**Introduction to Computer Science Engineering**

**B.Tech. 1st Semester**



**Name :Bhaskar Shirol**

**Reg Number :24ETCS002239**

**Programme :B-TECH**

**Department :Computer Science Engineering**

**Faculty of Engineering & Technology**

**Ramaiah University of Applied Sciences**

**Ramaiah University of Applied Sciences**

Private University Established in Karnataka State by Act No. 15 of 2013

|  |  |
| --- | --- |
| Faculty | Engineering & Technology |
| Programme | B. Tech. in Computer Science and Engineering |
| Year/Semester | 1st Year /1st Semester |
| Name of the Laboratory | Introduction to Computer Science Engineering |
| Laboratory Code | CSD103B |

List of Experiments

1. Introduction to Python programming environment
2. Variables, data types, operators and expressions
3. Input output operations
4. Logic operations and decision making
5. Loop statements
6. Character and string operations
7. String and Dictionary operations
8. Functions
9. File handling

# Index Sheet

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No.** |  |  |  |  | | **Lab Experiment** | **Lab document** | **Viva** |
| 1 | Introduction to Python programming environment | Lab document  10% of Lab CE |  |
| 2 | Variables, data types, operators and expressions | Lab document  10% of Lab CE |  |
| 3 | Input output operations | Lab document  10% of Lab CE |  |
| 4 | Logic operations and decision making | Lab document  10% of Lab CE |  |
| 5 | Loop statements | Lab document  10% of Lab CE |  |
| 6 | Character and string operations | Lab document  10% of Lab CE |  |
| 7 | String and Dictionary operations | Lab document  10% of Lab CE |  |
| 8 | Functions | Lab document  10% of Lab CE |  |
| 9 | File handling | Lab document  10% of Lab CE |  |
| 10 | Lab Internal Test | 10% of Lab CE | |

# Laboratory 1

Title of the Laboratory Exercise: Introduction to Python programming environment

1. Introduction and Purpose of Experiment

Python is a high level, interactive, interpreted, object oriented scripting language. Wing IDE is an open source integrated development environment for programming in the Python language. In this laboratory exercise, students get familiar with the program development tool such as Wing IDE to edit and run Python programs using a set of simple exercises.

1. Aim and Objectives

Aim

* To familiarise IDE and simple Python programs

Objectives

At the end of this lab, the student will be able to

* Explain the features and use of IDE to develop Python programs
* Edit and execute simple Python programs

1. Experimental Procedure

Students are given a set of Python programs. Edit and execute Python programs using

* Python command line
* Using IDE
  + Interactive mode
  + Script mode

1. Problem solving logic
2. Using command line format finding area of circle
3. Using interactive mode

1) swapping 2 numbers

2)finding perimeter of rectangle

1. Algorithm
2. Algorithm: area(var r:integer)

Begin

Area=3.14\*r\*r

Write(Area)

1. 1) Algorithm: Swap( var a,b:integer)

Begin

Var c:integer

c=a

a=b

b=c

write(a.b)

end

2) Algorithm: perimeter (var l,b:integer)

begin

p=2\*(l+b)

write(p)

end

1. Implementation :
2. Area of a circle

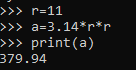


Figure 1

Figure 1 shows the python program and output for area of circle when radius is given

1. 1) swapping two numbers

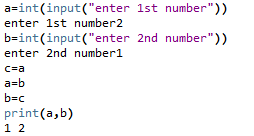


Figure 2

Figure 2 shows the python program and output for swapping 2 numbers:

2) perimeter of a rectangle

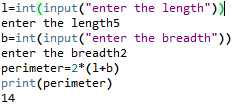


Figure3

Figure 3 shows the python program and output for finding perimeter of rectangle

1. Results:
2. Figure 1 (python command line)
3. 1) figure 2 (interactive mode)

2) figure 3 (interactive mode)

1. Conclusions

i)Limitations of Experiments

Input of the should be positive numbers because length cannot be in negative.

ii) Learning happened

using python command line and script and interactive modes

iii)Future Recommendations

No future recommendation

# Laboratory 2

Title of the Laboratory Exercise: Variables, data types, operators and expressions

1. Introduction and Purpose of Experiment

Variables are the basic data objects that are manipulated in a program. Every value in Python has a datatype. Operators specify what is to be done to them. Expressions combine variables and constants to produce new values. These building blocks are the topics of this Lab. By solving the given programming problems, the students will be able to apply the concepts of variables, data types, operators and expressions.

