PRATHYUSHA

ENGINEERING COLLEGE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

GE3171 PROBLEM SOLVING AND PYTHON

PROGRAMMING LABORATORY

List of experiments:

1. Identification and solving of simple real life or scientific or technical problems, and developing flow charts for the same. (Electricity Billing, Retail shop billing, Sin series, weight of a motorbike, Weight of a steel bar, compute Electrical Current in Three Phase AC Circuit, etc.)
2. Python programming using simple statements and expressions (exchange the values of two variables, circulate the values of n variables, distance between two points).
3. Scientific problems using Conditionals and Iterative loops. (Number series, Number Patterns, pyramid pattern) 4. Implementing real-time/technical applications using Lists, Tuples. (Items present in a library/Components of a car/ Materials required for construction of a building –operations of list & tuples)
5. Implementing real-time/technical applications using Sets, Dictionaries. (Language, components of an automobile, Elements of a civil structure, etc.- operations of Sets & Dictionaries)
6. Implementing programs using Functions. (Factorial, largest number in a list, area of shape)
7. Implementing programs using Strings. (reverse, palindrome, character count, replacing characters)
8. Implementing programs using written modules and Python Standard Libraries (pandas, numpy. Matplotlib, scipy)
9. Implementing real-time/technical applications using File handling. (copy from one file to another, word count, longest word)
10. Implementing real-time/technical applications using Exception handling. (divide by zero error, voter's age validity, student mark range validation)
11. Exploring Pygame tool.
12. Developing a game activity using Pygame like bouncing ball, car race etc.

Exp No : 1) a. Flowchart for Electricity Bill Calculation

Aim :

To draw the flowchart for Electricity bill calculation with the following rates for its customers: -

No. of unit consumed	Charges/unit (RM)
1-200	2.50
201 - 500	3.50
over 501	5.00

Draw a flowchart to calculated the net amount of the bill for each consumer and print it.

Algorithm :

Step 1 : Start

Step 2 : Read the unit values

Step 3 : Check 1<=unit<=200

Then calculate total_charges=unit*2.50

Step 4 : Check 201<=unit<=500

Then calculate total_charges=unit*3.50

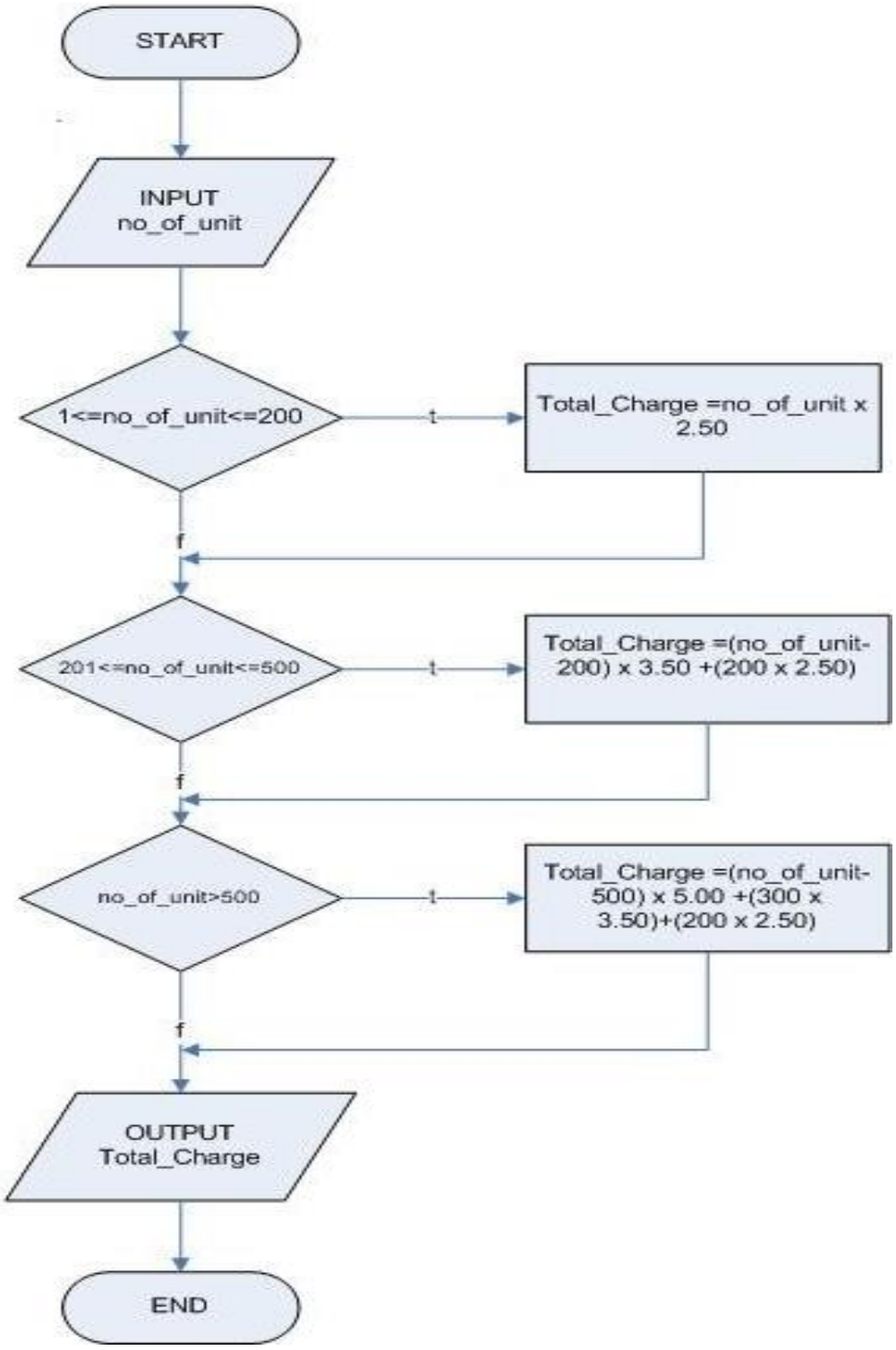
Step 5 : Check unit>500

Then calculate total_charges=unit*5.0

Step 6 : print total_charges

Step 7 : Stop

Flowchart:


START
INPUT
no_of_unit
1<=no_of_unit<=200
t
Total_Charge =no_of_unit x 2.50
f
201<=no_of_unit<=500
t
Total_Charge =(no_of_unit-200) x 3.50 +(200 x 2.50)
f
no_of_unit>500
t
Total_Charge =(no_of_unit-500) x 5.00 +(300 x 3.50)+(200 x 2.50)
f
OUTPUT
Total_Charge
END

Exp No : 1) b. Flowchart for calculation of gross salary of an employee

Aim :

To draw a flowchart for calculating the gross salary of an employee

Algorithm :

Step 1: Start

Step 2: Read basic salary of an employee

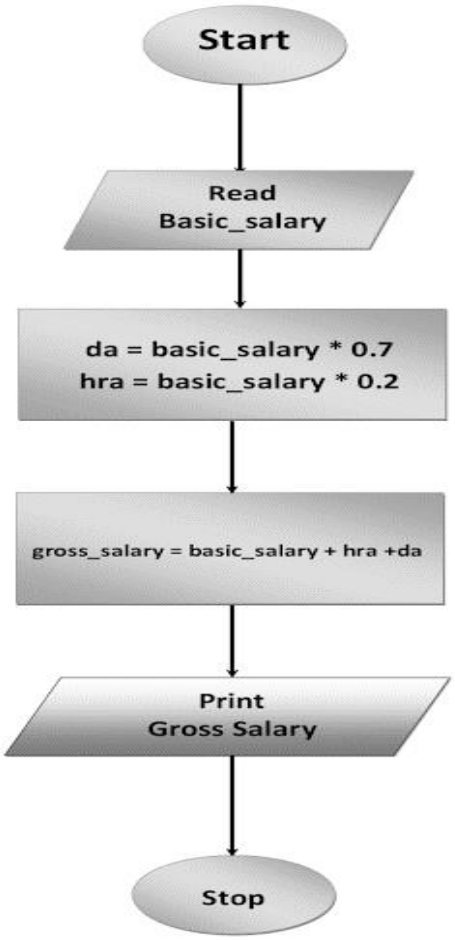
Step 3: Use the formula hra = basic_salary * 0.2 and da = basic_salary *0.7.

