

# PYTHON LAB MANUAL

**DEPARTMENT OF CIVIL ENGINEERING**



**MARRI LAXMAN REDDY**  
**INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

**2022-2023**



# **MARRI LAXMAN REDDY**

## **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

## **CERTIFICATE**

This is to certify that this manual is a bonafide record of practical work in the **Python Lab in Second Semester of Second Year B. Tech (Civil) programme** during the academic year **2022-23**. The book is prepared by **Mr. Amish Kumar Aman, Assistant Professor, Department of Civil Engineering.**

Signature of HOD

Signature of Dean Academics

Signature of Principal



# **MARRI LAXMAN REDDY**

## **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

### **INDEX**

<b><u>S. No</u></b>	<b><u>CONTENT</u></b>	<b><u>PAGE NO</u></b>
1	PREFACE	i
2	ACKNOWLEDGEMENT	ii
3	GENERAL INSTRUCTIONS	iii
4	SAFETY PRECAUTIONS	iv
5	INSTITUTE VISION AND MISSION	v
6	DEPARTMENT VISION AND MISSION	vi
7	PROGRAMME EDUCATIONAL OBJECTIVES	vi
8	PROGRAM SPECIFIC OUTCOMES	vi
9	PROGRAMME OUTCOMES	vii - viii
10	COURSE STRUCTURE, OBJECTIVES AND OUTCOMES	ix – x
11	BASICS	1
12	RUNNING INSTRUCTIONS IN INTERACTIVE INTERPRETER	1
13	WRITE A PROGRAM TO PURPOSEFULLY RAISE INDENTATION ERROR AND CORRECT IT	1
14	OPERATIONS	2
15	WRITE A PROGRAM TO COMPUTE DISTANCE BETWEEN TWO POINTS TAKING INPUT FROM THE USER (PYTHAGOREAN THEOREM)	2
16	WRITE A PROGRAM ADD.PY THAT TAKES 2 NUMBERS AS COMMAND LINE ARGUMENTS AND PRINTS ITS SUM.	2
17	CONTROL FLOW	3
18	WRITE A PROGRAM FOR CHECKING WHETHER THE GIVEN NUMBER IS A EVEN NUMBER OR NOT.	3
19	USING A FOR LOOP, WRITE A PROGRAM THAT PRINTS OUT THE DECIMAL EQUIVALENTS OF 1/2, 1/3, 1/4, . . . , 1/10	3 – 4
20	WRITE A PROGRAM USING A FOR LOOP THAT LOOPS OVER A SEQUENCE. WHAT IS SEQUENCE?	4 – 5



# MARRI LAXMAN REDDY INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)


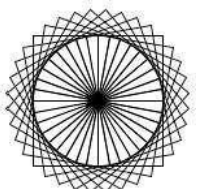
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

## INDEX

21	WRITE A PROGRAM USING A WHILE LOOP THAT ASKS THE USER FOR A NUMBER, AND PRINTS A COUNTDOWN FROM THAT NUMBER TO ZERO.	4 – 5
22	<b>CONTROL FLOW - CONTINUED</b>	<b>6</b>
23	FIND THE SUM OF ALL THE PRIMES BELOW TWO MILLION. EACH NEW TERM IN THE FIBONACCI SEQUENCE IS GENERATED BY ADDING THE PREVIOUS TWO TERMS. BY STARTING WITH 1 AND 2, THE FIRST 10 TERMS WILL BE: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...	6
24	BY CONSIDERING THE TERMS IN THE FIBONACCI SEQUENCE WHOSE VALUES DO NOT EXCEED FOUR MILLION, FIND THE SUM OF THE EVEN-VALUED TERMS.	6 – 7
25	<b>FILES</b>	<b>8</b>
26	WRITE A PROGRAM TO PRINT EACH LINE OF A FILE IN REVERSE ORDER	8
27	WRITE A PROGRAM TO COMPUTE THE NUMBER OF CHARACTERS, WORDS AND LINES IN A FILE.	8
28	<b>FUNCTIONS</b>	<b>9</b>
29	WRITE A FUNCTION BALL_COLLIDE THAT TAKES TWO BALLS AS PARAMETERS AND COMPUTES IF THEY ARE COLLIDING. YOUR FUNCTION SHOULD RETURN A BOOLEAN REPRESENTING WHETHER OR NOT THE BALLS ARE COLLIDING. HINT: REPRESENT A BALL ON A PLANE AS A TUPLE OF (x, y, r), r BEING THE RADIUS IF (DISTANCE BETWEEN TWO BALLS CENTERS) <= (SUM OF THEIR RADII) THEN (THEY ARE COLLIDING)	9
30	FIND MEAN, MEDIAN, MODE FOR THE GIVEN SET OF NUMBERS IN A LIST.	9 – 10
31	<b>FUNCTIONS - CONTINUED</b>	<b>11</b>
32	WRITE A FUNCTION NEARLY_EQUAL TO TEST WHETHER TWO STRINGS ARE NEARLY EQUAL. TWO STRINGS A AND B ARE NEARLY EQUAL WHEN A CAN BE GENERATED BY A SINGLE MUTATION ON B.	11
33	WRITE A FUNCTION DUPS TO FIND ALL DUPLICATES IN THE LIST	11 – 12