1. Aim and Objectives

Aim

* To develop programs using variables of basic data types and compute simple expressions involving operators

Objectives

At the end of this lab, the student will be able to

* Use variables of the basic data types
* Apply various operators in expressions
* Create Python programs to solve simple numeric problems

1. Experimental Procedure
2. Analyse the problem statement
3. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
4. Implement the algorithm in Python language
5. Execute the Python program
6. Test the implemented program
7. Document the Results
8. Analyse and discuss the outcomes of the experiment

1. Questions
2. Write a Python program to exchange the values of two variables using a temporary variable.
3. Write a Python program to find the area and perimeter of a rectangle.
4. Problem solving logic
5. Entering 2 numbers and swapping them one by one with the help of third temporary variable
6. Calculating perimeter and area of rectangle using the formula p=2(l+b) and a=l\*b respectively
7. Algorithm
8. Algorithm Swap( var a,b:integer)

Begin

Var c:integer

c=a

a=b

b=c

write(a.b)

end

1. Algorithm dimensions(var l,b:integer)

begin

a=l\*b

p=2\*(l+b)

write(a,p)

end

1. Implementation :
2. Swapping 2 numbers

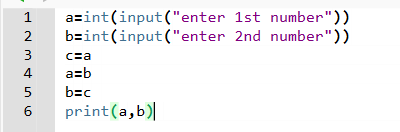


Figure 1

Figure 1: shows the python program to receive input for a and b in interger data type and swapping them with the help of temporary variable c.

1. Perimeter of rectangle

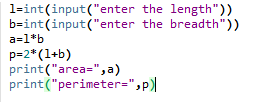


Figure 2

Figure 2:shows the python program to receive input for length and breadth and calculating the perimeter and area of rectangle

1. Results :
2. Result of program a



Figure 3

Figure 3: shows the output for program a (figure 1) and the numbers are swapped

1. Result of program b



Figure 4

Figure 4 shows the output for program b(figure 2) therefore the area and perimeter of rectangle are calculated

1. Conclusions
2. Limitations of Experiments

For the program b the input for length and breadth should be positive integer as length or breadth cannot be negative

1. Learning happened
2. Using temporary variables
3. How to calculate area and perimeter of rectangle

iii)Future Recommendations

No future recommendation.

# Laboratory 3

Title of the Laboratory Exercise: Input Output operations

1. Introduction and Purpose of Experiment

Python provides the Input-Output facility to interact with the user. Some of the functions such as input() and print() are widely used for standard input and output operations respectively. By solving the problems, students will be able to develop programs to read input from screen and display output to the screen.

1. Aim and Objectives

Aim

* To develop programs involving input output operations

Objectives

At the end of this lab, the student will be able to

* Apply input and output operations

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions
   1. Write a Python program to accept three numbers from the user (in a single line separated by spaces) and display their average.
   2. Write a Python program to enter an integer number and perform the following operations (use built in methods) on the integer
      * Square, Cube, and Factorial
3. Problem solving logic
4. Accepting 3 variables in one line and calculating their average using formula (a+b+c)/3
5. Using python functions to square and cube a number and find its factorial
6. Algorithm
7. algorithm: average

begin

read(a,b,c)

a=int(a)

b=int(b)

c=int(c)

avg=(a+b+c)/3

write(a,b,c)

end

1. algorithm: math(var n:integer)

import math

begin

s=math.pow(n,2)

c=math.pow(n,3)

f=math.factorial(n)

write(s,c,f)

end

1. Implementation :
2. Printing average of 3 variables

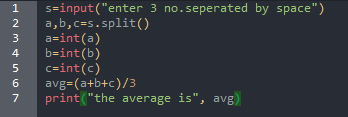


Figure 1

Figure 1: shows a python program to receive input of 3 numbers in a single line which are separated by spaces and converting them into integers and displaying their average

1. Finding square, cube and factorial of a number

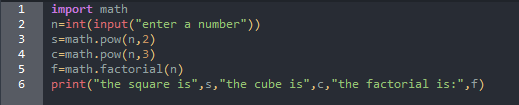


Figure 2

Figure 2: shows a python program to find square, cube and factorial of a number using math library

1. Results :
2. Result for program a



Figure 3

Figure 3: shows the output for receiving 3 numbers separated by spaces and displaying their average

1. Results for program b



Figure 4

Figure 4: shows the square, cube and factorial of the given input (5)



Figure 5

Figure 5 :shows the output when the input is “1”



Figure 6

Figure 6: shows the output when the input is “0”

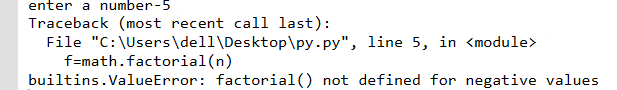


Figure 7

Figure 7: shows the output when the input is negative which gives the error as factorial doesn’t exist for negative numbers

1. Conclusions
2. Limitations of Experiments

In the program b the input should be positive numbers only as for negative numbers factorial doesn’t exist but square and cube is possible

1. Learning happened
2. Using split function and converting string into integer
3. Using math library and calculating square, cube and factorial of a number

iii)Future Recommendations

# Laboratory 4

Title of the Laboratory Exercise: Logic operations and decision making

1. Introduction and Purpose of Experiment

Python provides number of control flow instructions/statements to control the flow of program execution conditionally. By solving the problems, students will be able to apply conditional control statements to control the program execution.