Step 4: Calculate gross salary by using formula gross_salary = basic_salary + hra + da.

Step 6: Print gross salary.

Step 7: End

Flowchart:



Exp No : 1) c. Flowchart for Movie ticket booking

Aim :

To draw a flowchart for booking a movie ticket.

Algorithm :

Step 1: Start

Step 2: Read and select the show timings

Step 3: Check if tickets==available

Then select the type of a movie

else:

goto Step 3

Step 4: Select a movie type

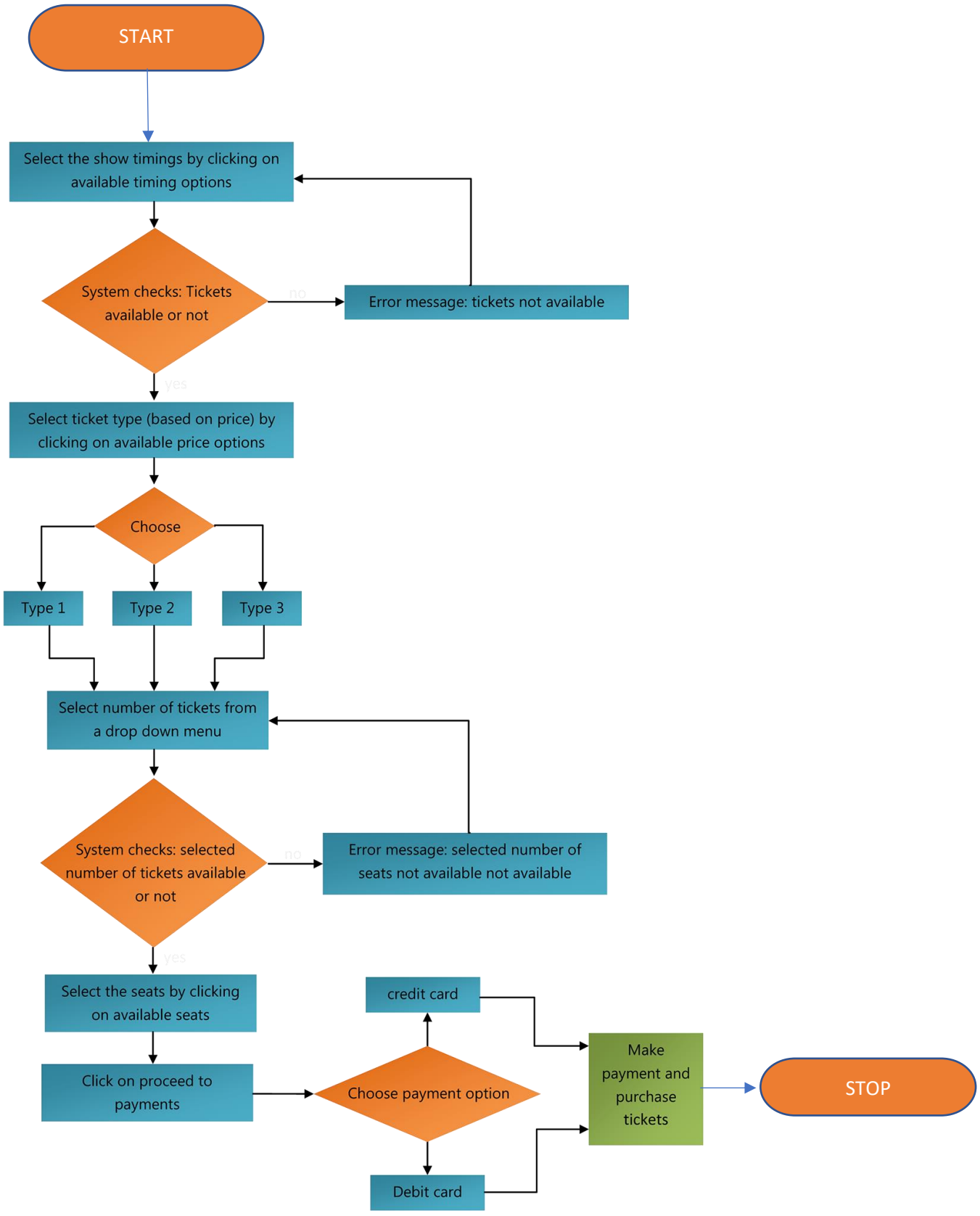
Step 5: Select the seats

Step 6: Choose a payment option

Step 7: Make payment and get the tickets

Step 8: Stop

Flowchart:


START
Select the show timings by clicking on available timing options
System checks: Tickets available or not
no
Error message: tickets not available
yes
Select ticket type (based on price) by clicking on available price options
Choose
Type 1
Type 2
Type 3
Select number of tickets from a drop down menu
System checks: selected number of tickets available or not
no
Error message: selected number of seats not available not available
yes
Select the seats by clicking on available seats
Click on proceed to payments
Choose payment option
credit card
Debit card
Make payment and purchase tickets
STOP

Exp No : 1) d. Flowchart for Student grade printing

Aim :

To draw a flowchart for student grade printing.

Algorithm :

Step 1: Start

Step 2: Read score

Step 3: if score > 90:

print grade is “O”

Step 4: if score>80:

Print grade is “A+”

Step 5: if score>70:

Print grade is “A”

Step 6: if score>60:

Print grade is “B+”

Step 7: if score>50:

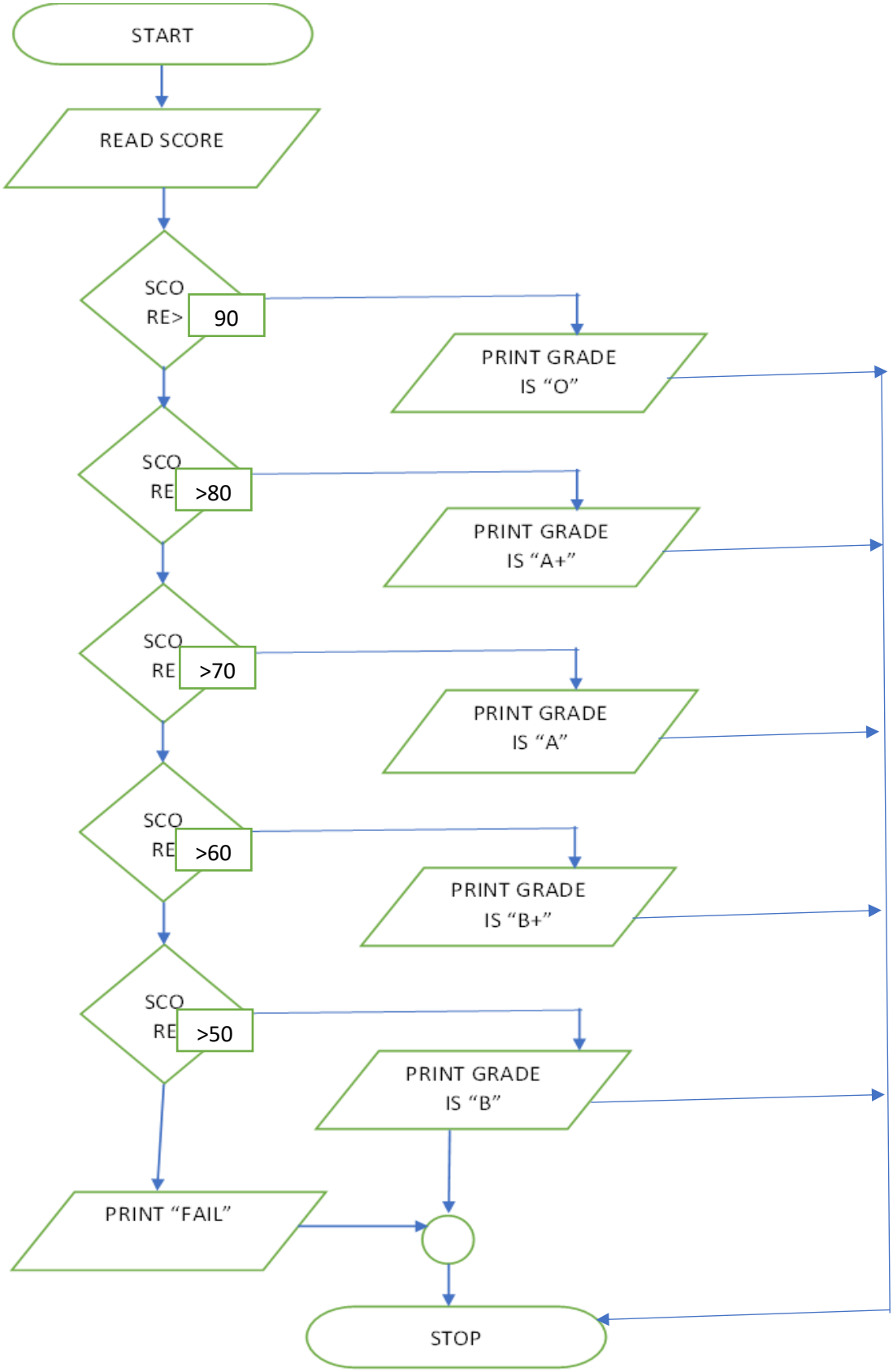
Print grade is “B”