## INDEX

34	WRITE A FUNCTION UNIQUE TO FIND ALL THE UNIQUE ELEMENTS OF A LIST	12
35	<b>FUNCTIONS – PROBLEM SOLVING</b>	13
36	WRITE A FUNCTION CUMULATIVE PRODUCT TO COMPUTE CUMULATIVE PRODUCT OF A LIST OF NUMBERS.	13
37	WRITE A FUNCTION REVERSE TO REVERSE A LIST. WITHOUT USING THE REVERSE FUNCTION.	13
38	WRITE FUNCTION TO COMPUTE gcd, lcm OF TWO NUMBERS. EACH FUNCTION SHOULDN'T EXCEED ONE LINE	14
39	<b>MULTI – D - LISTS</b>	15
40	WRITE A PROGRAM THAT DEFINES A MATRIX AND PRINTS	15
41	WRITE A PROGRAM TO PERFORM ADDITION OF TWO SQUARE MATRICES	16
42	WRITE A PROGRAM TO PERFORM MULTIPLICATION OF TWO SQUARE MATRICES	16 – 17
43	<b>GUI, GRAPHICS</b>	18
44	WRITE A GUI FOR AN EXPRESSION CALCULATOR USINGTK  WRITE A PROGRAM TO IMPLEMENT THE FOLLOWING FIGURES USING TURTLE	18 – 21
45	<div style="display: flex; justify-content: space-around; align-items: center;">   </div>	21 – 22

## **PREFACE**

This book entitled “Python Programming Lab Manual” is intended for the use of fourth semester (i.e., II - II) B. Tech (civil) students of Marri laxman Reddy Institute of Technology and Management, Dundigal, Hyderabad. The main objective of the Python Programming Lab Manual is to teach the student basic python fundamentals in various engineering applications. This book lays foundation of certain basic concepts and skills that can be repeatedly employed by the students in their future endeavours. The main aim of this book is to develop the habit of scientific reasoning and providing answers to all the doubts that arise during the course of conducting experiments. The book was written as per the new syllabus prescribed by the JNTUH university in a simple language. Some of the additional experiments apart from the syllabus also included in the book. These experiments will help the students to expertise in the analysis and reporting the water quality for drinking purpose. Hence, we hope this book serve for better understanding by the student community with all details of experiments

By,  
Mr. Basavaraj Deshmukh  
Asst Professor, Department of civil engineering

## **ACKNOWLEDGEMENT**

It was really a good experience, working at Python Programming Lab. First, I would like to thank Ms. Nanditha Mandava, Asst. Professor, Department of Civil Engineering, Marri Laxman Reddy Institute of technology & Management for giving the technical support in preparing the document.

I express my sincere thanks to Mr. K. Murali, Head of the Department of Civil Engineering, Marri Laxman Reddy Institute of technology & Management, for his concern towards me and gave me opportunity to prepare Python Programming laboratory manual.

I am deeply indebted and gratefully acknowledge the constant support and valuable patronage of Dr. Balarengadurai Chinnaiah, Dean Academics, Marri Laxman Reddy Institute of technology & Management. I am unboundedly grateful to him for timely corrections and scholarly guidance.

I express my hearty thanks to Dr. K. Venkateswara Reddy, Principal, Marri Laxman Reddy Institute of technology & Management, for giving me this wonderful opportunity for preparing the Python Programming laboratory manual.

At last, but not the least I would like to thank the entire Civil Department faculties those who had inspired and helped me to achieve my goal.

By,  
Mr. Basavaraj Deshmukh  
Asst Professor, Department of civil engineering

## **GENERAL INSTRUCTIONS**

1. Students are instructed to come to Python Programming laboratory on time. Late comers are not entertained in the lab.
2. Students should be punctual to the lab. If not, conducted experiments will not be repeated.
3. Students are expected to come prepared at home with the experiments which are going to be performed.
4. Students are instructed to display their identity cards and apron before entering into the lab.
5. Students are instructed not to bring mobile phones to the lab.
6. The equipment's and other accessories used in Python Programming lab should be handled with care and responsibility.
7. Any damage to the equipment's during the lab session is student's responsibility and penalty or fine will be collected from the student.
8. Students should update the records and lab observation books session wise. Before leaving the lab, the student should get his lab observation book signed by the faculty.
9. Students should submit the lab records 2/3 days in advance to the concerned faculty members in the staffroom for their correction and return.
10. Students should not move around the lab during the lab session.
11. If any emergency arises, the student should take the permission from faculty member concerned in written format.
12. The faculty members may suspend any student from the lab session on disciplinary grounds.



## **SAFETY PRECAUTIONS**

1. While working in the laboratory suitable precautions should be observed to prevent accidents.
2. Always follow the experimental instructions strictly.
3. Use the first aid box in case of any accident/mishap.
4. Never work in the laboratory unless a demonstrator or teaching assistant is present.
5. When the experiment is completed, students should disconnect the setup made by them, and should return all the components/instruments taken for the purpose.

# **INSTITUTION VISION AND MISSION**

## **VISION**

To establish as an ideal academic institution in the service of the nation, the world and the humanity by graduating talented engineers to be ethically strong, globally competent by conducting high quality research, developing breakthrough technologies, and disseminating and preserving technical knowledge.

## **OUR MISSION**

To fulfil the promised vision through the following strategic characteristics and aspirations:

- Contemporary and rigorous educational experiences that develop the engineers and managers.
- An atmosphere that facilitates personal commitment to the educational success of students in an environment that values diversity and community.
- Prudent and accountable resource management.
- Undergraduate programs that integrate global awareness, communication skills and team building.
- Leadership and service to meet society's needs
- Education and research partnerships with colleges, universities, and industries to graduate.
- Education and training that prepares students for interdisciplinary engineering research and advanced problem-solving abilities.
- Highly successful alumni who contribute to the profession in the global society.