1. Aim and Objectives

Aim

* To develop programs involving branching using appropriate control statements in   
  Python

Objectives

At the end of this lab, the student will be able to

* Apply conditional control statements such as *if-else* and nested *if-else* to express decisions

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions
   1. Write a Python program to check whether the given number is zero, positive or negative.
   2. Write a Python program to display the roots of a quadratic equation.
3. Problem solving logic
4. Using greater than or less than function checking if a number is positive or negative
5. Using for for roots of quadratic equation finding roots of quadratic equations
6. Algorithm
7. Algorithm: check( var a:integer)

Begin

If a==0:

Write(“a is equal to zero”)

Elif a>0:

Write(“a is positive”)

Else :

Write(“a is negative”)

end

1. Algorithm: roots(var a,b,c:integer)

Begin

D=b^2-4\*a\*c

If d==0:

Write(“r1=-b/2\*a)

Elif d>0:

Write(“r2=(-b)+math.sqrt(d)/2\*a”,r3=(-b)-maths.sqrt(d)/2\*a)

Else:

Write(“r4=(-b/2a)+i\*math.sqrt(abs(d))/2a”)

Write(“r5=(-b/2a)-i\*math.sqrt(abs(d))/2a”)

7) Implementation :

a) finding number is positive or negative

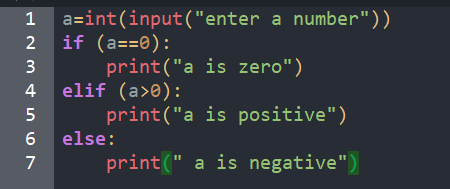


Figure 1

Figure 1: shows the python program to check weather the number is positive, negative or equal to zero

b)Finding square root of quadratic equation

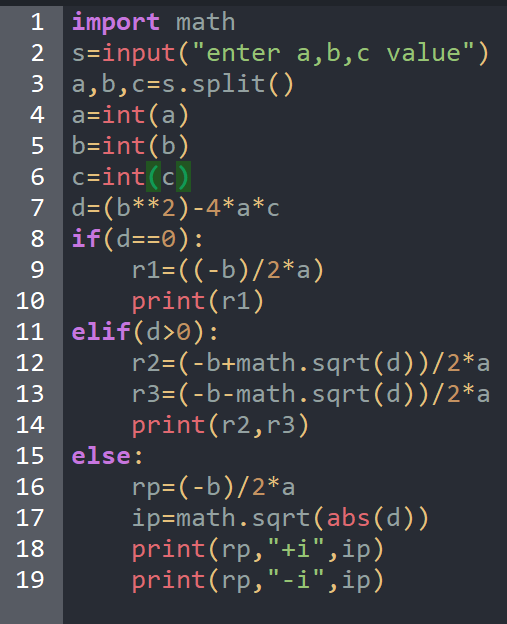


Figure 2

Figure 2: shows the python program to find the roots of a quadratic equation

1. Results :
2. Result for program a



Figure 3

Figure 3: shows the output for figure 1when the input is positive number



Figure 4

Figure 4 :shows the output for figure 1 when the input is 0



Figure 5

Figure 5: shows the output for figure 1 when the input is negative number

1. Results for program b



Figure 6

Figure 6: shows the result for figure 2 when b^2>4\*a\*c



Figure 7

Figure 7: shows the result for figure 2 when b^2<4\*a\*c



Figure 8

Figure 8: shows the result for figure 2 when b^2=4\*a\*c

1. Conclusions
2. Limitations of Experiments

No limitation

ii) Learning happened

1. Usage of nested if or if, elif and else usage
2. Using abs function
3. Using python libraries

iii)Future Recommendations

No future recommendation.

# Laboratory 5

Title of the Laboratory Exercise: Loop statements

1. Introduction and Purpose of Experiment

Loop statements are used to repeat a statement or set of statements multiple times. By solving the problems students will be able to apply iterative control statements to control the program execution.

