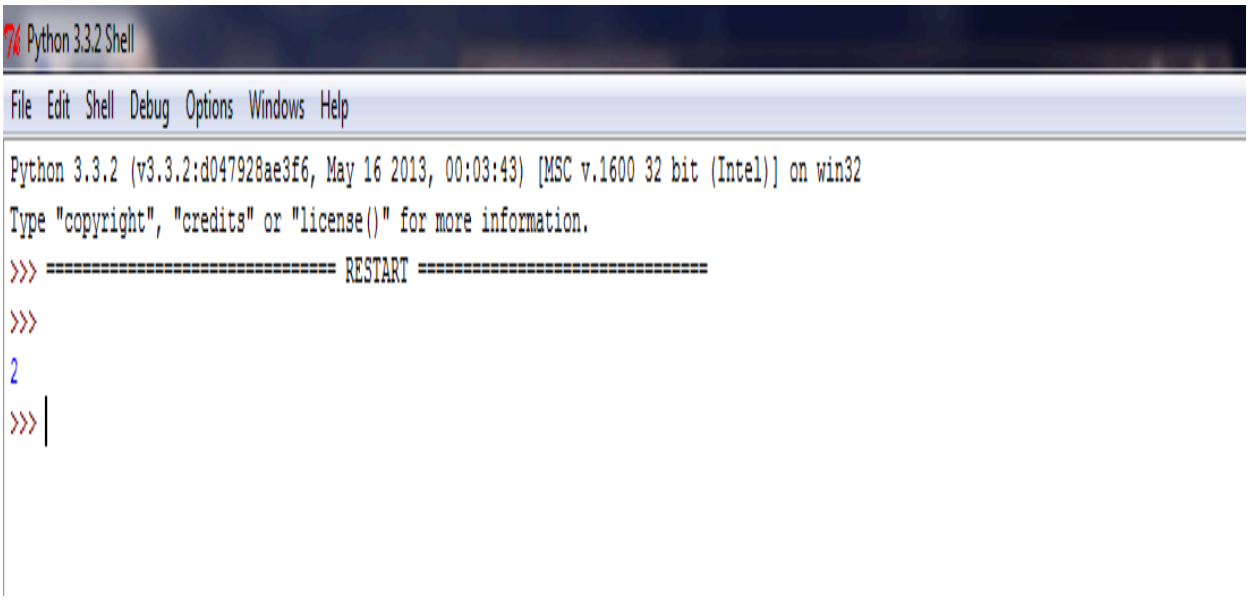
1. Write a program to find the largest from a list of numbers

```
def max_num_in_list( list ):
    max = list[ 0 ]
    for a in list:
        if a > max:
            max = a
    return max
print(max_num_in_list([1, 2, -8, 0]))
```

Output



2. Write a program to generate first n perfect numbers

```
def print_perfect_nums(start, end):
  for i in range(start, end + 1):
      sum1 = 0
      for x in range(1, i):
         if(i % x == 0):
            sum1 = sum1 + x
      if (sum1 == i):
         print(i)
print_perfect_nums(1, 300)
```

<u>Output</u>

```
>>> ================================ RESTART ================================
>>>
6
28
>>>
```

3. Write a program to perform the binary search

```
def binarySearchAppr (arr, start, end, x):
   if end >= start:
      mid = start + (end- start)//2
      if arr[mid] == x:
            return mid
      elif arr[mid] > x:
            return binarySearchAppr(arr, start, mid-1, x)
      else:
            return binarySearchAppr(arr, mid+1, end, x)
   else:
      return -1
arr = sorted(['g','u','r','u','d','e','v'])
x ='r'
result = binarySearchAppr(arr, 0, len(arr)-1, x)
if result != -1:
   print ("Element is present at index "+str(result))
else:
   print ("Element is not present in array")
```

Output

```
>>> ================================ RESTART ================================
>>>
Element is present at index 3
>>>
```

4. **Write a program to find the square root of a number using bisection search method.**

```
y = float(input('Enter the number that you want to find the square root of:
'))
num = y
x = 0
ans = 0

while abs(ans**2 - abs(num)) > 0.0001 and ans <= y:
    ans = (x + y) / 2.0
    if ans**2 < num:
        x = ans
    else:
        y = ans

print ('The square root of', num, 'is', ans)
```

Output

```
>>> ================================ RESTART ================================
>>>
Enter the number that you want to find the square root of: 4
The square root of 4.0 is 2.0
>>>
```

5. Write a program to generate Fibonacci series using recursion

```
def fibonacci(n):
    if(n <= 1):
        return n
    else:
        return(fibonacci(n-1) + fibonacci(n-2))
n = int(input("Enter number of terms:"))
print("Fibonacci sequence:")
for i in range(n):
    print(fibonacci(i))
```

Output

```
>>> ================================ RESTART ================================
>>>
Enter number of terms:10
Fibonacci sequence:
0
1
1
2
3
5
8
13
21
34
>>>
```