Step 8: if score <50:

Print “Fail”

Step 9: Stop

Flowchart:


START
READ SCORE
SCO
RE>
90
PRINT GRADE
IS "O"
SCO
RE
>80
PRINT GRADE
IS "A+"
SCO
RE
>70
PRINT GRADE
IS "A"
SCO
RE
>60
PRINT GRADE
IS "B+"
SCO
RE
>50
PRINT GRADE
IS "B"
PRINT "FAIL"
STOP

Exp No : 1) e. Flowchart for college admission

Aim :

To draw a flowchart for college admission

Algorithm :

Step 1: Start

Step 2: Get enquiry form

Step 3: Check eligibility

Step 4: Upload required documents

Step 5: Check if eligible: then goto Step 6 else goto Step 4

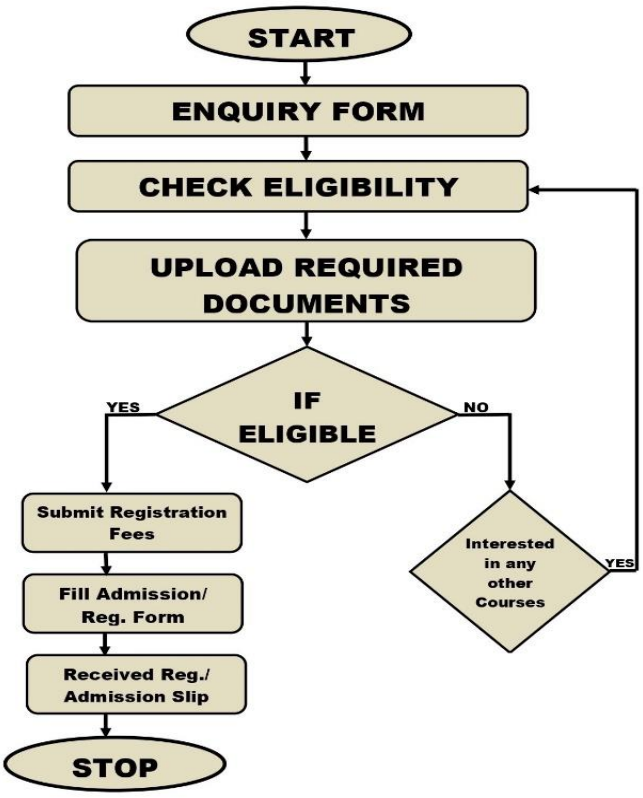
Step 6:Submit registration fees

Fill admission form

Get the admission slip

Step 7: Stop

Flowchart:



Exp No : 1) f. Flowchart for retail shop bill calculation

Aim :

To draw a flowchart for retail bill preparation

Algorithm :

Step 1: Start

Step 2: Read item name

Step 3: Read price per unit

Step 4: Read No of Quantity

Step 5: Compute Totalcost=Quantity*price per unit

Step 6: Display total

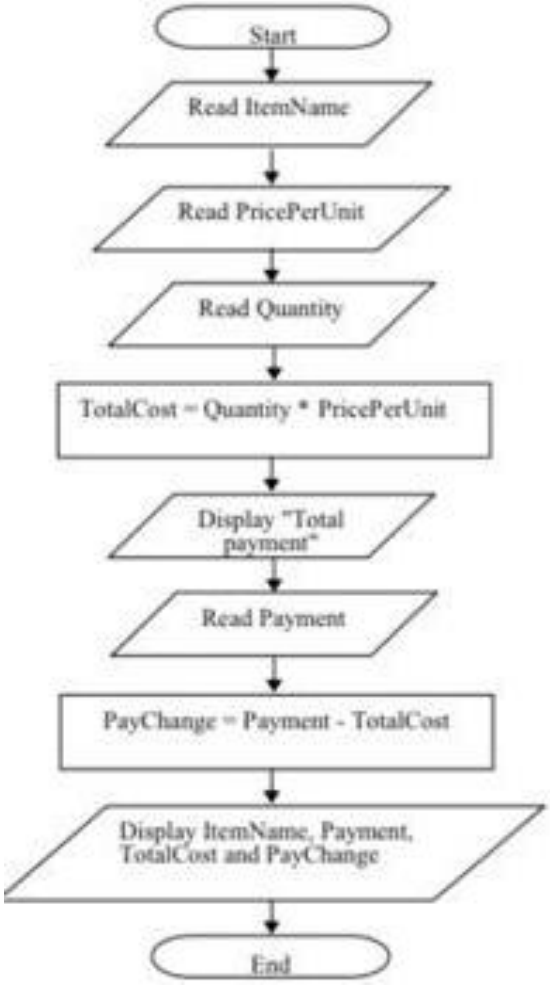
Step 7: Read payment

Step 8: Compute paychange=payment-Totalcost

Step 9: Display the items,payment and paychange

Step 10: Stop

Flowchart:


Start
Read ItemName
Read PricePerUnit
Read Quantity
TotalCost = Quantity * PricePerUnit
Display "Total payment"
Read Payment
PayChange = Payment - TotalCost
Display ItemName, Payment, TotalCost and PayChange
End

Exp No : 1) g. Flowchart for computing sin series

Aim :

To draw a flowchart for computing sin series

Algorithm :

Step 1: Take in the value of x in degrees and the number of terms and store it in separate variables.

Step 2: Pass these values to the sine function as arguments.

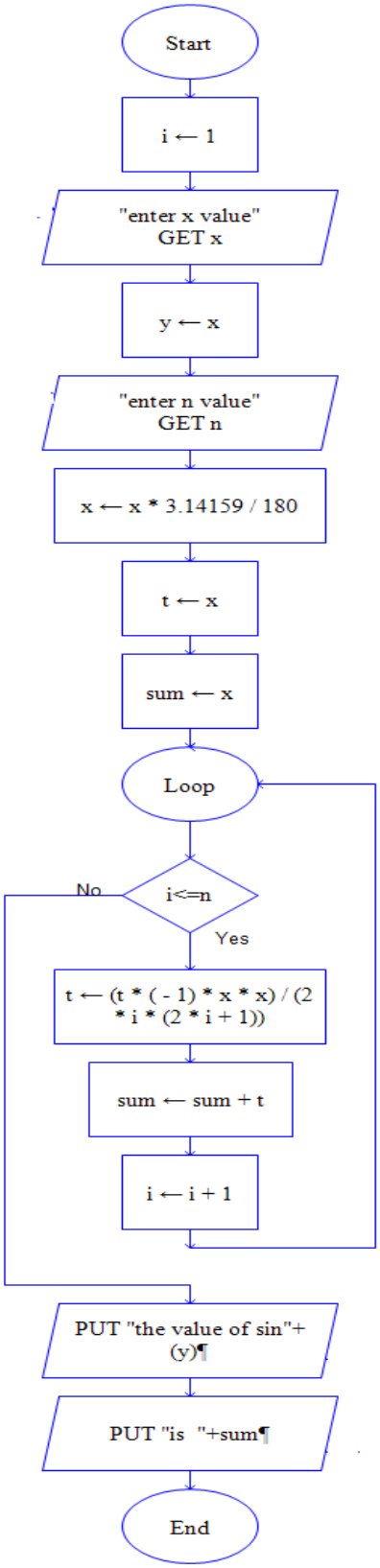
Step 3: Define a sine function and using a for loop, first convert degrees to radians.

Step 4: Then use the sine formula expansion and add each term to the sum variable.

Step 5: Then print the final sum of the expansion.