1. Aim and Objectives

Aim

* To develop programs involving loops using appropriate control statements in Python

Objectives

At the end of this lab, the student will be able to

* Apply loop control statements such as *for* and *while* to repeat a block of code

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions
   1. Write a Python program to accept an integer number and find sum of digits in the number.
   2. Write a program to perform addition and subtraction of two matrices
      1. Without using *numpy* module
      2. Using *numpy* module
3. Problem solving logic
4. Using while loop to divide the given number by 10 to get its last digit and then add all to find their sum
5. Algorithm
   1. Algorithm . sign(var a: integer)

Step 1 : Start

Step 2 : Read user input

Step 3 : Create a variable to hold the result , ‘sum’

Step 4 : Convert the input number into string

Step 5 : Write a looping statement to extract each digit of the number

Step 6 : Convert the extracted digits into integer number and add each digit to the variable, ‘sum’

Step 7 : Print ‘sum’

Step 8 : Stop



i) WITHOUT USING NUMPY MODULE

Step 1 : Start

Step 2 : Define two matrices M1 and M2

Step 3 : Print the elements of M1 and M2 using nested for loop

Step 4 : Define an empty matrix for storing the result of sum of matrix , ‘sum’ and ‘diff’

Step 5 : Add and subtract the corresponding elements of M1 and M2 using nested for loop and store the result in ‘sum’ and ‘diff’ respectively .

Step 6 : Using nested loop print the results

Step 7 : Stop

ii) USING NUMPY MODULE

Step 1 : Start

Step 2 : Import the ‘numpy’ module into the program

Step 3 : Declare a variable for matrix M1 and give values

Step 4 : Declare a variable for matrix M2 and give values

Step 5 : Call the add() and subtract() functions separately and perform the operation

Step 6 : Print the results

Step 7 : Stop

1. Implementation :
2. Fum of digits of a number

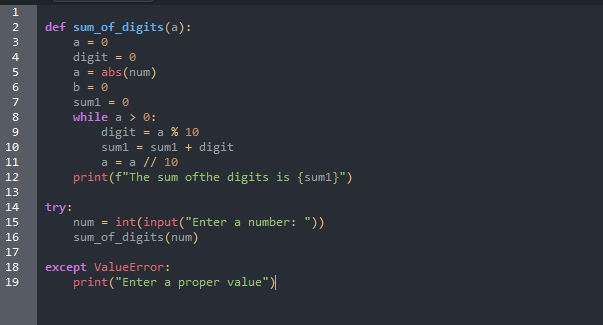


Figure 1 shows the python program to accept an integer and find the sum of its digits using a while loop

1. b) To perform addition and subtraction of two matrices

I) WITHOUT USING NUMPY MODULE

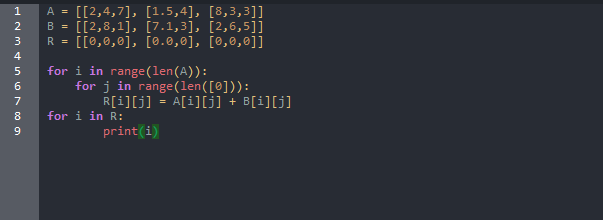


Figure 2

Figure 2 : Python code to perform addition and subtraction of two matrices without using numpy module

II) Using numpy module

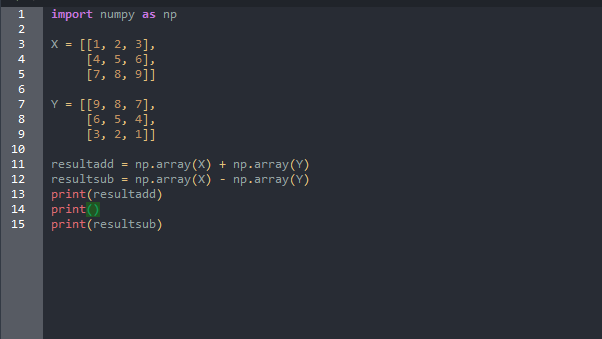


Figure 3

Figure 3 : : Python code to perform addition and subtraction of two matrices using numpy module

1. Results :
2. Result for program a

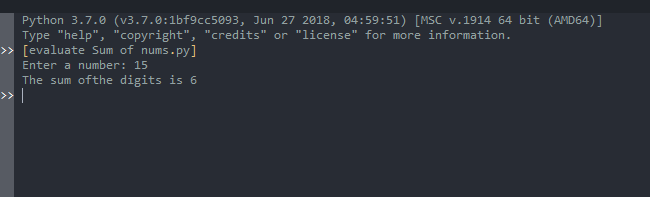


Figure 4

Figure 4 shows the output for figure 1 when the input is 28.