6. Write a program to find the LCM and HCF of 2 numbers

```
print("Enter Two Numbers: ")
numOne = int(input())
numTwo = int(input())

if numOne>numTwo:
    lcm = numOne
    hcf = numOne
else:
    lcm = numTwo
    hcf = numTwo

while True:
```

```
    if lcm%numOne==0 and lcm%numTwo==0:
        break
    else:
        lcm = lcm + 1
print("\nLCM =", lcm)

while True:
    if numOne %hcf==0 and numTwo %hcf==0:
        break
    else:
        hcf = hcf - 1
print("\nHCF =", hcf)
```

Output

```
>>> ================================ RESTART ================================
>>>
Enter Two Numbers:
8
10

LCM = 40

HCF = 2
>>> |
```

7. Write a program to perform merge sort

```
def merge_sort(list1, left_index, right_index):
    if left_index >= right_index:
        return

    middle = (left_index + right_index)//2
    merge_sort(list1, left_index, middle)
    merge_sort(list1, middle + 1, right_index)
    merge(list1, left_index, right_index, middle)


def merge(list1, left_index, right_index, middle):
    left_sublist = list1[left_index:middle + 1]
    right_sublist = list1[middle+1:right_index+1]
    left_sublist_index = 0
    right_sublist_index = 0
    sorted_index = left_index

    while left_sublist_index < len(left_sublist) and right_sublist_index <
len(right_sublist):
```

```
        if left_sublist[left_sublist_index] <= right_sublist[right_sublist_index]:
            list1[sorted_index] = left_sublist[left_sublist_index]
            left_sublist_index = left_sublist_index + 1
        else:
            list1[sorted_index] = right_sublist[right_sublist_index]
            right_sublist_index = right_sublist_index + 1

        sorted_index = sorted_index + 1
    while left_sublist_index < len(left_sublist):
        list1[sorted_index] = left_sublist[left_sublist_index]
        left_sublist_index = left_sublist_index + 1
        sorted_index = sorted_index + 1

    while right_sublist_index < len(right_sublist):
        list1[sorted_index] = right_sublist[right_sublist_index]
        right_sublist_index = right_sublist_index + 1
        sorted_index = sorted_index + 1
list1 = [44, 65, 2, 3, 58, 14, 57, 23, 10, 1, 7, 74, 48]
merge_sort(list1, 0, len(list1) -1)
print(list1)
```

Output

```
>>> ================================ RESTART ================================
>>>
[1, 2, 3, 7, 10, 14, 23, 44, 48, 57, 58, 65, 74]
>>>
```

8. **Write a program which reads the contents of a file and copy the contents to another file after changing all the letter to upper case. Exceptions should be handled.**

```
try:
      f1 = open("samplefile1.txt", "r")
except IOError:
      print("Error: can\'t find file or read data")

for x in f1.read():
      y = x.upper()
      f2 = open("samplefile2.txt", "a")
      f2.write(y)
f2 = open("samplefile2.txt", "r")
for x in f2.read():
      print(x,end="")
```

Output

```
>>> ================================ RESTART ================================
>>>
GURUDEV COLLEGE MATHIL
PAYYANUR
>>>
```

9. Write a program to find the prime numbers in a list of numbers.

```
numberList = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
ansList = []
for num in numberList :
    if num == 0 or num == 1 :
       continue
    for i in range(2, num // 2 + 1) :
        if num % i == 0 :
           break
    else :
      ansList.append(num)
if len(ansList) :
   print("Prime Number : ")
   for ans in ansList :
       print(ans)
else :
   print("No number in the given list is Prime")
```

Output

```
>>> ================================ RESTART ==========
>>>
Prime Number :
2
3
5
7
>>>
```

10. Write a python program to perform the following

a) Create table students with fields name, sex, rollno, marks
b) Insert some rows into the table
c) Update the marks of all students by adding 2 marks
d) Delete a student with a given rollno
e) Display the details of a student with a given rollno

```
import mysql.connector

mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
  database="mydatabase"
)

mycursor = mydb.cursor()

mycursor.execute("create table students(rollno int, name varchar(30),sex varchar(1),
marks int)")
mycursor = mydb.cursor()
```