Step 6: Stop

Flowchart:


Start
i ← 1
"enter x value"
GET x
y ← x
"enter n value"
GET n
x ← x * 3.14159 / 180
t ← x
sum ← x
Loop
No
i<=n
Yes
t ← (t * (- 1) * x * x) / (2 * i * (2 * i + 1))
sum ← sum + t
i ← i + 1
PUT "the value of sin"+ (y)¶
PUT "is "+sum¶
End

Exp No : 1) h. Flowchart for calculating the weight of a motorbike

Aim :

To draw a flowchart for weight of a motorbike.

Algorithm :

Step 1: Start

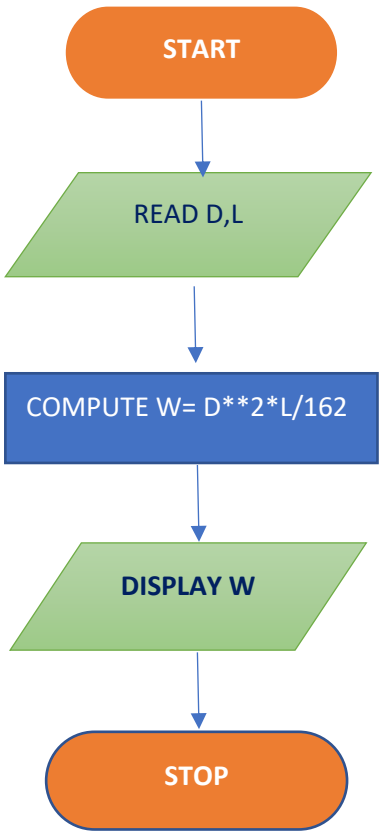
Step 2: Read D,L

Step 3: Compute w=D**2L/162

Step 4: Print w

Step 5: Stop

Flowchart:



Exp No : 1) i. Flowchart for calculating the weight of a Steelbar

Aim :

To draw a flowchart for weight of a steel bar

Algorithm :

Step 1: Start

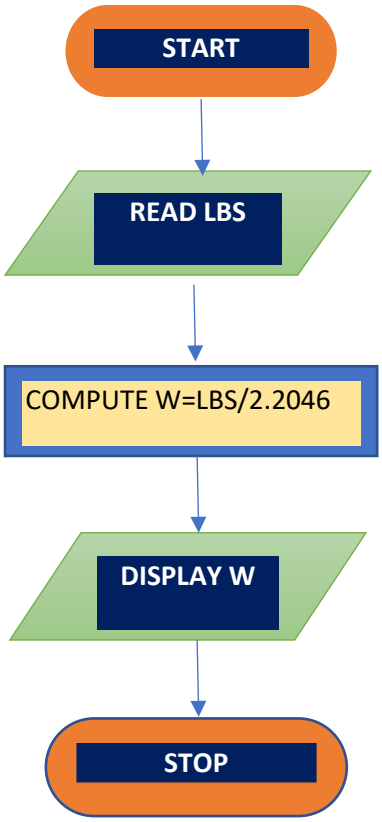
Step 2: Read LBS

Step 3: Compute w=LBS/2.2046

Step 4: Print w

Step 5: Stop

Flowchart:



Exp No : 1) j. Flowchart for calculating the compute Electrical Current in Three Phase AC Circuit

Aim :

To draw a flowchart for compute Electrical Current in Three Phase AC Circuit

Algorithm :

Step 1: Start

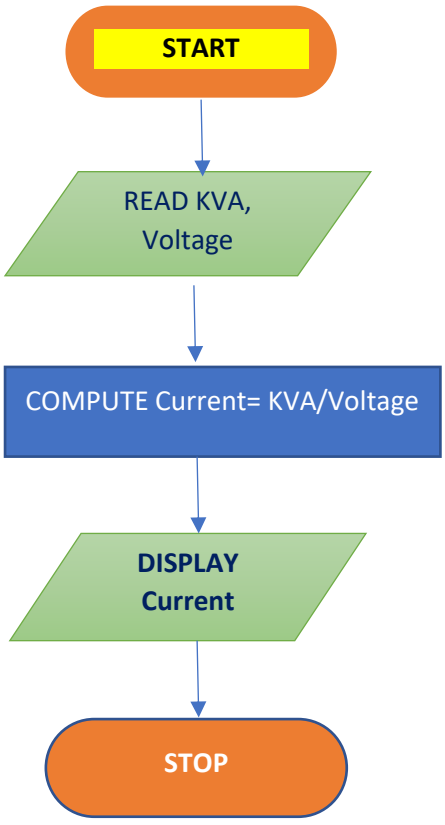
Step 2: Read kva,voltage

Step 3: Compute current=kva/voltage

Step 4: Print current

Step 5: Stop

Flowchart:



Result: Thus the flow charts were successfully drawn.

Ex. 2. Python programs using simple statements and expressions

Aim:

To write python programs using simple statements and expressions.

2(a). Exchange the value of two variables

Algorithm:

STEP 1: Read the values of P and Q

STEP 2: temp = P

STEP 3: P = Q

STEP 4: temp = P

STEP 5: PRINT P, Q

Program:

```
P = int( input("Please enter value for P: "))
Q = int( input("Please enter value for Q: "))
temp= P
P = Q
Q = temp
print ("The Value of P after swapping: ", P)
print ("The Value of Q after swapping: ", Q)
```

Output:

```
Please enter value for P: 5
Please enter value for Q: 7
The Value of P after swapping:  7
The Value of Q after swapping:  5
```

2 (b) Circulate ‘n’ values

Algorithm:

STEP 1: Read the values to be circulated in a list

STEP 2: Read the number of times of shifting in n

STEP 3: Using list slicing, perform rotation

STEP 4: Print (list[n:]+list[:n])

STEP 5: Stop

Program:

```
list=[10,20,30,40,50]
n=2
print(list[n:]+list[:n])
```

Output:

[30, 40, 50, 10, 20]

2 (c) Distance between two points

Algorithm:

STEP 1: Read the values of x1,x2 and y1,y2

STEP 2: Import math module to calculate the square root

STEP 3: Compute the distance using the formula

STEP 4: Using the formula, find square root of (((p1[0]-p2[0])**2)+((p1[1]-p2[1])**2))

STEP 5: Print the result

STEP 6: Stop

Program:

```
import math
p1 = [4, 0]
p2 = [6, 6]
distance = math.sqrt( ((p1[0]-p2[0])**2)+((p1[1]-p2[1])**2) )
print(distance)
```

Output:

6.324555320336759

2 (d) Calculate the Area of a Triangle

Algorithm:

STEP 1: Read the values of three sides of a triangle

STEP 2: Calculate ‘s’ value

STEP 3: Calculate the area using the formula (s*(s-a)*(s-b)*(s-c))

STEP 4: Find the square root of the result

STEP 5: Print the result

STEP 6: Stop

Program:

```
a = float(input('Enter first side: '))
b = float(input('Enter second side: '))
c = float(input('Enter third side: '))
s = (a + b + c) / 2
area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
print('The area of the triangle is:', area)
```

Output:

Enter first side: 5

Enter second side: 4

Enter third side: 2

The area of the triangle is: 3.799671038392666

2 (e) Convert Kilometers to Miles

Algorithm:

STEP 1: Read the values of distance in kilometers

STEP 2: Read the conversion factor

STEP 3: Calculate the miles using kilometers * conv_fac

STEP 4: Print the result

STEP 5: Stop

Program:

```
kilometers = float(input("Enter value in kilometers: "))
conv_fac = 0.621371
miles = kilometers * conv_fac
print('kilometers is equal to:', miles , 'miles')
```

Output:

Enter value in kilometers: 20

kilometers is equal to: 12.42742 miles

2 (f) Convert Celsius To Fahrenheit

Algorithm:

STEP 1: Read the values of temperature in celcius

STEP 2: Calculate the temperature using (celsius * 1.8) + 32

STEP 4: Print the result

STEP 5: Stop

Program:

```
celsius = float(input("Enter the temperature in celcius:"))
fahrenheit = (celsius * 1.8) + 32
print(' Celsius is equal to ', fahrenheit)
```

Output:

```
Enter the temperature in celcius:3.5
 Celsius is equal to  38.3
```

2 (g) Calculation of Simple Interest and Compound Interest

Algorithm:

STEP 1: Read the values of principal amount, no of years and rate of interest

STEP 2: Calculate the simple interest using (principal*time*rate)/100

STEP 3: Calculate the compound interest using principal * ((1+rate/100)**time - 1)

