

# **MORADABAD INSTITUTE OF TECHNOLOGY**

## **MORADABAD**

Department Of Computer Science and Engineering

**COURSE: B. TECH**

**YEAR: 2<sup>ND</sup>**

**SEMESTER: 4<sup>TH</sup>**



### **PYTHON LANGUAGE PROGRAMMING LAB FILE**

**( KCS 453 )**

**SUBMITTED TO:**

**MR. VIBHOR KUMAR VISHNOI**

**SUBMITTED BY:**

**YASH BHATNAGAR**

**ROLL NO.: 2000820100156**

**SECTION: C**

# INDEX

S.NO.	PROGRAM	DATE	REMARK	SIGN
1.	Write a python program that takes in command line arguments as input and print the number of arguments.			
2.	Write a python program to perform matrix multiplication.			
3.	Write a python program to compute the GCD of two numbers.			
4.	Write a python program to find the most frequent words in a text file.			
5.	Write a python program to find the square root of a number (Newton's method)			
6.	Write a python program exponentiation (power of a number).			
7.	Write a python program to find the maximum of a list of numbers			
8.	Write a python program for linear search.			
9.	Write a python program for Binary search.			
10.	Write a python program for Selection sort.			
11.	Write a python program for insertion sort.			
12.	Write a python program for merge sort.			
13.	Write a python program for first n prime numbers.			
14.	Write a python program to simulate bouncing ball in pygame.			
15.	Write a python program to check a number is palindrome.			
16.				
17.				

## **PROGRAM - 01**

**OBJECTIVE:-** Write a python program that takes in command line arguments as input and print the number of arguments.

### **SOURCE CODE:-**

```
import sys

print(sys.argv)

n=len(sys.argv)

print("Length of Argument is:",n)
```

### **OUTPUT:-**

#### **USING IDLE:-**

```
===== RESTART: C:\Users\Yash Bhatnagar\Desktop\python lab prg\cmdline.py =====
['C:\\Users\\Yash Bhatnagar\\Desktop\\python lab prg\\cmdline.py']
Length of Argument is: 1
>>> |
```

#### **USING CMD:-**

```
C:\Users\Yash Bhatnagar\Desktop\python lab prg>python3 cmdline.py this is yash
['cmdline.py', 'this', 'is', 'yash']
Length of Argument is: 4
```

## **PROGRAM - 02**

**OBJECTIVE:-** Write a python program to perform Matrix Multiplication.

### **SOURCE CODE:-**

#### **METHOD 1:-**

```
R1 = int(input("Enter the number of rows in 1st matrix: "))
C1 = int(input("Enter the number of columns in 1st matrix: "))

A = []
print("Enter the values in 1st matrix:")

for i in range(R1):
    a = []
    for j in range(C1):
        a.append(int(input()))
    A.append(a)
print(A)

R2 = int(input("Enter the number of rows in 2nd matrix: "))
C2 = int(input("Enter the number of columns in 2nd matrix: "))

B = []
print("Enter the values in 2nd matrix:")

for i in range(R2):
    b = []
    for j in range(C2):
        b.append(int(input()))
    B.append(b)
print(B)

result = []

for i in range(R1):
    res = []
    for j in range(C2):
        res.append(0)
    result.append(res)

for i in range(len(A)):
    for j in range(len(B[0])):
        for k in range(len(B)):
            result[i][j] += A[i][k] * B[k][j]

print("\nFinal Matrix is:\n")
for r in result:
    print(r)
```

## **METHOD 2:- By NumPy Library**

```
import numpy as np
a=np.array([[5,8],[6,3]])
b=np.array([[3,8],[4,5]])
prod=np.dot(a,b)
print(prod)
```