1. i) Without numpy module

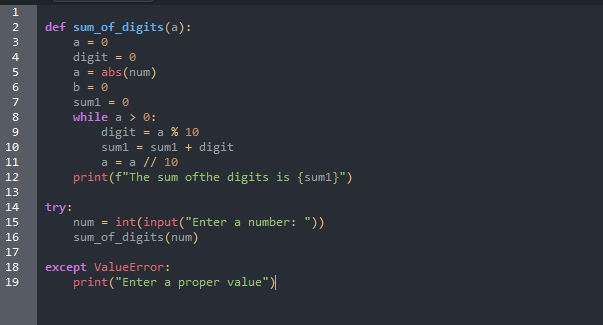


Figure 1 shows the python program to accept an integer and find the sum of its digits using a while loop

1. b) To perform addition and subtraction of two matrices

I) WITHOUT USING NUMPY MODULE

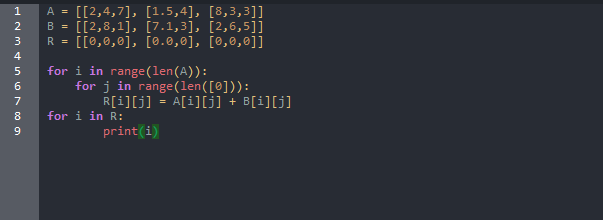


Figure 2

Figure 2 : Python code to perform addition and subtraction of two matrices without using numpy module

II) Using numpy module

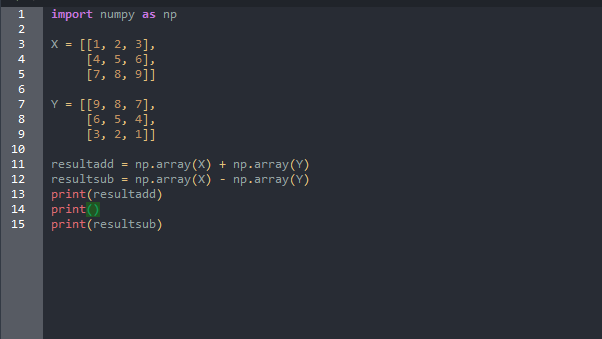


Figure 3

Figure 3 : : Python code to perform addition and subtraction of two matrices using numpy module

1. Results :
2. Result for program a

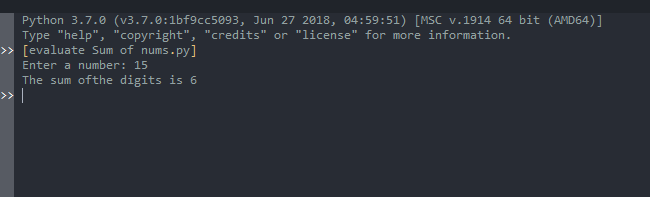


Figure 4

Figure 4 shows the output for figure 1 when the input is 28.

1. i) Without numpy module

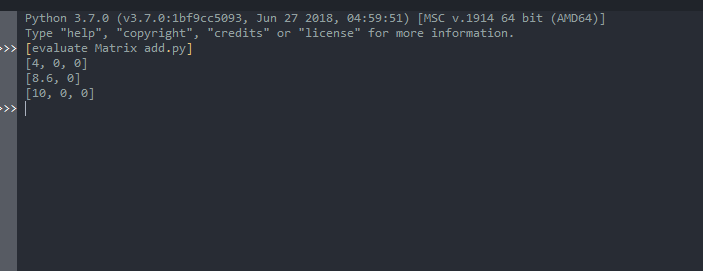


Figure 5

Figure 5 : : Output obtained by the python interpreter of Wing IDE for addition and subtraction of two matrices without numpy module

II) Using numpy module

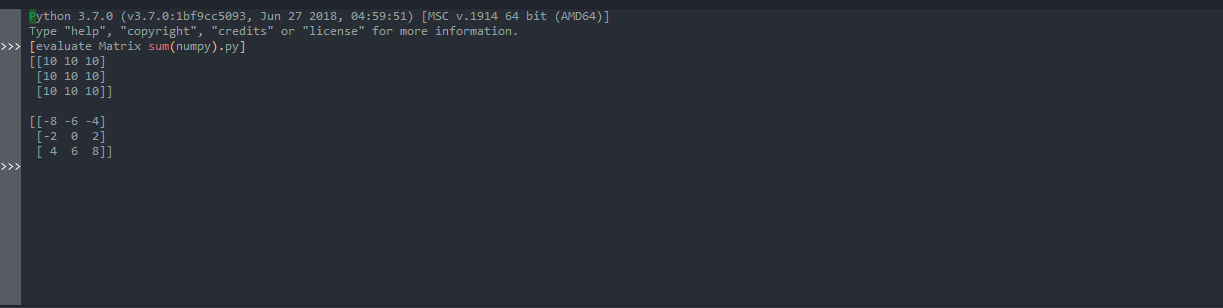


Figure 6

Figure 6 : Output obtained by the python interpreter of Wing IDE for addition and subtraction of two matrices using numpy module .