STEP 4: Display the two values

STEP 5: Stop

Program:

```
principal = float(input('Enter amount: '))
time = float(input('Enter time: '))
rate = float(input('Enter rate: '))
simple_interest = (principal*time*rate)/100
```

```
compound_interest = principal * ( (1+rate/100)**time - 1)
print('Simple interest is:',simple_interest)
print('Compound interest is:' , compound_interest)
```

Output:

```
Enter amount: 20000
Enter time: 5
Enter rate: 6.7
Simple interest is: 6700.0
Compound interest is: 7659.994714602134
```

2 (h) Calculation of Simple Interest and Compound Interest

Algorithm:

STEP 1: Read the values two numbers

STEP 2: Perform all arithmetic operations using appropriate operators

STEP 3: Use print statements to print the calculated values

STEP 4: Display the results

STEP 5: Stop

Program:

```
num1 = float(input('Enter first number: '))
num2 = float (input('Enter second number: '))
sum1=num1+num2
diff1=num1-num2
prod1=num1*num2
div1=num1/num2
rem1=num1%num2
floordiv1=num1//num2
exp1=num1**num2
```

```
print("Sum is:", sum1)
print("Difference is:", diff1)
print("Product is:", prod1)
print("Quotient is:", div1)
print("Reminder is:", rem1)
print("Floor division result is:", floordiv1)
print("Exponent is:", exp1)
```

Output:

```
Enter first number: 2
Enter second number: 3
Sum is: 5.0
Difference is: -1.0
Product is: 6.0
Quotient is: 0.6666666666666666
Reminder is: 2.0
Floor division result is: 0.0
Exponent is: 8.0
```

2 (i) Display Random value in python

Algorithm:

STEP 1: Read the values in n

STEP 2: Using import, import random in python

STEP 3: Use print statements, print the random value

STEP 4: Display the results

STEP 5: Stop

Program:

```
import random
n = random.random()
print(n)
```

Output:

0.8496781335157302

Result:

Thus, the program for simple python statements and expressions were executed and the output is verified.

Ex. 3. Scientific problems using Conditionals and Iterative loops

Aim:

To write python programs using conditional statements and loops.

3(a). Biggest of three numbers

Algorithm:

Step 1: Ask the user to enter three integer values.

Step 2: Read the three integer values in num1, num2, and num3.

Step 3: Check if num1 is greater than num2.

If true, then check if num1 is greater than num3.

If true, then print ‘num1’ as the greatest number.

Step 4: If false, then print ‘num3’ as the greatest number.

If false, then check if num2 is greater than num3.

I f true, then print ‘num2’ as the greatest number.

If false, then print ‘num3’ as the greatest number

Step 5: Stop

Program:

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))
if (num1 > num2) and (num1 > num3):
   largest = num1
elif (num2 > num1) and (num2 > num3):
   largest = num2
else:
   largest = num3
print("The largest number is",largest)
```

Output:

Enter first number: 55

Enter second number: 64

Enter third number: 34

The largest number is 64.0

3 (b) Sum of digits of a number

Algorithm:

Step 1: User must first enter the value and store it in a variable.

Step 2: The while loop is used and the last digit of the number is obtained by using the modulus operator.

Step 3: The digit is added to another variable each time the loop is executed.

Step 4: This loop terminates when the value of the number is 0.

Step 5: The total sum of the number is then printed.

Step 6: Stop

Program:

```
n=int(input("Enter a number:"))
tot=0
while(n>0):
    dig=n%10
    tot=tot+dig
    n=n//10
print("The total sum of digits is:",tot)
```

Output:

Enter a number:456

The total sum of digits is: 15

3 (c) Perfect Number or Not

Algorithm:

Step 1: Start

Step 2: Read n

Step 3: Initialize s=0

Step 4: for i=1 to n do

if(n%i)==0, then

s=s+i

Step 5: if s==n

then Print "Given Number is Perfect Number". Goto Step 7

Step 6: Print "Given Number is Not a Perfect Number"

Step 7: Stop

Program:

```
Number = int(input(" Please Enter any Number: "))
Sum = 0
for i in range(1, Number):
   if(Number % i == 0):
      Sum = Sum + i
if (Sum == Number):
   print(" %d is a Perfect Number" %Number)
else:
   print(" %d is not a Perfect Number" %Number)
```

Output:

```
Please Enter any Number: 6
 6 is a Perfect Number
```

3 (d) Printing Fibonacci Series

Algorithm:

Step 1: Start

Step 2: Declare variables i, t1,t2 , sum

Step 3: Initialize the variables, t1=0, t2=1, and sum=0

Step 4: Enter the number of terms of Fibonacci series to be printed

Step 5: Print First two terms of series

Step 6: Use loop for the following steps

-> sum=t1+t3

-> t1=t2

-> t2=sum

-> increase value of i each time by 1

-> print the value of show

Step 7: Stop

Program:

```
n = int(input("Enter the value of 'n': "))
t1 = 0
t2 = 1
sum = 0
count = 1
print("Fibonacci Series: ")
while(count <= n):
  print(sum, end = " ")
  count += 1
  t1 = t2
  t2 = sum
  sum = t1 + t2
```

Output:

Enter the value of 'n': 7

Fibonacci Series:

0 1 1 2 3 5 8

3 (e) Print Different Patterns in python

Algorithm:

STEP 1: Read the row and column values

STEP 2: Using nested loops, print the required pattern

STEP 3: Stop

Pattern 1:

```
num_rows = int(input("Enter the number of rows"));
for i in range(0, num_rows):
  for j in range(0, num_rows-i-1):
    print(end=" ")
  for j in  range(0, i+1):
    print("*", end=" ")
  print()
```

Output:

Enter the number of rows5

```
    *
   * *
  * * *
 * * * *
* * * * *
```

Pattern 2:

```
num_rows = int(input("Enter the number of rows"));
k = 1
for i in range(0, num_rows):
  for j in range(0, k):
    print("* ", end="")
  k = k + 1
  print()
```

Output:

```
Enter the number of rows5
*
* *
* * *
* * * *
* * * * *
```

Pattern 3:

```
num_rows = int(input("Enter the number of rows"));
k = 8
for i in range(0, num_rows):
  for j in range(0, k):
    print(end=" ")
  k = k - 2
  for j in range(0, i+1):
    print("* ", end="")
  print()
```

Output:

Enter the number of rows5

```
    *
   * *
  * * *
 * * * *
* * * * *
```

Pattern 4:

```
num_rows = int(input("Please enter the number of rows"));
for i in range(num_rows,0,-1):
  for j in range(0, num_rows-i):
    print(end=" ")
  for j in range(0,i):
    print("* ", end=" ")
  print()
```

Output:

Please enter the number of rows5

```
*  *  *  *  *
 *  *  *  *
  *  *  *
   *  *
    *
```

Pattern 5:

```
rows = 6
for i in range(rows):
    for j in range(i):
        print(i, end=' ')
```

```
    print('')
```

Output:

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

3 (f) Pascals Triangle

Algorithm:

STEP 1: Read the number of rows to be printed

STEP 2: Using nested loop, print the spaces needed

STEP 3: Using binomial coefficient, print the values

STEP 4: Stop

Program:

```
n = int(input("Enter the rows:"))
for i in range(1, n+1):
    for j in range(0, n-i+1):
        print(' ', end='')
    C = 1
    for j in range(1, i+1):
        print(' ', C, sep='', end='')
        C = C * (i - j) // j
    print()
```

Output:

Enter the rows:5

```
    1
   1 1
  1 2 1
 1 3 3 1
1 4 6 4 1
```

3 (g) Alphabet Patterns in Python

Algorithm:

STEP 1: Read the row and column values

STEP 2: Using nested loops, print the required pattern

STEP 3: Stop

Pattern 1:

```
for i in range (65,70):
    # inner loop
    for j in range(65,i+1):
        print(chr(j),end="")
    print()
```

Output:

A

AB

ABC

ABCD

ABCDE

Pattern 2:

```
for i in range(65,70):
    k=i
    # Inner loop
    for j in range(65,i+1):
        print(chr(k),end="")
        k=k+1
    print()
```

Output:

A

BC

CDE

DEFG

EFGHI

Pattern 3:

```
str= input("Enter your name:")
for i in range(0,len(str)):
    for j in range(0,i+1):
        print(str[j],end="")
    print()
```

Enter your name:Ajin

A

Aj

Aji

Ajin

Pattern 4:

```
n = int(input("Enter the row"))
for i in range(n):
    # printing spaces
    for j in range(i):
        print(' ', end='')
    # printing alphabet
    for j in range(2*(n-i)-1):
        print(chr(65 + j), end='')
    print()
```

Output:

```
Enter the row5
ABCDEFGHI
 ABCDEFG
  ABCDE
   ABC
    A
```

Pattern 5:

```
n = 5

# upward pyramid
for i in range(n):
    for j in range(n - i - 1):
        print(' ', end='')
    for j in range(2 * i + 1):
        print(chr(65 + j), end='')
    print()
for i in range(n - 1):
```

```
    for j in range(i + 1):
        print(' ', end='')
    for j in range(2*(n - i - 1) - 1):
        print(chr(65 + j), end='')
    print()
```

Output:

```
     A
    ABC
   ABCDE
  ABCDEFG
 ABCDEFGHI
  ABCDEFG
   ABCDE
    ABC
     A
```

3 (h) Prime Numbers in a range

Algorithm:

Step 1: Start

Step 2: Read lower and upper range

Step 3: Using a for loop, iterate on all the numbers in the range

Step 4 : For each number, check if its prime or not

Step 5 : If the number is prime, print

Else skip

Step 6 : Stop

Program:

```
lower = 900
upper = 1000
print("Prime numbers between", lower, "and", upper, "are:")
for num in range(lower, upper + 1):
  if num > 1:
      for i in range(2, num):
          if (num % i) == 0:
              break
      else:
          print(num)
```

Output:

Prime numbers between 900 and 1000 are:

907

911

919

929

937

941

947

953

967

971

977

983

991

997

Result:

Thus, the program using conditionals and loops has been executed and verified.

Ex. 4. Implementing real-time/technical applications using Lists, Tuples

Aim:

To write python programs using Lists and Tuples.

4(a). Display the Book details using a list

Algorithm:

Step 1: Start

Step 2: Create an empty list

Step 3: Print the menu and read the choice

Step4 : If the choice is 1, read the book name from the user and append to list

Step 5: If the choice is 2, print the books in the list

Step 6: Stop

Program:

```
my_book=[]
ch='y'
while(ch=='y'):
    print('''
1. Add New Book
2. Display Books
''')
    choice=int(input("Enter choice: "))
    if(choice==1):
        book=input("Enter the name of the book:")
        my_book.append(book)
    elif(choice==2):
        for i in my_book:
            print(i)
    else:
        print("Invalid choice!")
```

```
    ch=input("Do you want to continue...?")
print("Bye!")
```

Output:

```
1. Add New Book
2. Display Books
Enter choice: 1
Enter the name of the book:Python programming
Do you want to continue...?y
```

4(b). Automobile Components using Lists

Algorithm:

Step 1: Start

Step 2: Create a list of components of automobile

Step 3: Print the menu and read the choice

Step 4 : If the choice is 1, display the main components in the list

Step 5: If the choice is 2, print the components of list

Step 6: If the choice is 3, Print the total components

Step 7: If the choice is 4, arrange the components in order

Step : Stop

Program:

```
components=['RADIATOR','AC
COMPRESSOR','BATTERY','ALTERNATOR','AXLE','BRAKES','SHOCK
ABSORBERS','TRANSMISSION','FUEL TANK']
components1=['CATALYTIC CONVERTER','MUFFLER','TAILPIPE']
ch='y'
while(ch=='y'):
    print('''
1. Display Main Components
2. Display All Components
3. Total Components
4. Components in Alphabetical Order
```

```
''')
    choice=int(input("Enter choice: "))
    if(choice==1):
        print("The Main components are:")
        print(components)

    elif(choice==2):
        print("The components are:")
        print(components+components1)
    elif(choice==3):
        print("The components are:")
        print("Main Components:",len(components))
        print("Other Components:",len(components1))
    elif(choice==4):
        print("The components in alphabetical order are:")
        components.sort()
        print("Main Components:",components)
        print("Sorted order")

    else:
        print("Invalid choice!")
    ch=input("Do you want to continue...?")
print("Bye!")
```

Output:

1. Display Main Components
2. Display All Components
3. Total Components
4. Components in Alphabetical Order

Enter choice: 4

The components in alphabetical order are:

Main Components: ['AC COMPRESSOR', 'ALTERNATOR', 'AXLE', 'BATTERY', 'BRAKES', 'FUEL TANK', 'RADIATOR', 'SHOCK ABSORBERS', 'TRANSMISSION']

Sorted order

Do you want to continue...?

4(c). Materials of Construction using Tuple

Algorithm:

Step 1: Start

Step 2: Create a tuple of construction materials

Step 3: Print the menu and read the choice

Step 4 : If the choice is 1, display the materials in the tuple

Step 5: If the choice is 2, display the types of materials in the tuple

Step 6: If the choice is 3, Print the reverse order of materials

Step 7: If the choice is 4, print the total number

Step 8: If the choice is 5, print any material at random

Step 9: Stop

Program:

```
types=('Artificial Materials','Natural Materials')
material=('Cement','Aggregates','Stones and Rocks','Mud and Clay',
'Concrete','Bricks','Glass','Plastic','Structural Steel',
'Foam','Fabric','Thatch','Timber and Wood','Tiles (Ceramics)',
'Electrical Items')
ch='y'
while(ch=='y'):
    print('''
1. Display Materials
2. Display types of Materials
```

```
3. Display Reverse order of materials
4. Count of Needed materials
5. Choose a material at random
''')
    choice=int(input("Enter choice: "))
    if(choice==1):
        print(material)

    elif(choice==2):
        print(types)
    elif(choice==3):
        print(material[::-1])
    elif(choice==4):
        print(len(material))
    elif(choice==5):
        print(material[5])

    else:
        print("Invalid choice!")
    ch=input("Do you want to continue...?")
print("Bye!")
```

Output:

```
1. Display Materials
2. Display types of Materials
3. Display Reverse order of materials
4. Count of Needed materials
5. Choose a material at random
```

Enter choice: 4

15

Do you want to continue...?y

1. Display Materials

2. Display types of Materials

3. Display Reverse order of materials

4. Count of Needed materials

5. Choose a material at random

Enter choice: 5

Bricks

Do you want to continue...?

4(d). List Operations

Algorithm:

Step 1: Start

Step 2: Create a list of items

Step 3: Print the output for each of the operations

Indexing, extend, pop, slicing, clear, remove and del

Step 4: Stop

Program:

```
numbers = [11, 22, 33, 100, 200, 300]
print(numbers[0])
print(numbers[5])
print(numbers[1])
print(numbers[-1])
print(numbers[-3])
```

```
print(numbers[-4])
n_list = ["hello", "world", "hi", "bye"]
print(n_list[1:3])
print(n_list[:3])
print(n_list[3:])
print(n_list[:])
n_list.insert(3, 100)
print(n_list)
n_list.append(99)
print(n_list)
n_list.extend([11, 22])
print(n_list)
n_list[2] = 100
print(n_list)
n_list[1:4] = [11, 22, 33]
print(n_list)
del n_list[2:4]
print(n_list)
ch_list = ['A', 'F', 'B', 'Z', 'O', 'L']
ch_list.remove('B')
print(ch_list)
ch_list.pop(1)
print(ch_list)
ch_list.clear()
print(ch_list)
```

Output:

```
11
300
22
300
```

100

33

['world', 'hi']

['hello', 'world', 'hi']

['bye']

['hello', 'world', 'hi', 'bye']

['hello', 'world', 'hi', 100, 'bye']

['hello', 'world', 'hi', 100, 'bye', 99]

['hello', 'world', 'hi', 100, 'bye', 99, 11, 22]

['hello', 'world', 100, 100, 'bye', 99, 11, 22]

['hello', 11, 22, 33, 'bye', 99, 11, 22]

['hello', 11, 'bye', 99, 11, 22]

['A', 'F', 'Z', 'O', 'L']

['A', 'Z', 'O', 'L']

[]

4(e). List Operations

Algorithm:

Step 1: Start

Step 2: Create a tuple of items

Step 3: Print the output for each of the operations

Type, del, sum, max, min, sorted, packing, slicing, indexing, membership

Step 4: Stop

Program:

```
type1 = (1, 3, 4, 5, 'test')
print (type1)
tuple1 = ('a', 'b', 1, 3, [5, 'x', 'y', 'z'])
print(tuple1[0])
print(tuple1[4][1])
```

```
print(tuple1[-1])
print(tuple1[2:5])
Tuple1 = (1, 3, 4)
Tuple2 = ('red’, ’green’, ‘blue')
Tuple3 = (Tuple1, Tuple2)
print (Tuple3)
del (Tuple2)
print (1 in Tuple1)
print (5 in Tuple1)
print(max(Tuple1))
print(sum(Tuple1))
print(sorted(Tuple1))
person = ("Sai", '5 ft', "Actor")
(name, height, profession) = person
print(name)
print(height)
print(profession)
```

Output:

```
(1, 3, 4, 5, 'test')
a
x
[5, 'x', 'y', 'z']
(1, 3, [5, 'x', 'y', 'z'])
((1, 3, 4), 'red", ’green’, "blue')
True
False
4
8
[1, 3, 4]
```

Sai

5 ft

Actor

Result: Thus, the applications and operations of list , tuple has been executed.

5. Implementing real-time/technical applications using Sets, Dictionaries.

Ex.No. 5(a) Components of an automobile using Set Operations

AIM:
To write a python program for to add ,remove and pop the automobile components using python set operations.

ALGORITHM:

1. Start the Program
2. Initialize the empty dictionary with components {}
3. Print the type of components {} so that dictionary type will be printed.
4. Initialize the components as set()
5. Add the values of automobile components to the set.
6. Get the extra component to be added to the set as input.
7. Add the extra component through add() method.
8. Print the set as components
9. Remove any one component from the set using pop() method.
10. Print the updated components.
11. Stop the Program.

PROGRAM:

```
components={}
print(type(components))
components=set()
print(type(components))
components={"Clutch","Gear Box ","Front dead axle","Engine","Rear drive
axle","Differential"}
print(components)
b=input("Enter the components to be added" )
components.add(b)
print("Components of automobile after adding an element in set")
print(components)
print("Removing the one Component")
components.pop()
print("Updated Components after removing an element is set")
print(components)
```

OUTPUT:

```
<class 'dict'>
<class 'set'>
{'Gear Box ', 'Front dead axle', 'Engine', 'Clutch', 'Differential', 'Rear drive axle'}
Enter the components to be addedAxle
Components of automobile after adding an element in set
{'Gear Box ', 'Front dead axle', 'Engine', 'Clutch', 'Differential', 'Rear drive axle', 'Axle'}
Removing the one Component
```

Updated Components after removing an element is set
{'Front dead axle', 'Engine', 'Clutch', 'Differential', 'Rear drive axle', 'Axle'}

Ex.No 5.(b) **Elements of a civil structure using dictionaries**
Date:

AIM:

To implement Elements of a civil structure using dictionaries in python

ALGORITHM:

1.Start the Program.
2. Create the empty dictionary with Elements_dictionary
3. Add the elements to dictionary with the key values.
4. Print the type of Elements_dictionary
5. Display the elements in the dictionary.
6. Remove the particular item in the Elements_dictionary by using key with pop() method.
7.Display the popped element in the civil structure
8. Add the single element into the Elements_dictionary
9. Display the modified dictionary elements.
10.Stop the Program

PROGRAM:

```
# Elements of a civil structure
Elements_dictionary = {}

# dictionary with integer keys
Elements_dictionary = {1: 'Roof', 2:
'Parapet',3:'Lintels',4:'Beams',5:'Columns',6:'Walls',7:'Floor',8:'Stairs'}
print(type(Elements_dictionary))
print("The dictionary elements of Civil Structure")
print(Elements_dictionary)
print("Remove the particular item in the Civil structure")
print(Elements_dictionary.pop(4))
Elements_dictionary[9]='Plinth Beam'
print("The Dictionary Elements after adding one elements in Civil Structure")
print(Elements_dictionary)
```

OUTPUT:

```
<class 'dict'>
The dictionary elements of Civil Structure
{1: 'Roof', 2: 'Parapet', 3: 'Lintels', 4: 'Beams', 5: 'Columns', 6: 'Walls', 7: 'Floor', 8: 'Stairs'}
Remove the particular item in the Civil structure
Beams
The Dictionary Elements after adding one elements in Civil Structure
{1: 'Roof', 2: 'Parapet', 3: 'Lintels', 5: 'Columns', 6: 'Walls', 7: 'Floor', 8: 'Stairs', 9: 'Plinth
Beam'
```

Ex.No.5(c) Add and Search student record using dictionary
Date:

AIM:
To implement a python Program to search student record using dictionary

ALGORITHM:
1.Create a list to store the student details
2. Read the data such as rno, name, marks
3. Calculate the total, average
4. Convert the list to the dictionary
5. Print the student information's
6. Stop the Program.

PROGRAM:
```
n = int(input("Please enter number of students:"))

all_students = []

for i in range(0, n):
    stud_name = input('Enter the name of student: ')
    print (stud_name)

    stud_rollno = int(input('Enter the roll number of student: '))
    print (stud_rollno)

    mark1 = int(input('Enter the marks in subject 1: '))
    print (mark1)

    mark2 = int(input('Enter the marks in subject 2: '))
    print (mark2)

    mark3 = int(input('Enter the marks in subject 3: '))
    print (mark3)

    total = (mark1 + mark2 + mark3)
    print("Total is: ", total)

    average = total / 3
    print ("Average is :", average)

    all_students.append({
                        'Name': stud_name,
                        'Rollno': stud_rollno,
                        'Mark1': mark1,
                        'Mark2': mark2,
                        'Mark3': mark3,
                        'Total': total,
```

```
                    'Average': average
                    })

for student in all_students:
   print ('\n')
   for key, value in student.items():
      print (key, value)
```

OUTPUT:

```
ENTER ZERO NUMBER FOR EXIT !!!!!!!!!!!
ENTER STUDENT INFORMATION ----------------------
enter roll no.. :: -- 1
enter marks   1  :: --   77
enter marks   2  :: --   88
enter marks   3  :: --   77
Please enter number of students:2
Enter the name of student: Alex
Alex
Enter the roll number of student: 1
1
Enter the marks in subject 1: 99
99
Enter the marks in subject 2: 88
88
Enter the marks in subject 3: 88
88
Total is:  275
Average is : 91.66666666666667
Enter the name of student: Avin
Avin
Enter the roll number of student: 2
2
Enter the marks in subject 1: 77
77
Enter the marks in subject 2: 77
77
Enter the marks in subject 3: 77
77
Total is:  231
Average is : 77.0

Name Alex
Rollno 1
Mark1 99
Mark2 88
Mark3 88
Total 275
Average 91.66666666666667
```

Name Avin
Rollno 2
Mark1 77
Mark2 77
Mark3 77
Total 231
Average 77.0

Ex.No.5(d) Operations in a set using Components of a Language

Aim:

To Implement operations in a set using Components of a Language as example.

ALGORITHM:

1. Start the Program
2. Initialize the dictionaries L1, L2
3. Perform the set operations
4. Stop

Program:

```
L1 = {'Pitch', 'Syllabus', 'Script', 'Grammar', 'Sentences'};

L2 = {'Grammar', 'Syllabus', 'Context', 'Words', 'Phonetics'};

print("Union of L1 and L2 is ",L1 | L2)

print("Intersection of L1 and L2 is ",L1 & L2)

print("Difference of L1 and L2 is ",L1 - L2)

print("Symmetric difference of L1 and L2 is ",L1 ^ L2)
```

Output:

```
Union of L1 and L2 is {'Words', 'Pitch', 'Sentences', 'Phonetics', 'Script', 'Grammar',

'Syllabus', 'Context'}

Intersection of L1 and L2 is {'Grammar', 'Syllabus'}

Difference of L1 and L2 is {'Script', 'Pitch', 'Sentences'}

Symmetric difference of L1 and L2 is {'Words', 'Context', 'Script', 'Pitch',

'Sentences', 'Phonetics'}
```

Result: Thus the basic set and dictionary operations were executed and the output verified.

Exp Num : 6.a **Implementing programs using Functions**

/*Program for Factorial of a number*/

Aim:

To write a python program for implementing a factorial of a given number using recursion.

Algorithm:

Step 1 : Start
Step 2 :Read a number
Step 3 : Pass the number as an argument to a recursive factorial function.
Step 4 : Define the base condition as the number to be lesser than or equal to 1 and return 1 if it is.
Step 5 : Otherwise call the function recursively with the number minus 1 multiplied by the number itself.
Step 6 : Print the result.
Step 7: Stop.

Program :