## **DEPARTMENT VISION, MISSION, PROGRAMME EDUCATIONAL OBJECTIVES**

### **AND SPECIFIC OUTCOMES**

#### **VISION**

The Civil Engineering department strives to impart quality education by extracting the innovative skills of students and to face the challenges in latest technological advancements and to serve the society.

#### **MISSION**

Provide quality education and to motivate students towards professionalism

Address the advanced technologies in research and industrial issues

#### **PROGRAMME EDUCATIONAL OBJECTIVES**

The Programme Educational Objectives (PEOs) that are formulated for the civil engineering programme are listed below;

**PEO-I** solving civil engineering problems in different circumstances **PEO-II** Pursue higher education and research for professional development.

**PEO-III** Inculcate qualities of leadership for technology innovation and entrepreneurship.

#### **PROGRAM SPECIFIC OUUTCOMES**

##### **PSO1-UNDERSTANDING:**

Graduates will have ability to describe, analyse and solve problems using mathematical, scientific, and engineering knowledge.

##### **PSO2-ANALYTICAL SKILLS**

Graduates will have an ability to plan, execute, maintain, manage, and rehabilitate civil engineering systems and processes.

##### **PSO3-EXECUTIVE SKILLS**

Graduates will have an ability to interact and work effectively in multi-disciplinary teams.

## **PROGRAMME OUT COMES**

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

# **COURSE STRUCTURE, OBJECTIVES & OUTCOMES**

## **COURSE STRUCTURE**

Concrete Technology lab will have a continuous evaluation during 4<sup>th</sup> semester for 30 sessional marks and 70 end semester examination marks.

Out of the 30 marks for internal evaluation, day-to-day work in the laboratory shall be evaluated for 15 marks and internal practical examination shall be evaluated for 15 marks conducted by the laboratory teacher concerned.

The end semester examination shall be conducted with an external examiner and internal examiner. The external examiner shall be appointed by the principal / Chief Controller of examinations

## **COURSE OBJECTIVE**

The objective of this lab is to teach the student basic python fundamentals in various engineering applications.

## **COURSE OUTCOME**

At the end of the course, the student will be able to: Master the design of various mix proportions for concrete required for different civil engineering applications.

**CE229.1:** Interpret the fundamental Python syntax and semantics and be fluent in the use of Python control flow statements.

**CE229.2:** Express proficiency in the handling of strings and functions

**CE229.3:** Determine the methods to create and manipulate Python programs by utilizing the data structures like lists, dictionaries, tuples and sets.

**CE229.4:** Identify the commonly used operations involving file systems and regular expressions

**CE229.5:** Articulate the Object-Oriented Programming concepts such as encapsulation, inheritance and polymorphism as used in Python.

## **COURSE ARTICULATION MATRIX (CO - PO / PSO MAPPING):**

<b>Program outcomes</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>PSO 1</b>	<b>PSO 2</b>	<b>PSO 3</b>
CE229.1	3	3	3	2	0	0	0	0	0	0	0	1	3	1	0
CE229.2	3	3	3	2	0	0	0	0	0	0	0	1	3	1	0
CE229.3	3	3	3	2	0	0	0	0	0	0	0	1	3	1	0
CE229.4	3	3	3	2	0	0	0	0	0	0	0	1	3	1	0
CE229.5	3	3	3	2	0	0	0	0	0	0	0	1	3	1	0
<b>Total</b>	<b>15</b>	<b>15</b>	<b>15</b>	<b>10</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>5</b>	<b>15</b>	<b>5</b>	<b>0</b>
<b>Average</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>2</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>1</b>	<b>3</b>	<b>1</b>	<b>0</b>

## **LIST OF EXPERIMENTS**

### **Exercise 1 - Basics**

- Running instructions in Interactive interpreter and a Python Script
- Write a program to purposefully raise Indentation Error and Correct it

### **Exercise 2 -Operations**

- Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

- b) Write a program `add.py` that takes 2 numbers as command line arguments and prints its sum.

**Exercise - 3 Control Flow**

- a) Write a Program for checking whether the given number is a even number or not.
- b) Using a for loop, write a program that prints out the decimal equivalents of  $1/2$ ,  $1/3$ ,  $1/4$ , . . . ,  $1/10$

- c) Write a program using a for loop that loops over a sequence. What is sequence?
- d) Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero.

#### Exercise 4 - Control Flow -Continued

- a) Find the sum of all the primes below two million.  
Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...
- b) By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

#### Exercise - 5 Files

- a) Write a program to print each line of a file in reverse order.
- b) Write a program to compute the number of characters, words and lines in a file.

#### Exercise - 6 Functions

- a) Write a function ball\_collide that takes two balls as parameters and computes if they are colliding. Your function should return a Boolean representing whether or not the balls are colliding.  
Hint: Represent a ball on a plane as a tuple of (x, y, r), r being the radius  
If (distance between two balls centers)  $\leq$  (sum of their radii) then (they are colliding)
- b) Find mean, median, mode for the given set of numbers in a list.

#### Exercise - 7 Functions - Continued

- a) Write a function nearly\_equal to test whether two strings are nearly equal. Two strings a and b are nearly equal when a can be generated by a single mutation on b.
- b) Write a function dups to find all duplicates in the list.
- c) Write a function unique to find all the unique elements of a list.