1. Conclusions
2. Limitations of Experiments

No limitation.

1. Learning happened

Finding sum of digits using loop

Finding sum of matrix with and without using numpy

iii)Future Recommendations

No future recommendation

# Laboratory 6

Title of the Laboratory Exercise: List and String operations

1. Introduction and Purpose of Experiment

A string is a sequence of characters. Strings can be manipulated using a number of built in functions. By solving this, students will be able to manipulate string data types.

1. Aim and Objectives

Aim

* To develop programs for manipulating characters and strings

Objectives

At the end of this lab, the student will be able to

* Create programs to perform string operations
* Create programs to manipulate strings using built in functions

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions

All programs to be coded without using any built-in functions also use program defined input.

1. Write a Python program to sum all the items in a list.
2. Write a Python program to get the largest number from a list.
3. Write a Python program to calculate the length of a string
4. Write a Python program to count the number of characters (character frequency) in a string.   
   Sample String : google.com'  
   Expected Result : ['g': 2, 'o': 3, 'l': 1, 'e': 1, '.': 1, 'c': 1, 'm': 1]
5. Problem solving logic
6. Traversing through and adding all the elements in the list to find the sum
7. Traversing through the list and comparing the elements with each other to find the largest element
8. Converting the string into list and using count to find the length of the list
9. ..
10. Algorithm
11. Algorithm: sum(var a:list)

Begin

Sum=0

For I in range(len(a)):

S+=a[i]

Write(sum)

End

1. Algorithm: largest(var a:list)

Begin

L=a[0]

For I in range(len(a)):

If a[i]>l:

L==a[i]

Write(L)

End

1. Algorithm: length(var s:str)

Begin

l=0

a=list(s)

for I in range(len(a)):

l+=1

write(l)

end

1. Algorithm: frequency(var l:str)

Begin

ch=[]

co=[]

for i in l:

if i in ch:

a=ch.index(i)

co[a]=c[a]+1

else:

ch.append(i)

co.append(1)

write(ch)

write(co)

end

1. Implementation :
2. To find sun of elements of list

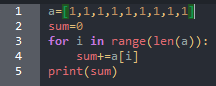


Figure 2

Figure 1: shows the python program to fins the sum of elements in the list when list a is pre defined

1. To find largest element in list

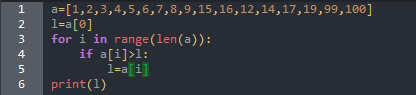


Figure 2

Figure 2 : shows the python program to find the largest element in the list

1. To find length od=f string

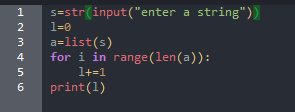


Figure 3

Figure 3 : shows the python program to find the length of the string

1. To count frequency of a character in the string

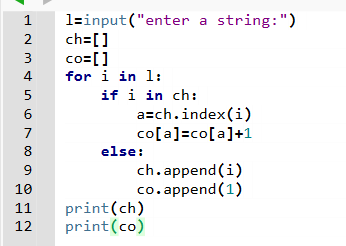


Figure 4

1. Figure 4 : To count frequency of a character in the string
2. Results :
3. Output for program a



Figure 5

Figure 5 : shows the output for figure 1

1. Output for program b



Figure 6

Figure 6 : shows the output for program 2

1. Output for program c



Figure 7

Figure 7 : shows the output for figure 3

1. Output for program d



Figure 8

Figure 8 : show the result for figure 4

1. Conclusions
2. Limitations of Experiments

No limitation

ii) Learning happened

1. Using of for loop
2. Using of range function in for loop

iii)Future Recommendations

No future recommendation

# Laboratory 7

Title of the Laboratory Exercise: String and Dictionary operations

1. Introduction and Purpose of Experiment

A string is a sequence of characters. Strings can be manipulated using a number of built in functions. By solving this, students will be able to manipulate string data types.

1. Aim and Objectives

Aim

* To develop programs for manipulating characters and strings

Objectives

At the end of this lab, the student will be able to

* Create programs to perform string operations
* Create programs to manipulate strings using built in functions

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions
   1. Write a Python program to find duplicate characters in the given string and store them in a list.
   2. Write a Python program to count the numbers of occurrences of characters in the given string and store them in a dictionary.
3. Problem solving logic
4. Using for loop and inserting elements which appear more than once in the empty list and printing the list
5. Using for loop counting the occurrence of one character and adding them in the dictionary
6. Algorithm
7. Algorithm: similar(var s:str)

Begin

L=[]

For i in range(len(s)):