## **OUTPUT:-**

### **METHOD 1:-**

```
===== RESTART: C:\Users\Yash Bhatnagar\Desktop\python lab prg\mm.py
Enter the number of rows in 1st matrix: 3
Enter the number of columns in 1st matrix: 4
Enter the values in 1st matrix:
1
2
34
5
6
8
1
9
4
1
2
5
[[1, 2, 34, 5], [6, 8, 1, 9], [4, 1, 2, 5]]
Enter the number of rows in 2nd matrix: 2
Enter the number of columns in 2nd matrix: 3
Enter the values in 2nd matrix:
4
6
5
2
7
8
[[4, 6, 5], [2, 7, 8]]

Final Matrix is:

[8, 20, 21]
[40, 92, 94]
[18, 31, 28]
>>> |
```

## **METHOD 2:- By NumPy Library**

```
=====
[[19 48]
 [13 34]]
>>> |
```

## **PROGRAM - 03**

**OBJECTIVE:-** Write a python program to compute the GCD of two numbers.

### **SOURCE CODE:-**

#### **METHOD 1:- By Function**

```
def fun_gcd(a,b):
    if(b==0):
        return a
    else:
        return fun_gcd(b,a%b)
a=int(input("Enter First Number: "))
b=int(input("Enter Second Number: "))
print("The GCD of two numbers are:",fun_gcd(a,b))
```

#### **METHOD 2:- By Math Module**

```
import math
m = int(input("Enter First Number: "))
n = int(input("Enter Second Number: "))
print("The GCD of",m,"and",n,"is:",math.gcd(m,n))
```

### **OUTPUT:-**

#### **METHOD 1:- By Function**

```
=====
Enter First Number: 6
Enter Second Number: 4
The GCD of two numbers are: 2
>>> |
```

#### **METHOD 2:- By Math Module**

```
=====
Enter First Number: 6
Enter Second Number: 4
The GCD of 6 and 4 is: 2
>>>
```

---

## PROGRAM - 04

**OBJECTIVE:-** Write a python program to find the most frequent words in a text file.

### **SOURCE CODE:-**

```
from collections import Counter
f = open("D:/ Yash.txt","r")
a=list((f.read().lower()).split(" "))
print(a)
length=len(a)
print(length)
mydict= Counter(a)
print(mydict)
max_value = max(mydict.values())
print(max_value)
print("Most frequent words in the text files are:")
for i in mydict:
    if mydict[i] == max_value:
        print(i,"occurring",mydict[i],"times")
f.close()
```