#### Exercise - 8 - Functions - Problem Solving

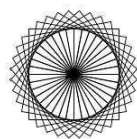
- a) Write a function cumulative\_product to compute cumulative product of a list of numbers.
- b) Write a function reverse to reverse a list. Without using the reverse function.
- c) Write function to compute gcd, lcm of two numbers. Each function shouldn't exceed one line.

#### Exercise 9 - Multi-D Lists

- a) Write a program that defines a matrix and prints
- b) Write a program to perform addition of two square matrices
- c) Write a program to perform multiplication of two square matrices

#### Exercise - 10 GUI, Graphics

1. Write a GUI for an Expression Calculator usingtk
2. Write a program to implement the following figures using turtle



## EXPERIMENT 1

### BASICS

#### a) RUNNING INSTRUCTIONS IN INTERACTIVE INTERPRETER AND A PYTHON SCRIPT

##### AIM:

Running instructions in Interactive interpreter and a Python Script

##### PROCEDURE:

```
>>> 3+5
```

```
8
```

```
>>> 4-2
```

```
2
```

```
>>> 8/2
```

```
4.0
```

#### b) WRITE A PROGRAM TO PURPOSEFULLY RAISE INDENTATION ERROR AND CORRECT IT

##### AIM:

Write a program to purposefully raise Indentation Error and Correct it.

##### PROCEDURE:

```
n1=int(input("enter n1 value"))
n2=int(input("enter n2 value")) if
n1>n2:
print("n1 is big")
else:
print("n2 is big")
```

##### OUTPUT:

Error: Excepted an indented block

##### CORRECT PROGRAM:

```
n1=int(input("enter n1 value"))
n2=int(input("enter n1 value")) if
n1>n2:
    print("n1 is big")else:
    print("n2 is big")
```

##### OUTPUT:

```
enter n1 value10
enter n1 value20
n2 is big
```



## EXPERIMENT 2

### OPERATIONS

a) **WRITE A PROGRAM TO COMPUTE DISTANCE BETWEEN TWO POINTS TAKING INPUT FROM THE USER (PYTHAGOREAN THEOREM)**

**AIM:**

Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

**PROCEDURE:**

```
import math; x1=int(input("Enter
x1--->"))
y1=int(input("Enter y1--->"))

x2=int(input("Enter x2--->"))
y2=int(input("Enter y2--->"))

d1 = (x2 - x1) * (x2 - x1);
d2 = (y2 - y1) * (y2 - y1);
res = math.sqrt(d1+d2)
print ("Distance between two points:",res);
```

**OUTPUT:**

```
Enter x1--->10
Enter y1--->20
Enter x2--->15
Enter y2--->19
Distance between two points: 5.0990195135927845
```

b) **WRITE A PROGRAM ADD.PY THAT TAKES 2 NUMBERS AS COMMAND LINE ARGUMENTS AND PRINTSITS SUM.**

**AIM:**

Write a program add.py that takes 2 numbers as command line arguments and printsits sum.

**PROCEDURE:**

```
step 1: open notepad and write the below program
import sys;
n1=int(sys.argv[1]);
n2=int(sys.argv[2]);
print (n1+n2)
step 2: SAVE
step 3: Open DOS SHELL and go to file saved location (D:\)step 4:
python filename.py argument1 argument2
```

**OUTPUT:**

```
python filename.py 10 20
30
```

## EXPERIMENT 3

### CONTROL FLOW

#### a) WRITE A PROGRAM FOR CHECKING WHETHER THE GIVEN NUMBER IS A EVEN NUMBER OR NOT.

##### AIM:

Write a Program for checking whether the given number is a even number or not.

##### PROCEDURE:

```
n=int(input("Enter a number --- >"))if n %
2 == 0:
    print ("EVEN Number");
else:
    print ("ODD Number");
```

##### OUTPUT:

Enter a number--- >10

EVEN Number

Enter a number--- >11

ODD Number

#### b) USING A FOR LOOP, WRITE A PROGRAM THAT PRINTS OUT THE DECIMAL EQUIVALENTS OF 1/2, 1/3, 1/4,.....,1/10

##### AIM:

Using a for loop, write a program that prints out the decimal equivalents of 1/2, 1/3, 1/4, . . . , 1/10

##### PROCEDURE:

```
i=1;
for j in range(2,10):
    print("i:",i,"j:",j)
    print(i,"/",j) print
    (i/j);
```

##### OUTPUT:

i: 1 j: 2

1 / 2

0.5

i: 1 j: 3

1 / 3

0.3333333333333333

i: 1 j: 4

1 / 4

0.25

i: 1 j: 5  
1 / 5  
0.2  
i: 1 j: 6  
1 / 6  
0.166666666666666666  
i: 1 j: 7  
1 / 7  
0.14285714285714285  
i: 1 j: 8  
1 / 8  
0.125  
i: 1 j: 9  
1 / 9  
0.111111111111111111

**c) WRITE A PROGRAM USING A FOR LOOP THAT LOOPS OVER A SEQUENCE. WHAT IS SEQUENCE?**

**AIM:**

Write a program using a for loop that loops over a sequence. What is sequence?

**PROCEDURE:**

```
str="i am python developer"
for i in str:
    print(i)
```

**OUTPUT:**

i  
a  
m  
python  
developer

**d) WRITE A PROGRAM USING A WHILE LOOP THAT ASKS THE USER FOR A NUMBER, AND PRINTS A COUNTDOWN FROM THAT NUMBER TO ZERO.**

**AIM:**

Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero.