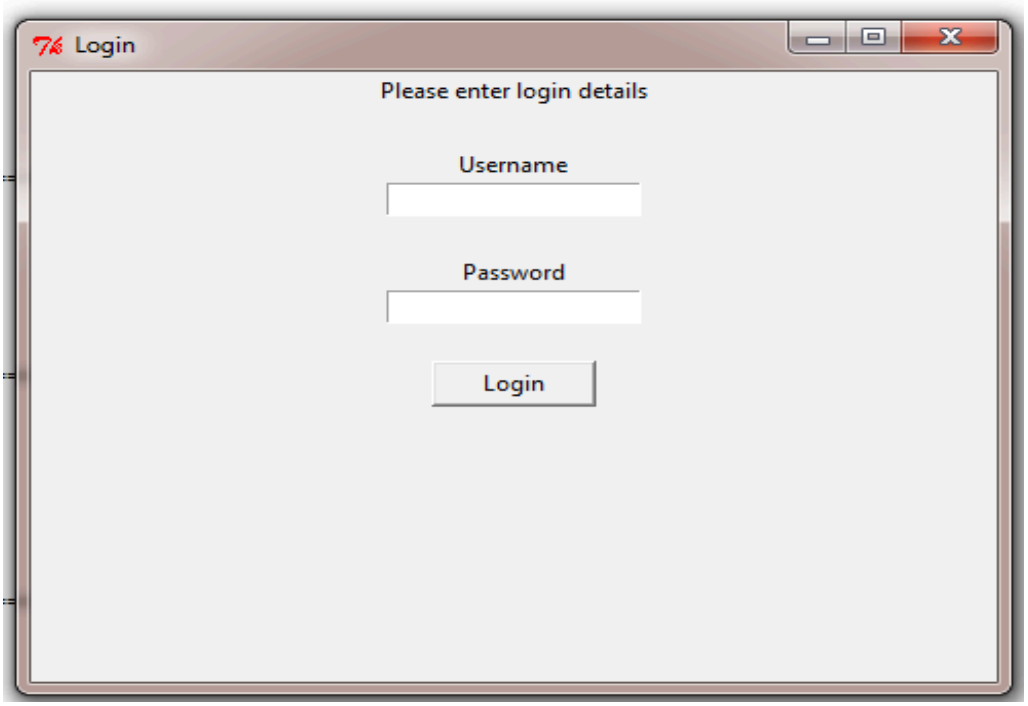
```
sql = "insert into students values (%d, %s, %s, %d)"
val = [
  (1,'peter', 'm',49),
  (1,'diya', 'f',40),
  (1,'manu', 'm',39),
  (1,'anjali', 'f',45),
  (1,'manas', 'm',48)
]

mycursor.executemany(sql, val)
mydb.commit()

print(mycursor.rowcount, "was inserted.")

mycursor.execute("select * from students")

myresult = mycursor.fetchall()

for x in myresult:
  print(x)

sql = “update customers set marks = marks + 2”
mycursor.execute(sql)

mydb.commit()
mycursor.execute("select * from students")

myresult = mycursor.fetchall()

for x in myresult:
  print(x)

rno = input('enter the roll number of student to delete :')
sql = "delete from students where rollno = %d"

mycursor.execute(sql,rno)

mydb.commit()

print(mycursor.rowcount, "record(s) deleted")
rno = input('enter the roll number of student to display the details :')

sql = "select * from customers where rollno = %d"

mycursor.execute(sql, rno)
mydb.commit()

myresult = mycursor.fetchall()
```

```
for x in myresult:
  print(x)
```

11. Create a simple Login window using Tkinter

```
from tkinter import *
def LoginPage():
    login_screen=Tk()
    login_screen.title("Login")
    login_screen.geometry("300x250")
    Label(login_screen, text="Please enter login details").pack()
    Label(login_screen, text="").pack()
    Label(login_screen, text="Username").pack()
    username_login_entry = Entry(login_screen, textvariable="username")
    username_login_entry.pack()
    Label(login_screen, text="").pack()
    Label(login_screen, text="Password").pack()
    password__login_entry = Entry(login_screen, textvariable="password", show= '*')
    password__login_entry.pack()
    Label(login_screen, text="").pack()
    Button(login_screen, text="Login", width=10, height=1).pack()
    login_screen.mainloop()
LoginPage()
```

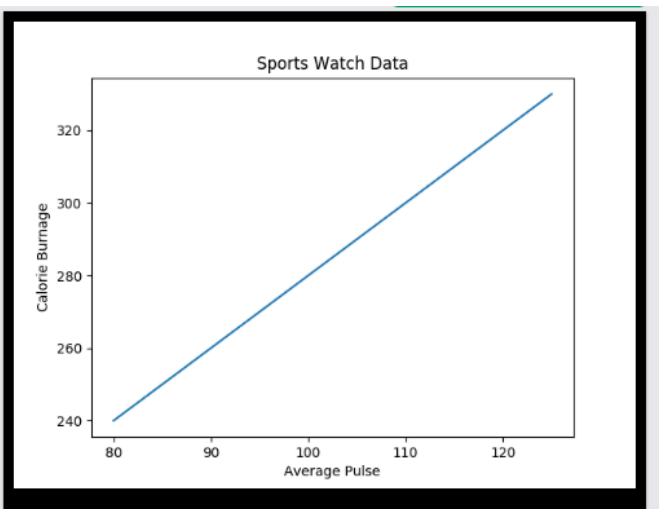
Output



12. Create a plot in Python. The title of the plot and the axes should be labelled.

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])
plt.plot(x, y)
plt.title("Sports Watch Data")
plt.xlabel("Average Pulse")
plt.ylabel("Calorie Burnage")
plt.show()
```

Output


Sports Watch Data
Calorie Burnage
320
300
280
260
240
80
90
100
110
120
Average Pulse