### **OUTPUT:-**

```
===== RESTART: C:\Users\Yash Bhatnagar\Desktop\python lab prg\4.py =====
['jlazy', 'parents', 'bread-butter', 'â&tagain,*', 'i', 'grumbled,-glutting', 'a', 'moe', 'plastic', 'tiffin', 'in', 'the-second', 'row', 'ragh&y-', 'audi', 'moved', 'to', 'the', 'next', 'desk,\nforget', 'it', 'gopal.', 'the', 'class', 'will', 'be', 'back', 'any', 'time', 'raghav', 'said,\nmil\ntw', 'brought', 'pori-aloo', 'we', 'can', 'share', 'drat', 'its', 'wrong', 'to', 'steal', 'from', 'others\ni', 'battl', 'ed', 'a', 'small', 'round', 'steel', 'tiffin', 'box', 'how', 'does', 'otto', 'open', 'thisf\neither', 'of', 'us', 'had', 'time', 'sharp', 'nalk', 'required', 'to', 'open', 'the', 'thin', 'steel', 'lid', 'of', 'the', 'stubborn', 'box', 'we', 'had', 'skipped', 'the', 'morning', 'assembly', 'for', 'our', 'weekly', 'tiffin', 'raid.', 'we', 'had', 'tea', 'more', 'minutes', 'till', 'the', '.national', 'anthem', 'bega', 'n', 'outside.', 'after', 'that', 'class', '5', 'c', 'would', 'be', 'back.', 'we', 'had', 'to', 'find', 'eat', 'and', 'keep', 'the', 'filing', 'back', 'within', 'that', 'time.\nif', 'a', 'pickle', 'and', 'parat', 'has-', 'raghav', 'said', 'having', 'opened', 'the', 'lid.', '4', 'you', 'want', 'it!\nforget', 'it', 'i', 'said', 'as', 'i', 'returned', 'the', 'steel', 'box', 'to', 'the', 'students', 'bag.', 'my', 'eye', 's', 'darted', 'from', 'one', 'bag', 'to', 'another,,', 'tills', 'one', 'i', 'said', 'pointing', 'to', 'a', 'pink', 'imported', 'rucksack', 'in', 'the', 'first', 'row', 'liat', 'bag', 'looks', 'expensive;', 'she', 'must', 'be', 'getting', 'good', 'food', 'come\nwe', 'xoslied', 'to', 'the', 'target's', 'seat', 'i', 'grabbed-the.', 'barme.bag.', 'nimipped', 'the', 'front', 'flap', 'and', 'found', 'a', 'rei', 'shiny', 'reetabgolar', 'tiffin/ilie', 'oo&er', 'had', 'a', 'spoon', 'compartment,', 'fancy', 'box!5l', 'said', 'clicking', 'the lid', 'open.\nidlis', 'a.', 'ilftle', 'box', 'of', 'chutney', 'a&d', 'a', 'la', 'rge', 'piece', 'of', 'chocolate', 'cake.', 'wed', 'bit', 'the', 'jackpot.\n!', 'only', 'want', 'the', 'cake"', 'i', 'said', '$&', 'l', 'lifted']
230
Counter({'the': 14, 'to': 8, 'a': 7, 'i': 6, 'we': 4, 'of': 4, 'had': 4, 'tiffin': 3, 'be': 3, 'steel': 3, 'and': 3, 'said': 3, 'in': 2, 'class': 2, 'back': 2, 'raghav': 2, 'from': 2, 'open': 2, 'that': 2, 'wa', 'nt': 2, 'said': 2, 'box': 2, 'bag': 2, 'jlazy': 1, 'parents': 1, 'bread-butter': 1, 'â&tagain,*': 1, 'grumbled,-glutting': 1, 'moe': 1, 'plastic': 1, 'the-second': 1, 'row': 1, 'ragh&y-': 1, 'audi': 1, 'moved': 1, 'next': 1, 'desk,\nforget': 1, 'it': 1, 'gopal.': 1, 'will': 1, 'any': 1, 'time': 1, 'said,\nmil\ntw': 1, 'brought': 1, 'pori-aloo': 1, 'can': 1, 'share': 1, 'drat': 1, 'its': 1, 'wrong': 1, 'steal': 1, 'others\ni': 1, 'battled': 1, 'small': 1, 'round': 1, 'box': 1, 'how': 1, 'does': 1, 'otto': 1, 'thisf\neither': 1, 'us': 1, 'had': 1, 'time': 1, 'sharp': 1, 'nalk': 1, 'required': 1, 'thin': 1, 'lid': 1, 'stubborn': 1, 'box': 1, 'skipped': 1, 'morning': 1, 'assembly': 1, 'for': 1, 'our': 1, 'weekly': 1, 'raid.': 1, 'tea': 1, 'more': 1, 'minutes': 1, 'till': 1, '.national': 1, 'anthem': 1, 'began': 1, 'outs', 'ide.': 1, 'after': 1, '5': 1, 'c': 1, 'would': 1, 'back.': 1, 'find': 1, 'eat': 1, 'keep': 1, 'filing': 1, 'within': 1, 'time.\nif': 1, 'pickle': 1, 'parathas-': 1, 'having': 1, 'opened': 1, 'the': 1, 'lid.': 1, '4': 1, 'you': 1, 'it!\nforget': 1, 'it': 1, 'as': 1, 'returned': 1, 'students': 1, 'bag.': 1, 'my': 1, 'eyes': 1, 'darted': 1, 'one': 1, 'another,,': 1, 'tills': 1, 'one': 1, 'pointing': 1, 'pink': 1, 'imported': 1, 'rucksack': 1, 'first': 1, 'row': 1, 'liat': 1, 'looks': 1, 'expensive;': 1, 'she': 1, 'must': 1, 'getting': 1, 'good': 1, 'food': 1, 'come\nwe': 1, 'xoslied': 1, 'target's': 1, 'seat': 1, 'grabbed-the.': 1, 'barme.bag.': 1, 'nimipped': 1, 'front': 1, 'flap': 1, 'found': 1, 'rei': 1, 'shiny': 1, 'reetabgolar': 1, 'tiffin/ilie': 1, 'oo&er': 1, 'spoon': 1, 'compartment,': 1, 'fancy': 1, 'box! 5l': 1, 'clicking': 1, 'the lid': 1, 'open.\nidlis': 1, 'a.': 1, 'ilftle': 1, 'chutney': 1, 'a&d': 1, 'large': 1, 'piece': 1, 'chocolate': 1, 'cake.': 1, 'wed': 1, 'bit': 1, 'jackpot.\n!': 1, 'only': 1, 'cake': 1, '$&': 1, 'l': 1, 'lifted': 1})
14
Most frequent words in the text files are:
the occurring 14 times
>>>
```

