SRI SHANMUGHA COLLEGE OF ENGINEERING AND TECHNOLOGY

(APPROVED BY AICTE, NEW DELHI & AFFILIATED TO ANNAUNIVERSITYAND

ACCREDITED BY NAAC & NBA(ECE,CSE,MECH)

Tiruchengode-Sankari main road, Pullipalayam, Morur (Po),

Sankari (Tk), Salem (Dt) Pin: 637 304

RECORD NOTE BOOK

GE8151-PROBLEM SOLVING AND PYTHON PROGRAMMINGLABORATORY

NAME :

REG NO :

Submitted for the Anna University Practical Examinations

SRI SHANMUGHA COLLEGE OF ENGINEERING

AND TECHNOLOGY

(APPROVED BY AICTE, NEW DELHI & AFFILIATED TO ANNAUNIVERSITYAND

ACCREDITED BY NAAC & NBA(ECE,CSE,MECH)

Tiruchengode-Sankari main road, Pullipalayam, Morur (Po),

Sankari (Tk), Salem (Dt) Pin: 637 304

RECORD NOTE BOOK

REG NO

Certified that this is a Bonafide record of Practical work Done by

Mr/Ms……..of the……..

Semester……Branch during the Academic Year……

in the…….....Laboratory.

Staff-in-charge Head of the Department

Submitted for the University Practical

Examination held on……..

Internal Examiner External Examiner

TABLE OF CONTENT

6		SELECTION SORT			
		INSERTION SORT			
7		MERGE SORT			
8		FIRST N PRIME NUMBERS			
9		MULTIPLY MATRICES			
10		PROGRAMS THAT TAKE COMMAND LINE ARGUMENTS (WORD COUNT)			
11		FIND THE MOST FREQUENT WORDS IN A TEXT READ FROM A FILE			
12		SIMULATE ELLIPTICAL ORBITS IN PYGAME			
13		SIMULATE BOUNCING BALL USING PYGAME			

Ex.No.1 **COMPUTE GCD OF TWO NUMBERS**
Date:

Aim:
To Write a Python program to compute the GCD of two numbers.

Algorithm:
Step 1: Start
Step 2: Take two numbers from the user as input
Step 3: Calculate the remainder d1%d2
Step 4: While the remainder is not equal to 0
Step 5: d1=d2
Step 6: d2=remainder
Step 7: remainder=d1 % d2
Step 8: print the GCD
Step 9: Stop

Program:
Non Recursive:

```
d1=int(input("Enter a number: "))
d2=int(input("Enter another number: "))
rem=d1%d2
while rem!=0 :
   d1=d2
   d2=rem
   rem=d1%d2
print ("GCD of given numbers is : ")
print(d2)
```

Recursice:

```
def gcd(a,b):
        if(b==0):
                return a
        else:
                return gcd(b,a%b)
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))
print (gcd(a,b))
```

Result:

Thus the Python program to compute the GCD of two numbers was created and executed successfully.

Ex No: 2 **FIND THE SQUARE ROOT OF A NUMBER**
Date: **(NEWTON'S METHOD)**

Aim:
To write a Python Program to find the square root of a number by Newton's Method.

Algorithm:
1. Define a function named newtonSqrt().
2. Initialize approx as 0.5*n and better as 0.5*(approx.+n/approx.)
3. Use a while loop with a condition better!=approx to perform the following,
i. Set approx.=better
ii. Better=0.5*(approx.+n/approx.)
4. Print the value of approx.

Program:
```
def newtonSqrt(n):
        approx = 0.5 * n
        better = 0.5 * (approx + n/approx)
        while better != approx:
                approx = better
                better = 0.5 * (approx + n/approx)
        return approx
s=int(input("Enter the number:"))
print('The square root is' ,newtonSqrt(s))
```

Result:
Thus the Python program for finding the square root of a given number by Newton's Method is executed successfully and the output is verified.

Ex.No.3 **EXPONENTIATION (POWER OF A NUMBER)**
Date :

Aim:
To Write a Python program to find the exponentiation (Power of a number) .

Algorithm:
Step 1: start
Step 2: read values
Step 3: r= a**b
Step 4: print r
Step 5: stop

PROGRAM

```
num1=int(input("enter value a:"))
num2=int(input("enter value b:"))
r=num1**num2
print("exponentiation is", r)
```

Result:

Thus the a Python program to find the exponentiation (Power of a number) was created and executed successfully.