If s.count(s[i])>1:

l.append(s[i])

write(l)

end

1. Algorithm: counting( var s:str)

Begin

D={}

For i in range(len(s)):

b=s.count(s[i])

d[s[i]]=b

write(d)

end

1. Implementation:
2. Finding similar characters in a string

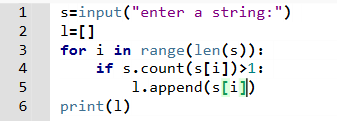


Figure 1

Figure 1 : shows the python program to find duplicate characters in a string and store them in a list

1. Counting occurrence of characters in a string

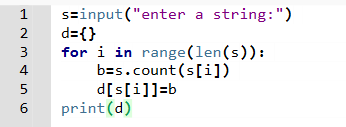


Figure 2

Figure 2 : shows the python program for counting how many times a character occurs in the string and displaying them in a dictionary

1. Results:
2. Result for program a



Figure 3

Figure 3 : shows the output for figure 1 where the string entered is “watermelon” and e gets repeated twice which is then stored in a list

1. Result for program b



Figure 4

Figure 4 : shows the output for figure 2 where the string entered is “dictionary” and it counts the occurrence of each characters and displays in a dictionary

1. Conclusions
2. Limitations of Experiments

No limitation

ii) Learning happened

1. Using for loop and range function to traverse the list
2. Appending items in the list and dictionary

iii)Future Recommendations

No future recommendation

# Laboratory 8

Title of the Laboratory Exercise: Functions

1. Introduction and Purpose of Experiment

A function serves as an abstraction mechanism to view many things as one thing. The function definition specifies the name of a new function and the sequence of statements that execute when the function is called. Once a function is defined, it can be called as many times as required. By solving these problems, students will be able to create user defined functions.

1. Aim and Objectives

Aim

* To develop programs using user defined functions

Objectives

At the end of this lab, the student will be able to

* Apply user defined functions with proper definition

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions
   1. Write a Python program to find factorial of a number using recursion.
   2. Write a Python function to reverse a string (without using slicing or built in method). Use the returned result to check whether the given string is palindrome or not.
3. Problem solving logic
4. Using for loop and recursion finding factorial of a number
5. Using for loop to reverse the string and compare the given and reversed string to find if string is palindrome or not
6. Algorithm
7. Algorithm: (var n:int)

Begin

def factorial(n):

f=1

if n==0:

write(“factorial is 1”)

elif n<0:

write (“factorial doesn’t exist”)

else:

for I in range(1,n+1)

f=f\*i

write(f)

end

1. Algorithm: (var s:string)

Begin

def palindrome(s):

s1=””

for i in range(len(s)-1,-1,-1):

s1=s1+s[i]

if s==s1:

write(“s is palindrome”)

else:

write(“s is not palindrome”)

end

1. Implementation :
2. Program for finding factorial of a number

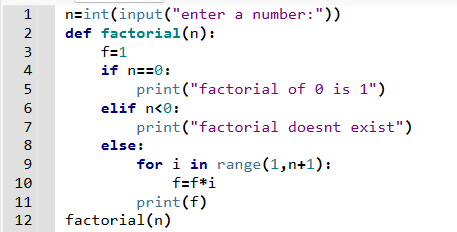


Figure 1

Figure 1: shows the python program for finding factorial of a number using recursion

1. Program for identifying if a string is palindrome or not

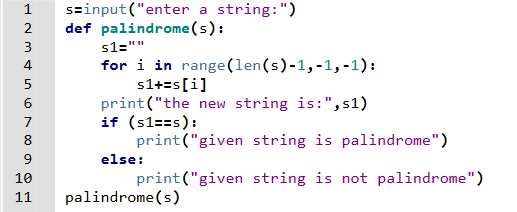


Figure 2

Figure 2 : shows the python program for reversing a string to find if the given string is palindrome or not

1. Results :
2. Output for program a



Figure 3

Figure 3 : shows the output for figure 1 when number is greater than 0



Figure 4

Figure 4 : shows the output for figure 1 when the input is 0



Figure 5

Figure 5 : shows the output for figure 1 when the input is less than 0

1. Output for program b



Figure 6

Figure 6 : shows the output for figure 2 when the input string is “Malayalam” (which is a palindrome string)



Figure 7

Figure 7 : also shows the output for figure 2 when the input is “aman” (which is not a palindrome string”)

1. Conclusions
2. Limitations of Experiments

No limitation

ii) Learning happened

1. Defining function and calling function
2. Using for loop to reverse string

iii)Future Recommendations

No future rrecommendation

# Laboratory 9

Title of the Laboratory Exercise: File handling

1. Introduction and Purpose of Experiment

A text file is a sequence of characters stored on a permanent medium such as hard drive, flash memory, and CD-ROM. In Python, instead of printing results to screen using print, results can be written to a file. A file object is used to open a connection to a text file for input or output. By solving these problems, students will be able to perform file handling functions.