**PROCEDURE:**

```
n = int(input("Enter A Number--->"));
while n >=0:
    print (n);n =
    n - 1;
```

**OUTPUT:**

Enter A  
Number--->10

10

9

8

7

6

5

4

3

2

1

0

## EXPERIMENT 4

### CONTROL FLOW -CONTINUED

a) **FIND THE SUM OF ALL THE PRIMES BELOW TWO MILLION. EACH NEW TERM IN THE FIBONACCI SEQUENCE IS GENERATED BY ADDING THE PREVIOUS TWO TERMS. BY STARTING WITH 1 AND 2, THE FIRST 10 TERMS WILL BE: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...**

#### AIM:

Find the sum of all the primes below two million.

Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

#### PROCEDURE:

```
a, b = 1, 2
total = 0
print(a,end=" ")
while (a <=2000000-1):
    if a % 2 != 0:
        total += a
    a, b = b, a+b
    print(a,end=" ")
print("\n sum of prime numbers term in fibonacci series: ",total)
```

#### OUTPUT:

```
1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657
46368
75025 121393 196418 317811 514229 832040 1346269 2178309
sum of prime numbers term in fibonacci series: 2435422
```

b) **BY CONSIDERING THE TERMS IN THE FIBONACCI SEQUENCE WHOSE VALUES DO NOT EXCEED FOUR MILLION, FIND THE SUM OF THE EVEN-VALUED TERMS.**

#### AIM:

By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

#### PROCEDURE:

```
a, b = 1, 2
total = 0
print(a,end=" ")
while (a <=4000000-1):
    if a % 2 == 0:
        total += a
    a, b = b, a+b
    print(a,end=" ")
print("\n sum of prime numbers term in fibonacci series: ",total)
```

**OUTPUT:**

1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657  
46368

75025 121393 196418 317811 514229 832040 1346269 2178309 3524578  
5702887

sum of prime numbers term in Fibonacci series: 4613732

## EXPERIMENT 5

### FILES

#### a) WRITE A PROGRAM TO PRINT EACH LINE OF A FILE IN REVERSE ORDER.

##### AIM:

Write a program to print each line of a file in reverse order.

##### PROCEDURE:

```
input_file=open('D:/a.txt','r')
for line
in input_file:
    l=len(line)
    s=' '
    while(l>=1):
        s=s+line[l-1]
        l=l-1
    print(s)
input_file.close()
a.txt file contain khit
```

##### OUTPUT:

tihk

#### b) WRITE A PROGRAM TO COMPUTE THE NUMBER OF CHARACTERS, WORDS AND LINES IN A FILE.

##### AIM:

Write a program to compute the number of characters, words and lines in a file.

##### PROCEDURE:

```
k=open('D:/a.txt','r')
char,wc,lc=0,0,0
for line in k:
    for k in range(0,len(line)):
        char
        +=1
        if(line[k]==' '):
            wc+=1
        if(line[k]=='\n'):
            wc,lc=wc+1,lc+1
print("The no.of chars is %d\n The no.of words is %d\n Theno.of
lines is %d"%(char,wc,lc))
```

a.txt contain

khit

Guntur

Hyderabad

##### OUTPUT:

The no. of chars is 21  
The no. of words is 2  
The no. of lines is 2

## EXPERIMENT 6

### FUNCTIONS

a) **WRITE A FUNCTION BALL\_COLLIDE THAT TAKES TWO BALLS AS PARAMETERS AND COMPUTES IF THEY ARE COLLIDING. YOUR FUNCTION SHOULD RETURN A BOOLEAN REPRESENTING WHETHER OR NOT THE BALLS ARE COLLIDING.**

**HINT: REPRESENT A BALL ON A PLANE AS A TUPLE OF (X, Y, R), R BEING THE RADIUS IF (DISTANCE BETWEEN TWO BALLS CENTERS)  $\leq$  (SUM OF THEIR RADII) THEN (THEY ARE COLLIDING)**

#### AIM:

Write a function ball collide that takes two balls as parameters and computes if they are colliding. Your function should return a Boolean representing whether or not the balls are colliding.

Hint: Represent a ball on a plane as a tuple of (x, y, r), r being the radius

If (distance between two balls centers)  $\leq$  (sum of their radii) then (they are colliding)

#### PROCEDURE:

```
import math
def ball_collide(x1,y1,r1,x2,y2,r2):
    dist=math.sqrt((x2-x1)**2+(y2-y1)**2);
    print("Distance b/w two balls:",dist)
    center=dist/2;
    print("Collision point",center);
    r=r1+r2;
    print("Sum of raiodius",r)
    if(center<=r):
        print("They are Colliding")return
        True;
    else:
        print("Not Colliding")return
        False;

c=ball_collide(4,4,3,2,2,3)print(c)
c=ball_collide(100,200,20,200,100,10)
print(c)
```

#### OUTPUT:

```
Distance b/w two balls: 2.8284271247461903
Collision point 1.4142135623730951
Sum of raiodius 6
They are CollidingTrue
Distance b/w two balls: 141.4213562373095
Collision point 70.71067811865476
Sum of raiodius 30
Not Colliding False
```



b) **FIND MEAN, MEDIAN, MODE FOR THE GIVEN SET OF NUMBERS IN A LIST.**

**AIM:**

Find mean, median, mode for the given set of numbers in a list.