Ex.No.4 **FIND THE MAXIMUM OF A LIST OF NUMBERS**
Date :

Aim:
To write a Python program to find the maximum of a list of numbers.

Algorithm:
Step 1: Start
Step 2: Take n the number of elements and store it in a variable.
Step 3: Take the elements of the list one by one.
Step 4: Print the last element of the list
Step 5: Stop

Program

```
n=int(input("Enter the list size:"))
a=[]
for i in range(n):
        num=int(input("Enter the number"))
        a.append(num)
print (a)
max=a[0]
for i in range(n):
        if(max<a[i]):
                max=a[i]
print ("maximum",max)
```

Result:
Thus a Python program to find the maximum of a list of numbers was created and executed successfully.

Ex.No.5a **LINEAR SEARCH**
Date :

Aim:
To write a python program to search an element in an array using Linear search technique.

Algorithm:
Step 1: Start
Step 2: Get list of elements from the user
Step 3: Get the number to be searched
Step 4: Set i to 1
Step 5: if i>n then go to step 7
Step 6: if A[i]=x then go to step 6
Step 7: Set I to i+1
Step 8: Go to Step 2
Step 9: Print Element x found at index I and step 8
Step 10: Print element not found
Step 11: Stop

Program

```
n=int(input("Enter the no. of element"))
i=0
a=[i for i in range(n)]
for i in range(0,n):
   a[i]=int(input("enter the array element"))
for i in range(0,n):
   print (a[i])
key=int(input("Enter the key element to be searched"))
for i in range(0,n):
   if a[i]==key:
      print ("key found")
```

Result:

Thus a python program to search an element was created using linear search technique and executed successfully.

Ex.No.5b **BINARY SEARCH**
Date:

Aim:
To write a Python program to search an element in a list of elements using Binary search technique.

Algorithm:
Step 1: Start
Step 2: Get list of elements from the user
Step 3: Get the number to be searched
Step 4: Found<-False
Step 5: While not found and first <=top
if List[Midpoint]=ItemSought Then
ItemFound=True
Elif first>=Last Then
SearchFailed=True
Elif List[Midpoint]>ItemSought Then
Last=Midpoint-1
Else
First=Midpoint+1
End if
End While
Step 7: Stop

Program:

```
def binary_search(a,n,key):
    low=0
    high=n
    while(low<=high):
        mid=int((low+high)/2)
        if(key==a[mid]):
            return mid
        elif(key<a[mid]):
            high=mid-1
        else:
            low=mid+1
    return -1
n=int(input("enter the no of element"))
a=[i for i in range(n)]
for i in range(0,n):
    a[i]=int(input("enter the array element"))
k=int(input("Enter the key element to be searched"))
position=binary_search(a,n,k)
```

Result:

Thus a python program to search an element using Binary search technique was created and executed successfully.

Ex.No.6a **SELECTION SORT**
Date:

Aim:
To write a Python program to sort the elements using selection sort.

Algorithm:
Step 1: Start
Step 2: Get the elements from the user
Step 3: Set MIN to location 0
Step 4: Search the minimum element in the list.
Step 5: Swap with value at location MIN
Step 6: Increment MIN to point to next element
Step 7: Repeat until list is sorted.
Step 8: Stop

Program:
```
def selectionSort(a):
    for i in range(len(a)):
        least=i
        for k in range(i+1,len(a)):
            if a[k]<a[least]:
                least=k
        temp=a[least]
        a[least]=a[i]
        a[i]=temp
a=[50,30,10,20,40,70,60]
print ("Original list",a)
selectionSort(a)
print("Selection Sort:",a)
```

Result:
Thus a Python programs to sort elements using selection sort method was created and executed successfully.

Ex.No.6b **INSERTION SORT**
Date:

Aim:

To write a Python program to sort the elements using insertion sort .