## **PROGRAM - 05**

**OBJECTIVE:-** Write a python program to find the square root of a number (Newton's method).

### **SOURCE CODE:-**

#### **METHOD 1:- BY NEWTON'S METHOD:-**

```
x=1
n=int(input("Enter Any Number: "))
for i in range(25):
    x=x-(x*x-n)/(2*x)print(round(x,4))
```

#### **METHOD 2:- BY MATH.SQRT METHOD:-**

```
import math
n=int(input("Enter Any Number: "))
print("The Square root of",n,"is: ",round(math.sqrt(n),4))
```

### **OUTPUT:-**

#### **METHOD 1:- BY NEWTON'S METHOD:-**

```
=====
Enter Any Number: 49
7.0
>>> |
```

#### **METHOD 2:- BY MATH.SQRT METHOD:-**

```
=====
Enter Any Number: 120
The Square root of 120 is: 10.9545
>>> |
```



## **PROGRAM - 06**

**OBJECTIVE:-** Write a python program exponentiation (power of a number).

### **SOURCE CODE:-**

```
n=int(input("Enter Any Number: "))
p=int(input("Enter Exponent: "))
pow=1
for i in range(1,p+1):
    pow=pow*n
print("The power of given number is: ",pow)
```

### **OUTPUT:-**

#### **BY USING ABOVE SOURCE CODE:-**

```
=====
Enter Any Number: 2
Enter Exponent: 5
The power of given number is: 32
>>> |
```

#### **BY USING ARITHMETIC OPERATOR:-**

```
=====
>>> 2**5
32
>>> 5**-3
0.008
>>> -3**3
-27
>>> |
```

#### **BY USING MATH MODULE:-**

```
=====
>>> import math
>>> math.pow(2,5)
32.0
>>> math.pow(5,3)
125.0
>>> |
```

## **PROGRAM - 07**

**OBJECTIVE:-** Write a python program to find the maximum of a list of numbers.

### **SOURCE CODE:-**

```
list=[]
n=int(input("Enter the size of List: "))
print("Enter the elements of List: ")

for i in range(0,n):
    list.append(int(input()))
max=list[0]
print(f"initial_max={max}")
for i in range(1,n):
    if list[i]>max:
        max=list[i]
        print(f"max={max}")
        print(list[i])
print(f"Maximum element of the list is {max}")
```

### **OUTPUT:-**

```
===== RES
Enter the size of List: 6
Enter the elements of List:
4
3
8
6
1
9
initial_max=4
max=8
8
max=9
9
Maximum element of the list is 9
>>> |
```

## **PROGRAM - 08**

**OBJECTIVE:-** Write a python program Linear search.

### **SOURCE CODE:-**

```
list=[]
n=int(input("Enter the size of List: "))
print("Enter the elememts of List: ")

for i in range(0,n):
    list.append(int(input()))

key=int(input("Enter the number that to be searched: "))

for i in range(0,n):
    if list[i]==key:
        print(f"The element found at index: {i}")
        break
    else:
        print("Element does not exist")
```