**PROCEDURE:**

```
from statistics import mean,median,mode =  
[15, 18, 2, 36, 12, 78, 5, 6, 9,18]  
print("Mean",mean(l))  
print("Median",median(l))  
print("Mode",mode(l))
```

**OUTPUT:**

Mean 19.9  
Median 13.5  
Mode 18

## EXPERIMENT 7

### FUNCTIONS - CONTINUED

- a) WRITE A FUNCTION NEARLY\_EQUAL TO TEST WHETHER TWO STRINGS ARE NEARLY EQUAL. TWO STRINGS a AND b ARE NEARLY EQUAL WHEN a CAN BE GENERATED BY A SINGLE MUTATION ON b.

#### AIM:

Write a function nearly\_equal to test whether two strings are nearly equal. Two strings a and b are nearly equal when a can be generated by a single mutation on b.

#### PROCEDURE:

```
from difflib import SequenceMatcher
def Nearly_Equal(a,b):
    return SequenceMatcher(None,a,b).ratio();
a="khit"
b="khitc" c=Nearly_Equal(a,b)
if(c*100>80):
    print("Both Strings are similar")
    print("a is mutation on b")
else:
    print("Both Strings are Different")
```

#### OUTPUT:

Both Strings are similar  
a is mutation on b

- b) WRITE A FUNCTION DUPS TO FIND ALL DUPLICATES IN THE LIST

#### AIM:

Write a function dups to find all duplicates in the list.

#### PROCEDURE:

```
def FindDuplicates(list):
    for i in list:
        count = list.count(i)
        if count > 1:
            print ('There are duplicates in list')
            return True
    print ('There are no duplicates in list' )
    return False

a = [8, 64, 16, 32, 4, 24]
b = [2,2,3,6,78,65,4,4,5]
print(a)
```

```
FindDuplicates(a)
print(b)
FindDuplicates(b)
```

### OUTPUT:

```
[8, 64, 16, 32, 4, 24]
There are no duplicates in list[2, 2, 3,
6, 78, 65, 4, 4, 5]
There are duplicates in list
```

### c) WRITE A FUNCTION UNIQUE TO FIND ALL THE UNIQUE ELEMENTS OF A LIST.

#### AIM:

Write a function unique to find all the unique elements of a list.

#### PROCEDURE:

```
def FindUnique(list):
    unique = set(list)
    for i in unique:
        count = list.count(i)
        if count > 1:
            print('There are no unique elements in list')
            return True
    print('There are unique elements in list')
    return False
a = [8, 64, 16, 32, 4, 24]
b = [2, 2, 3, 6, 78, 65, 4, 4, 5]
print(a)
FindUnique(a)
print(b)
FindUnique(b)
```

### OUTPUT:

```
[8, 64, 16, 32, 4, 24]
There are unique elements in list[2, 2, 3,
6, 78, 65, 4, 4, 5]
There are no unique elements in list
```

## EXPERIMENT 8

### FUNCTIONS - PROBLEM SOLVING

#### a) WRITE A FUNCTION CUMULATIVE PRODUCT TO COMPUTE CUMULATIVE PRODUCT OF A LIST OF NUMBERS.

##### AIM:

Write a function cumulative\_product to compute cumulative product of a list of numbers.

##### PROCEDURE:

```
def product(list):p =1
    for i in list:p *= i
        print(p)
    return p
arr= [1,2,3,4,5,6,7,8,9,10]
c=product(arr)
```

##### OUTPUT:

```
1
2
6
24
120
720
5040
40320
362880
3628800
```

#### b) WRITE A FUNCTION REVERSE TO REVERSE A LIST. WITHOUT USING THE REVERSE FUNCTION

##### AIM:

Write a function reverse to reverse a list. Without using the reverse function.

##### PROCEDURE:

```
l = [1,2,3,4,5]
print (l[::-1])
```

##### OUTPUT:

```
[5, 4, 3, 2, 1]
```

c) **WRITE FUNCTION TO COMPUTE GCD, LCM OF TWO NUMBERS. EACH FUNCTION SHOULDN'T EXCEED ONE LINE**

**AIM:**

Write function to compute gcd, lcm of two numbers. Each function shouldn't exceed one line.

**PROCEDURE:**

```
import fractions
n1 = int(input("Enter n1 value:"))n2 =
int(input("Enter n2 value:"))gcd =
fractions.gcd(n1, n2) print("GCD value
is:",gcd)
def lcm(n, m):
    return n * m / gcd

print("LCM value is:",int(lcm(n1,n2)))
```

**OUTPUT:**

```
Enter n1 value: 5
Enter n2 value: 9
GCD value is: 1
LCM value is: 45
```

## EXPERIMENT 9

### MULTI-D LISTS

#### a) WRITE A PROGRAM THAT DEFINES A MATRIX AND PRINTS

#### AIM:

Write a program that defines a matrix and prints

#### PROCEDURE:

```
row=int(input("Enter No of Rows for 1st Matrix:"))
column=int(input("Enter No of column for 1nd Matrix:"))
row1=int(input("Enter No of Rows for 2st Matrix:"))
column1=int(input("Enter No of column for 2nd Matrix:"))
X = [[int(input(("Enter value for X["i","j"]"))) for j in
range(column)] for i in range(row)]
Y = [[int(input(("Enter value for Y["i","j"]"))) for j in
range(column1)] for i in range(row1)]
print("1st Matrix X:",X)
print("2st Matrix Y:",Y)
```