Algorithm:

Step 1: Start
Step 2: Get the elements from the user
Step 3a: The second element in the list is compared with the elements that appear before it (only first element in this case).
Step 3b: If the second element is smaller than first element, second element is inserted in the position of first element. After first step, first two elements of an array will be sorted.
Step 3c: Pick next element
Step 4: Repeat step 3 until all the elements in the list is sorted
Step 5: Stop

Program:

```
def insertion(a):
    for i in range(1,len(a)):
        currentvalue=a[i]
        position=i
        while position>0 and a[position-1]>currentvalue:
            a[position]=a[position-1]
            position=position-1
        a[position]=currentvalue
a=[] #define empty list
n=int(input("Enter the upper limit"))
for i in range (n):
    a.append(int(input()))
insertion(a)
print ("Insertion list",a)
```

Result:

Thus a Python program to sort the elements using selection sort method was created and executed successfully.

Ex. No. : 7 **MERGE SORT**
Date:

AIM:
To write a Python program to sort all the elements in proper order using the logic of Merge sort.

ALGORITHM:
1. Create a function named mergesort
2. Find the mid of the list
3. Assign lefthalf = alist[:mid] and righthalf = alist[mid:]
4. Initialise i=j=k=0
5. while i < len(lefthalf) and j < len(righthalf), perform the following
if lefthalf[i] < righthalf[j]:
alist[k]=lefthalf[i]
Increment i
else
alist[k]=righthalf[j]
Increment j
Increment k
6. while i < len(lefthalf),perform the following
alist[k]=lefthalf[i]
Increment i
Increment k
7. while j < len(righthalf), perform the following
alist[k]=righthalf[j]
Increment j
Increment k
8. Print the sorted list.

Program:

```
def mergeSort(alist):
    # print("Splitting ",alist)
    if len(alist)>1:
        mid = len(alist)//2
        lefthalf = alist[:mid]
        righthalf = alist[mid:]
        mergeSort(lefthalf)
        mergeSort(righthalf)
        i=j=k=0
        while i < len(lefthalf) and j < len(righthalf):
            if lefthalf[i] < righthalf[j]:
                alist[k]=lefthalf[i]
                i=i+1
            else:
                alist[k]=righthalf[j]
                j=j+1
            k=k+1
        while i < len(lefthalf):
            alist[k]=lefthalf[i]
            i=i+1
            k=k+1
        while j < len(righthalf):
            alist[k]=righthalf[j]
            j=j+1
            k=k+1
#print("Merging ",alist)
alist = [54,26,93,17,77,31,44,55,20]
mergeSort(alist)
print(alist)
```

Result:

Thus the Python program to perform merge sort is successfully executed and the output is verified.

Ex. No. : 8 **FIRST N PRIME NUMBERS**
Date:

Aim:
To write a Python program to find first n prime numbers.

Algorithm:
1. Read the value of Lower range and upper range
2. for num in range(lower,upper+ 1), perform the following
3. if num%i is 0 then break
else print the value of num
4. Repeat step 3 for i in range(2,num)

Program:

```
lower = int(input("Enter lower range: "))
upper = int(input("Enter upper range: "))
for num in range(lower,upper + 1):
  if num > 1:
      for i in range(2,num):
          if (num % i) == 0:
              break
           else:
              print(num)
```

Result:

Thus the Python Program to find the first n prime numbers is executed successfully and the output is verified.

Ex No: 9 **MULTIPLY MATRICES**
DATE:

AIM:
To write a Python program to multiply the two matrices.

ALGORITHM:
Step 1 : Start
Step 2 : Assign elements for X matrix.
Step 3 : Assign elements for Y matrix.
Step 4 : Repeat the step 5 to 7 until range in 'i' occur false.
Step 5 : Repeat the step 6 to 7 until range in 'j' occur false.
Step 6 : Repeat the step 5 to 7 until range in 'k' occur false.
Step 7 : Calculate result[i][j]=result[i][j]+X[i][k]*Y[k][j].
Step 8 : Print the resultant matrix.
Step 9 : Stop.

Program:

```
X = [[1,2,3], [4,5,6], [7,8,9]]
Y = [[1,2,3], [1,2,3], [1,2,3]]
result = [[0,0,0], [0,0,0], [0,0,0]]
for i in range(len(X)):
   for j in range(len(Y[0])):
      for k in range(len(Y)):
         result[i][j] += X[i][k] * Y[k][j]
for r in result:
   print(r)
```

RESULT:

Thus the Python program was executed and the output is verified.

Ex.No.10 **PROGRAMS THAT TAKE COMMAND LINE ARGUMENTS**
Date: **(WORD COUNT)**

Aim:
To write a Python program to take command line arguments(word count).