### **OUTPUT:-**

```
=====
Enter the size of List: 7
Enter the elememts of List:
2
4
1
5
7
6
3
Enter the number that to be searched: 6
The element found at index: 5
>>> |
```

## **PROGRAM - 09**

**OBJECTIVE:-** Write a python program Binary search.

### **SOURCE CODE:-**

```
list=[]
n=int(input("Enter the size of List: "))
print("Enter the elememts of List: ")

for i in range(0,n):
    list.append(int(input()))

x = int(input("Enter the element to be searched: "))
lower_bound = 0
upper_bound = len(list)-1
mid = (lower_bound + upper_bound)//2

while lower_bound <= upper_bound:
    if x == list[mid]:
        print("Element Found at index:\t",mid)
        break
    else:
        if x > list[mid]:
            lower_bound = mid+1
            mid = (lower_bound+upper_bound)//2
        elif x < list[mid]:
            upper_bound = mid-1
            mid = (lower_bound+upper_bound)//2

if(lower_bound>upper_bound):
    print("Element Not Found")
```

### **OUTPUT:-**

```
=====
Enter the size of List: 6
Enter the elememts of List:
1
3
4
6
8
10
Enter the element to be searched: 4
Element Found at index: 2
>>> |
```

## **PROGRAM - 10**

**OBJECTIVE:-** Write a python program Selection Sort.

**SOURCE CODE:-**

**OUTPUT:-**

## **PROGRAM - 11**

**OBJECTIVE:-** Write a python program Insertion Sort.

**SOURCE CODE:-**

**OUTPUT:-**

## **PROGRAM - 12**

**OBJECTIVE:-** Write a python program Merge Sort.

**SOURCE CODE:-**

**OUTPUT:-**

### **PROGRAM - 13**

**OBJECTIVE:-** Write a python program first n prime numbers.

**SOURCE CODE:-**

**OUTPUT:-**



## **PROGRAM - 14**

**OBJECTIVE:-** Write a python program simulate bouncing ball in pygame.

**SOURCE CODE:-**

**OUTPUT:-**

## **PROGRAM - 15**

**OBJECTIVE:-** Write a python program to check a number is Palindrome.

### **SOURCE CODE:-**

```
n=int(input("Enter Any Number: "))
rev=n
palindrome=0
while rev>0:
    a=rev%10
    print("a: ",a)
    palindrome=palindrome*10+a
    print("palindrome: ",palindrome)
    rev=rev//10
    print("reverse: ",rev)
if n==palindrome:
    print(n, " is palindrome")
else:
    print(n, " is not palindrome")
```

### **OUTPUT:-**

```
=====
Enter Any Number: 121
a: 1
palindrome: 1
reverse: 12
a: 2
palindrome: 12
reverse: 1
a: 1
palindrome: 121
reverse: 0
121 is palindrome
>>> |
```

## **PROGRAM - 16**

**OBJECTIVE:-** Write a python program to implement Scipy library demonstration.

### **SOURCE CODE:-**

#### **METHOD 1:- METHOD OF INTERPOLATION**

```
import matplotlib.pyplot as plt
from scipy import interpolate
import numpy as np
x = np.arange(7,35)
print(x)
y = np.exp(x/3.0)
print(y)
f = interpolate.interp1d(x, y)
x1 = np.arange(8, 15)
print(x1)
y1 = f(x1)
print(y1)
plt.plot(x, y, 'o', x1, y1, '--')
plt.show()
```

#### **METHOD 2:- SPECIAL MODULE METHOD**

```
from scipy import special
c = special.sindg(45)
print(c)
d = round(special.tandg(10),4)
print(d)
```

