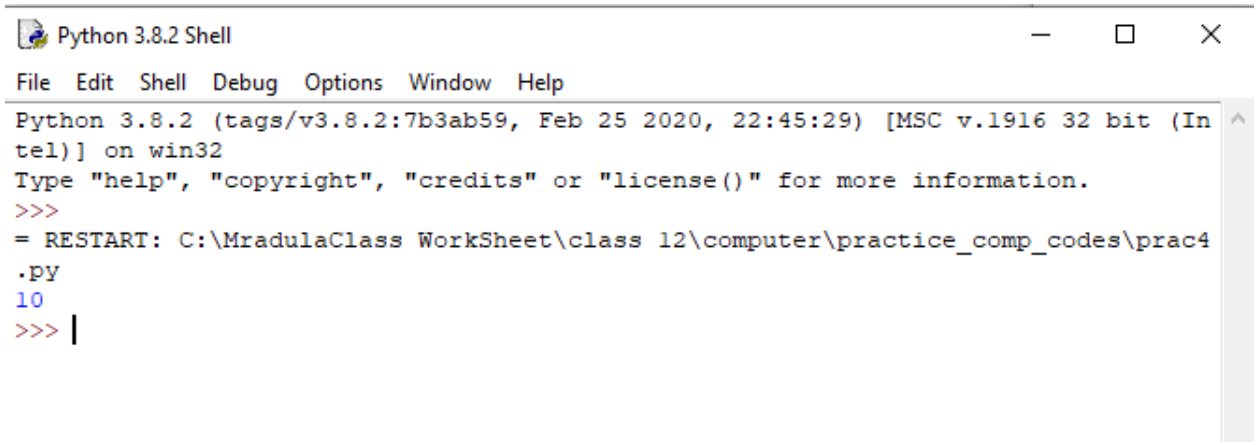
28. a python program to reverse a string

```
def reverse(n):
    t=''
    for i in n[::-1]:
        t+=i
    print(t)

reverse("sanjeev")
```



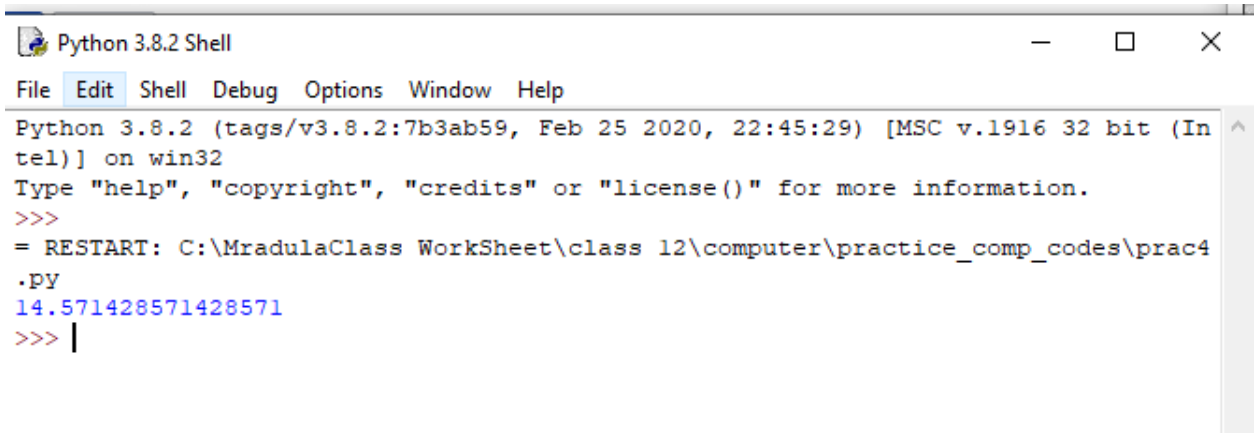
29. a python function to find the maximum of three.

```
def max1(x,y):
    if x>y:
        return x
    return y
def max2(x,y,z):
    return max1(x,max1(y,z))
print(max2(9,10,-15))
```



30. to calculate the arithmatic mean of list elements.

```
def aver(lst):
    l=len(lst)
    sum=0
    for i in lst:
        sum+=i
    return sum/l
avrg=[1,4,6,4,3,76,8]
print(aver(avrg))
```



31. program to find the roots of a quadratic equation.

```
def qdf(a,b,c):
    d=(b**2-4*a*c)**0.5
    return (-b-d)/2*a , (-b+d)/2*a
print(qdf(1,-5,6))
```