1. Aim and Objectives

Aim

* To develop programs using file handling functions

Objectives

At the end of this lab, the student will be able to

* Apply file handling functions for solving problems

1. Experimental Procedure
   1. Analyse the problem statement
   2. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code
   3. Implement the algorithm in Python language
   4. Execute the Python program
   5. Test the implemented program
   6. Document the Results
   7. Analyse and discuss the outcomes of the experiment
2. Questions
   1. Write a python program to display first ‘n’ lines in a text file.
   2. Write a python program to write ‘n’ random numbers in a file.
   3. Write a python program to compute the number of characters, words and lines in a text file.
3. Problem solving logic
4. To open and read files and print the n lines
5. Using random module to write n integers in a text file
6. Using for loop to traverse the lines words to find their count and print them

6.Algorithm

1. Algorithm: (var n:int)

Begin

f=open(‘file\_name’,’r’)

l=f.readlines()

for i in l:

write(i)

f.close()

end

1. Algorithm: numbers( var n:int)

Import random

Begin

f=open(‘file\_name’,’w’)

for i in range(n):

y=random.randint(0,10)

s=str(y)

f.writelines(s)

f.close()

g=open(‘file\_name’,’r’)

l=g.readlines

for i in l:

write(i)

g.close()

end

1. Algorithm: counting(var
2. Implementation :
3. To read and display n lines of a text file

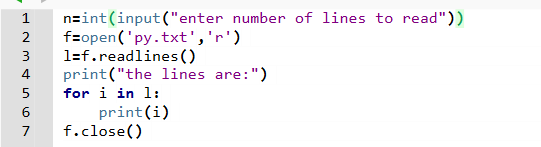


Figure 1

Figure 1: shows the python program for opening a files and reading n lines in the text file

1. To write random integers/numbers in a text file

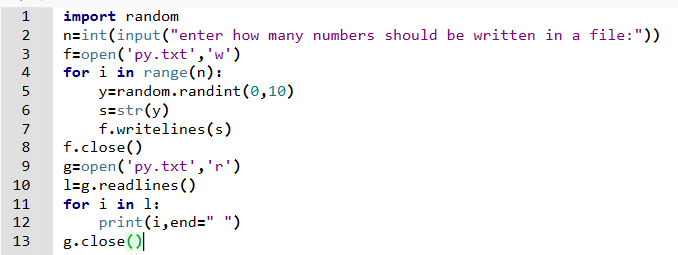


Figure 2

Figure 2 shows the python program to write n numbers in a text file

1. To count number of lines, words and characters

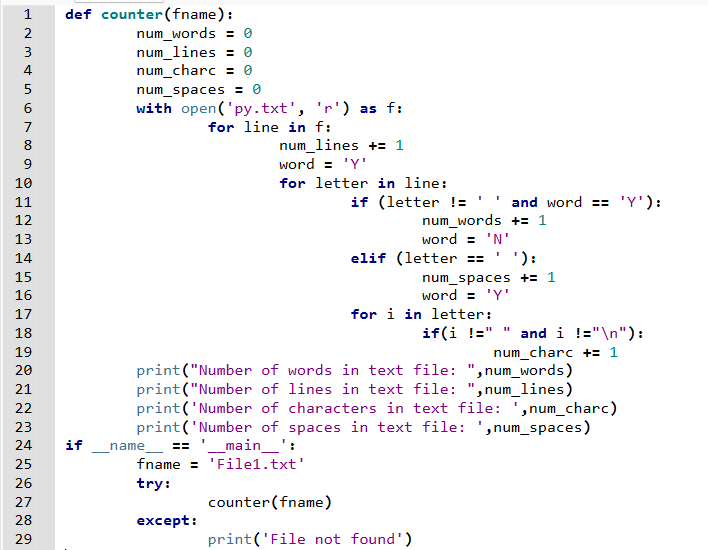


Figure 3

Figure 3 : shows the python program to calculate the number of words, lines and characters in a text file

1. Results :
2. Result for program a



Figure 4

Figure 4 shows the result for figure 1

1. Result for program b



Figure 5

Figure 5 shows the result for figure 2

1. Result for program c

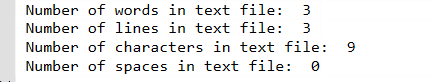


Figure 6

Figure 6 shows the output for figure 3

9. Conclusions

i) Limitations of Experiments

No limitation

1. Learning happened

1 How open and close files

1. How to write and read files
2. And how to traverse the files

iii)Future Recommendations

No future recommendation