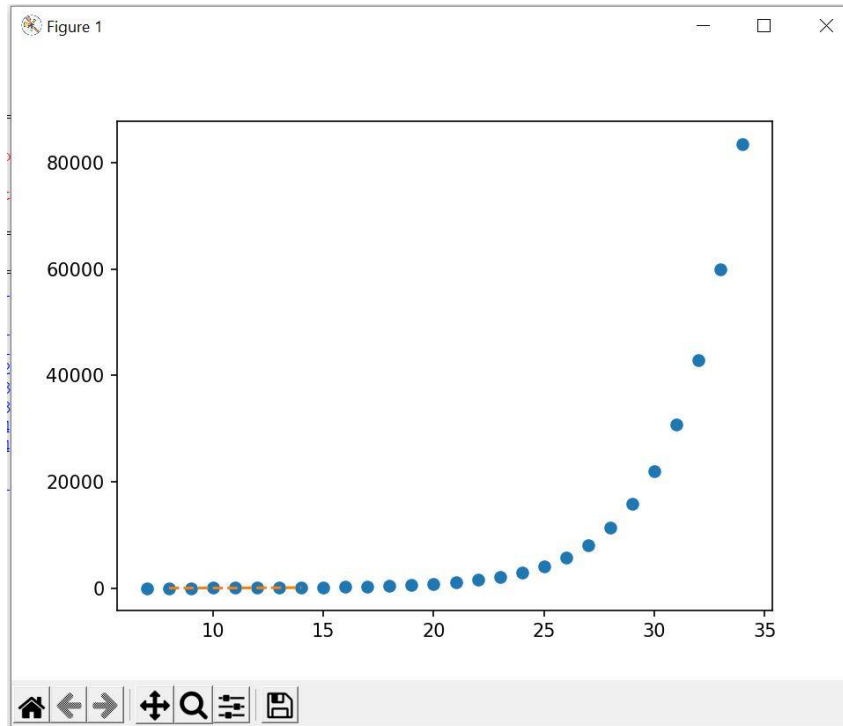
#### **METHOD 3:- LINEAR ALGEBRA PROGRAM**

```
# Program on linear algebra
import numpy as np
from scipy import linalg
from scipy.linalg import eigh
A = np.array([[6,3], [5,2]])
B = linalg.inv(A)
print(B)
c = linalg.det(A)
print(c)
D = np.array([[1, 3, 2, 4], [1, 5, 6, 3], [6, 3, 4, 1], [4, 3, 2, 5]])
a, b = eigh(D)

print("Selected eigenvalues:", a)
print("Complex ndarray:", b)
```

## OUTPUT:-

### METHOD 1:- METHOD OF INTERPOLATION



```
===== RESTA
[ 7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
 31 32 33 34]
[1.03122585e+01 1.43919161e+01 2.00855369e+01 2.80316249e+01
 3.91212840e+01 5.45981500e+01 7.61978566e+01 1.06342675e+02
 1.48413159e+02 2.07127249e+02 2.89069362e+02 4.03428793e+02
 5.63030237e+02 7.85771994e+02 1.09663316e+03 1.53047486e+03
 2.13594973e+03 2.98095799e+03 4.16026201e+03 5.80611335e+03
 8.10308393e+03 1.13087646e+04 1.57826524e+04 2.20264658e+04
 3.07404093e+04 4.29016972e+04 5.98741417e+04 8.35610961e+04]
[ 8  9 10 11 12 13 14]
[ 14.3919161  20.08553692  28.03162489  39.121284   54.59815003
 76.19785657 106.3426754 ]
>>> |
```

### METHOD 2:- SPECIAL MODULE METHOD

```
=====
0.7071067811865476
0.1763
>>> |
```

### METHOD 3:- LINEAR ALGEBRA PROGRAM

```
===== ]
[[-0.66666667  1.          ]
 [ 1.66666667 -2.          ]]
-3.0
Selected eigenvalues: [-4.42722137  2.52305677  3.5450208  13.3591438 ]
Complex ndarray: [[ 0.771467    0.07486071 -0.41774364 -0.47405146]
 [ 0.17804901 -0.45991801  0.74882712 -0.44275492]
 [-0.54916932 -0.42258584 -0.45814052 -0.55672394]
 [-0.26748883  0.77736253  0.23421243 -0.51894292]]
>>> |
```