Algorithm:
Step 1: Start
Step 2: Take the file name from the user
Step 3: Read each line from a file and split a lines to form a list of words
Step 4: Find the length of items in a list and print it
Step 5: Stop

Program:

```
import sys
print("\n\n\t Script Name : ",sys.argv[0])
print("\n")
le=len(sys.argv)
for i in range(1,le):
   print("\t Word : ",i," : ", sys.argv[i])
print("\n\n\t Word Count : ", len(sys.argv)-1)
```

Result:

Thus a Python program for word count was created and executed successfully.

Ex.No.11 **FIND THE MOST FREQUENT WORDS IN A TEXT READ**
Date: **FROM A FILE**

Aim:
To write a Python program to find the most frequent words in a text read from a file.

Algorithm:
Step 1: Start
Step 2: Create a file in a notepad and save it with .py extension.
Step 2: Take the file name and letter to be counted from the user
Step 2: Read each line from the file and split the line to form a list of words.
Step 3: Use a for loop to traverse through the words in the list and another for loop to traverse through the letters in the word
Step 4: Check if the letter provided by the user and the letter encountered over iteration is equal and if they are, increment the letter count.
Step 5: stop

Program:

```
from collections import Counter
def frequent(name):
   with open(name) as file:
      return Counter(file.read().split())
print ("Most frequent word")
print ("File name:")
name=input()
print (frequent(name))
```

input.txt:

```
Hindusthan Institute of technology
Hindusthan college of engineering and technology
Coimbatore
```

Result:

Thus a Python program to find the most frequent words in a text read from a file was created and successfully.

Ex.No.12 SIMULATE ELLIPTICAL ORBITS IN PYGAME
Date:

Aim:
To write a Python program to simulate elliptical orbits in pygame.

Algorithm:
Step 1: Start
Step 2: Import necessary mathematical function library from numpy package
Step 3: Import plot library
Step 4: Create the necessary window and
Step 5: draw elliptical orbit by using the equation ay2 + bxy + cx + dy + e = x2
Step 6: Plot the points using Scatter plot functions and draw orbit
Step 7: Stop

Program:

```
from visual import *
from visual.graph import *
scene=display(width=600,height=600,center=(0,5,0))
Sun=sphere(pos=(0,5,0),radius=100,color=color.orange)
earth=sphere(pos=(-200,0,0),radius=10,material=materials.earth,make_trial=true)
earthv=vector(0,0,5)
gd=gdisplay(x=800,y=0,width=600,height=600,
foreground=color.black,background=color.white,
xmax=3000,xmin=0,ymax=20,ymin=0)
f1=gcurve(color=color.blue)
t=0
for i in range(1000):
rate(50)
earth.pos=earth.pos+earthv
dist=(earth.x**2 + earth.y**2 + earth.z**2)**0.5
RadialVector=(earth.pos - Sun.pos)/dist
Fgrav=-10000*RadialVector/dist **2
earthv=earthv+Fgrav
earth.pos+=earthv
f1.plot(pos=(t,mag(earthv)))
t+=1
if dist<=Sun.radius:
break
```

Result

Thus a Python program to simulate elliptical orbits in pygame was created and executed successfully.

Ex.No.13 **SIMULATE BOUNCING BALL USING PYGAME**
Date :

Aim:
To write a Python program to simulate bouncing ball using pygame.

Algorithm:
Step 1: Start
Step 2: Import necessary GUI for simulation
Step 3: Set the window coordinates and the ball coordiates
Step 4: Assign various colors for the ball and set the base color
Step 5: Fix the color, shape and bouncing speed randomly
Step 6: Write a function to move the base according to the ball position
Step 7: Stop

Program:

```
from visual import *
floor = box(length=4, height=0.25, width=4, color=color.blue)
ball = sphere(pos=(0,4,0), color=color.red)
ball.velocity = vector(0,-1,0)
dt = 0.01
while 1:
rate(100)
ball.pos = ball.pos + ball.velocity*dt
if ball.y < 1:
ball.velocity.y = -ball.velocity.y
else:
ball.velocity.y = ball.velocity.y - 9.8*dt
```

Result:

Thus a Python program to simulate bouncing ball using pygame was simulated successfully