#### OUTPUT:

Enter No of Rows for 1st Matrix: 3

Enter No of column for 1nd Matrix: 3

Enter No of Rows for 2st Matrix: 3

Enter No of column for 2nd Matrix: 3

('Enter value for X['0, ']['0, ']:')5

('Enter value for X['0, ']['1, ']:')8

('Enter value for X['0, ']['2, ']:')2

('Enter value for X['1, ']['0, ']:')1

('Enter value for X['1, ']['1, ']:')9

('Enter value for X['1, ']['2, ']:')2

('Enter value for X['2, ']['0, ']:')3

('Enter value for X['2, ']['1, ']:')4

('Enter value for X['2, ']['2, ']:')5

('Enter value for Y['0, ']['0, ']:')6

('Enter value for Y['0, ']['1, ']:')0

('Enter value for Y['0, ']['2, ']:')1

('Enter value for Y['1, ']['0, ']:')4

('Enter value for Y['1, ']['1, ']:')2

('Enter value for Y['1, ']['2, ']:')5

('Enter value for Y['2, ']['0, ']:')6

('Enter value for Y['2, ']['1, ']:')7

('Enter value for Y['2, ']['2, ']:')8

1st Matrix X: [[5, 8, 2], [1, 9, 2], [3, 4, 5]]

2st Matrix Y: [[6, 0, 1], [4, 2, 5], [6, 7, 8]]

**b) WRITE A PROGRAM TO PERFORM ADDITION OF TWO SQUARE MATRICES**

**AIM:**

Write a program to perform addition of two square matrices

**PROCEDURE:**

```
row=int(input("Enter No of Rows for 1st Matrix:"))
column=int(input("Enter No of column for 1nd Matrix:"))
row1=int(input("Enter No of Rows for 2st Matrix:"))
column1=int(input("Enter No of column for 2nd Matrix:"))
X = [[int(input(("Enter value for X["i","j"]:"))) for j in
range(column)] for i in range(row)]
Y = [[int(input(("Enter value for Y["i","j"]:"))) for j in
range(column1)] for i in range(row1)]
print("1st Matrix X:",X)
print("2st Matrix Y:",Y)
if (row==row1 and column==column1):
    result = [[X[i][j] + Y[i][j] for j in range(len(X))] for i in
range(len(X[0]))]
    print(result)else:
    print("Adition 2 Matrix not Possible")
```

**OUTPUT:**

Enter No of Rows for 1st Matrix: 2 Enter No  
of column for 1nd Matrix: 2Enter No of Rows  
for 2st Matrix: 2 Enter No of column for 2nd  
Matrix: 2('Enter value for X[' 0, '][' 0, ']:')1  
(Enter value for X[' 0, '][' 1, ']:')2  
(Enter value for X[' 1, '][' 0, ']:')3  
(Enter value for X[' 1, '][' 1, ']:')4  
(Enter value for Y[' 0, '][' 0, ']:')5  
(Enter value for Y[' 0, '][' 1, ']:')6  
(Enter value for Y[' 1, '][' 0, ']:')7  
(Enter value for Y[' 1, '][' 1, ']:')8  
1st Matrix X: [[1, 2], [3, 4]]  
2st Matrix Y: [[5, 6], [7, 8]]  
[[6, 8], [10, 12]]

**c) WRITE A PROGRAM TO PERFORM MULTIPLICATION OF TWO SQUARE MATRICES**

**AIM:**

Write a program to perform multiplication of two square matrices

**PROCEDURE:**

```
row=int(input("Enter No of Rows for 1st Matrix:"))
column=int(input("Enter No of column for 1nd Matrix:"))
row1=int(input("Enter No of Rows for 2st Matrix:"))
column1=int(input("Enter No of column for 2nd Matrix:"))
```

```

if(column==row1):
    X = [[int(input("Enter value for X["i,"j"]")) for j in
range(column)] for i in range(row)]
    Y = [[int(input("Enter value for Y["i,"j"]")) for j in
range(column1)] for i in range(row1)]
    result = [[0 for j in range(column1)] for i in range(row)]
    print("result",result)
    print("1st Matrix X:",X)
    print("2st Matrix Y:",Y)
    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)
else:
    print("Multiplication is not possible")

```

### OUTPUT:

```

Enter No of Rows for
1st Matrix:2 Enter No of
column for 1nd Matrix:2
Enter No of Rows for2st
Matrix:2 Enter No of
column for 2ndMatrix:2
('Enter value for X['0,
']', 0, '):'4
('Enter value for X['0, ']', 1, ']:')3
('Enter value for X['1, ']', 0, ']:')5
('Enter value for X['1, ']', 1, ']:')6
('Enter value for Y['0, ']', 0, ']:')2
('Enter value for Y['0, ']', 1, ']:')1
('Enter value for Y['1, ']', 0, ']:')7
('Enter value for Y['1, ']', 1, ']:')3
result [[0, 0], [0, 0]]
1st Matrix X: [[4, 3], [5, 6]]
2st Matrix Y: [[2, 1], [7, 3]]
[29, 13]
[52, 23]

```



## EXPERIMENT 10

### GUI, GRAPHICS

#### a) WRITE A GUI FOR AN EXPRESSION CALCULATOR USINGTK

#### AIM:

Write a GUI for an Expression Calculator usingtk

#### PROCEDURE:

```
# Python program to create a simple GUI
# calculator using Tkinter

# import everything from tkinter module
from tkinter import *

# globally declare the expression variable
expression = ""

# Function to update expression
# in the text entry box
def press(num):
    # point out the global expression variable
    global expression

    # concatenation of string
    expression = expression + str(num)

# update the expression by using set method
equation.set(expression)

# Function to evaluate the final expression
def equalpress():
    # Try and except statement is used
    # for handling the errors like zero
    # division error etc.

    # Put that code inside the try block
    # which may generate the error
    try:

        global expression

        # eval function evaluate the expression
        # and str function convert the result
        # into string
        total = str(eval(expression))

        equation.set(total)

        # initialize the expression variable
        # by empty string
        expression = ""
```

```

# if error is generate then handle
# by the except block
except:
    equation.set(" error ")
    expression = ""

# Function to clear the contents
# of text entry box
def clear():
    global expression
    expression = ""
    equation.set("")

# Driver code
if __name__ == "__main__":
    # create a GUI window
    gui = Tk()

    # set the background colour of GUI window
    gui.configure(background="light green")

    # set the title of GUI window
    gui.title("Simple Calculator")

    # set the configuration of GUI window
    gui.geometry("270x150")

    # StringVar() is the variable class
    # we create an instance of this class
    equation = StringVar()

    # create the text entry box for
    # showing the expression .
    expression_field = Entry(gui, textvariable=equation)

    # grid method is used for placing
    # the widgets at respective positions
    # in table like structure .
    expression_field.grid(columnspan=4, ipadx=70)

    # create a Buttons and place at a particular
    # location inside the root window .
    # when user press the button, the command or
    # function affiliated to that button is executed .
    button1 = Button(gui, text=' 1 ', fg='black', bg='red',
                     command=lambda: press(1), height=1, width=7)
    button1.grid(row=2, column=0)

    button2 = Button(gui, text=' 2 ', fg='black', bg='red',
                     command=lambda: press(2), height=1, width=7)
    button2.grid(row=2, column=1)

    button3 = Button(gui, text=' 3 ', fg='black', bg='red',
                     command=lambda: press(3), height=1, width=7)

```

```
button3.grid(row=2, column=2)

button4 = Button(gui, text=' 4 ', fg='black', bg='red',
                  command=lambda: press(4), height=1, width=7)
button4.grid(row=3, column=0)

button5 = Button(gui, text=' 5 ', fg='black', bg='red',
                  command=lambda: press(5), height=1, width=7)
button5.grid(row=3, column=1)

button6 = Button(gui, text=' 6 ', fg='black', bg='red',
                  command=lambda: press(6), height=1, width=7)
button6.grid(row=3, column=2)

button7 = Button(gui, text=' 7 ', fg='black', bg='red',
                  command=lambda: press(7), height=1, width=7)
button7.grid(row=4, column=0)

button8 = Button(gui, text=' 8 ', fg='black', bg='red',
                  command=lambda: press(8), height=1, width=7)
button8.grid(row=4, column=1)

button9 = Button(gui, text=' 9 ', fg='black', bg='red',
                  command=lambda: press(9), height=1, width=7)
button9.grid(row=4, column=2)

button0 = Button(gui, text=' 0 ', fg='black', bg='red',
                  command=lambda: press(0), height=1, width=7)
button0.grid(row=5, column=0)

plus = Button(gui, text=' + ', fg='black', bg='red',
              command=lambda: press("+"), height=1, width=7)
plus.grid(row=2, column=3)

minus = Button(gui, text=' - ', fg='black', bg='red',
               command=lambda: press("-"), height=1, width=7)
minus.grid(row=3, column=3)

multiply = Button(gui, text=' * ', fg='black', bg='red',
                  command=lambda: press("*"), height=1, width=7)
multiply.grid(row=4, column=3)

divide = Button(gui, text=' / ', fg='black', bg='red',
                command=lambda: press("/"), height=1, width=7)
divide.grid(row=5, column=3)

equal = Button(gui, text=' = ', fg='black', bg='red',
               command=equalpress, height=1, width=7)
equal.grid(row=5, column=2)

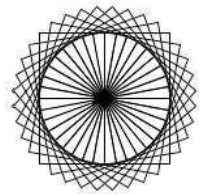
clear = Button(gui, text='Clear', fg='black', bg='red',
               command=clear, height=1, width=7)
clear.grid(row=5, column=1)
```

```

Decimal= Button(gui, text='.', fg='black', bg='red',
                command=lambda: press('.'), height=1, width=7)
Decimal.grid(row=6, column=0)
# start the GUI
gui.mainloop()

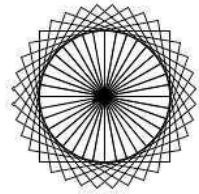
```

b) **WRITE A PROGRAM TO IMPLEMENT THE FOLLOWING FIGURES USING TURTLE**



**AIM:**

Write a program to implement the following figures using turtle



**PROCEDURE:**

```

i)
import turtle
loadWindow = turtle.Screen()
turtle.speed(2)

for i in range(100):
    turtle.circle(5*i)
    turtle.circle(-5*i)
    turtle.left(i)

turtle.exitonclick()

```

**OUTPUT:**



```
ii)
import turtle
t = turtle.Turtle()
# radius of the circle
r = 10
# Loop for printing concentric circles
for i in range(50):
    t.circle(r * i)
    t.up()
    t.sety((r * i)*(-1))
    t.down()
```

**OUTPUT:**