```
def fact(n):
  if n==1:
    return n
  else:
    return n*fact(n-1)
num = int(input("Enter a number: "))
print("The factorial of",num,"is",fact(num))
```

Output:

```
Enter a number: 5
The factorial of 5 is 120
```

Result:

Thus the program was implemented and executed successfully.

Exp Num : 6.b **/*Program to find the largest number in a list*/**

Aim :

To write a program to find the largest number in a list using python.

Algorithm :

Step 1 : Start
Step 2 : Read number of elements to be store in a list
Step 3 : Initialize an empty list lst = [].
Step 4 : Read each number in your python program using a for loop.
Step 5 : In the for loop append each number to the list.
Step 6 : Define a custom function, which is an accepted number list and used to find the largest number from list.
Step 7 : Call this custom function and store the function result.
Step 8 : Print the result.
Step 9 : Stop

Program :

```
def find_max( list ):
   max = list[ 0 ]
   for a in list:
      if a > max:
         max = a
   return max
num = int(input('How many numbers: '))
lst = []
for n in range(num):
   numbers = int(input('Enter number '))
   lst.append(numbers)
print("Maximum element in the list is :",find_max(lst))
```

Output:

```
How many numbers: 5
Enter number 10
Enter number 20
Enter number 30
Enter number 40
Enter number 50
Maximum element in the list is : 50
```

Result:

Thus the program was implemented and executed successfully.

Exp Num : 6.c **/*Program to find the area of different shapes*/**

Aim :

To write a program to find the area of different shapes using python.

Algorithm :

Step 1 : Start.
Step 2 : Call the appropriate shapes and get the inputs.
Step 3 : Compute the formula as the shapes.
Step 4 : Return the area
Step 5 : Print the area.
Step 6 : Stop

Program :

```
def cir():
  r=int(input("Enter the radious"))
  area=3.14*r*r
  return area
def tria():
  b=int(input("Enter the base"))
  h=int(input("Enter the height"))
  area=1/2*b*h
  return area
def square():
  a=int(input("Enter the side"))
  area=a*a
  return area
def rect():
  l=int(input("Enter the length"))
  w=int(input("Enter the width"))
  area=l*w
  return area

print(cir())
print(tria())
print(square())
print(rect())
```

Output:

```
Enter the radious2
12.56
Enter the base12
Enter the height2
12.0
Enter the side3
9
Enter the length12
Enter the width8
96
```

Result:

Thus the program was implemented and executed successfully.

Exp Num : 6.d **/*Program to find the sum of list of elements using function*/**

Aim :

To write a program to sum of a numbers in a list using function

Algorithm :

Step 1 : Start.
Step 2 : Create a empty list l=[]
Step 3 : Read number of elements to store in a list.
Step 4 : Get the input and store in a list using for loop
Step 5 : Make a call to sum_list() function.
Step 6 : Initialize sum=0
Step 7 : Iterate the elements using for loop and add the elements.
Step 8 : Return the sum
Step 9 : Print the sum
Step 10: Stop

Program :

```
def sum_list(l):
  sum=0
  for i in range(len(l)):
    sum=sum+l[i]
  return sum
l=[]
n=int(input("Enter the number of elements"))
for i in range(n):
  x=int(input("Enter the element"))
  l.append(x)
result=sum_list(l)
print(result)
```

Output:

```
Enter the number of elements5
Enter the element10
Enter the element20
Enter the element30
Enter the element40
Enter the element50
150
```

Result:

Thus the program was implemented and executed successfully.

Exp Num : 6.e /*Program to create a Simple arithmetic calculator using function*/

Aim :

To write a program to create a simple arithmetic calculator using function

Algorithm :

Step 1 : Take the values of two numbers from the user using the int(input()) function and store it in two variable say num1 and num2
Step 2 : Call function add and pass the values of num1 and num2 as an argument to the add function.
Step 3 : Call function sub and pass the values of num1 and num2 as an argument to the sub function.
Step 4 : Call function mul and pass the values of num1 and num2 as an argument to the mul function.
Step 5 : Call function div and pass the values of num1 and num2 as an argument to the div function.
Step 6 : Copy the values of num1 and num2 at variable n1,n2.
Step 7 : Create a user defined functions to say add, which takes the values of n1 and n2 as an argument using def keyword and return calculated result.
Step 8 : Create a user defined functions to say sub, which takes values of n1 and n2 as an argument using def keyword and return calculated result.
Step 9 : Create a user defined functions to say mul, which takes values of n1 and n2 as an argument using def keyword and return calculated result.
Step 10 : Create a user defined functions to say div, which takes values the of n1 and n2 as an argument using def keyword and return calculated result.
Step 11 : Print the result as output and exit.

Program :

```
# Program make a simple calculator

# This function adds two numbers
def add(x, y):
   return x + y

# This function subtracts two numbers
def subtract(x, y):
   return x - y

# This function multiplies two numbers
def multiply(x, y):
   return x * y

# This function divides two numbers
def divide(x, y):
   return x / y


print("Select operation.")
```

```
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")

while True:
   # take input from the user
   choice = input("Enter choice(1/2/3/4): ")

   # check if choice is one of the four options
   if choice in ('1', '2', '3', '4'):
      num1 = float(input("Enter first number: "))
      num2 = float(input("Enter second number: "))

      if choice == '1':
         print(num1, "+", num2, "=", add(num1, num2))

      elif choice == '2':
         print(num1, "-", num2, "=", subtract(num1, num2))

      elif choice == '3':
         print(num1, "*", num2, "=", multiply(num1, num2))

      elif choice == '4':
         print(num1, "/", num2, "=", divide(num1, num2))

      # check if user wants another calculation
      # break the while loop if answer is no
      next_calculation = input("Let's do next calculation? (yes/no): ")
      if next_calculation == "no":
        break

   else:
      print("Invalid Input")
```

Output

```
Select operation.
1.Add
2.Subtract
3.Multiply
4.Divide
Enter choice(1/2/3/4): 3
Enter first number: 15
Enter second number: 14
15.0 * 14.0 = 210.0
Let's do next calculation? (yes/no): no
```

Result:

Thus the program was implemented and executed successfully.

Exp Num : 6.f /*Program to find the GCD of a number using Euclidean Algorithm*/

Aim :

To write a Program to find the GCD of a number using Euclidean Algorithm

Algorithm :

Step 1 : Take two numbers from the user.

Step 2. Pass the two numbers as arguments to a recursive function.

Step 3. When the second number becomes 0, return the first number.

Step 4. Else recursively call the function with the arguments as the second number and the remainder when the first number is divided by the second number.

Step 5. Return the first number which is the GCD of the two numbers.

Step 6. Print the GCD.

Step 7. Stop

Program :

```
def find_gcd(a,b):
    while(b):
        a, a = b, a % b
        return a
a = int(input (" Enter the first number: ") )
b = int(input (" Enter the second number: "))
num = find_gcd(a, b)
print("  The GCD of two number a and b is ")
print(num) # call num
```

Output

Entre the first number :144

Enter the second number: 96

The GCD of two number and b is 48

7. Implementing programs using Strings

AIM:

To implement python program using strings.

7(a). REVERSE A STRING

ALGORITHM:

STEP1: Get the input from the user

STEP2: The function call is invoked, where the string is passed as input to reverse function

STEP3: For loop is executed until the last character in the string is reached

STEP4: The str variable is incremented for each iteration and returned as the reversed string.

PROGRAM:

```
def reverse(s):
    str = ""
    for i in s:
    str = i + str
    return str
s =input("Enter the string to reverse:")
print ("The original string  is : ",end="")
print (s)
print ("The reversed string is : ",end="")
print (reverse(s))
```

OUTPUT:

```
Enter the string to reverse:python
The original string  is : python
The reversed string is : nohtyp
```

7(b). PALINDROME OF A STRING

ALGORITHM:

STEP1: Get the input string from the user.

STEP2: Check whether string is equal to the reverse of the same string using slicing operation in an if condition.

STEP3: If the condition is satisfied, then the string is a palindrome.

PROGRAM:

```
string=input(("Enter a letter:"))
if(string==string[::-1]):
    print("The letter is a palindrome")
else:
    print("The letter is not a palindrome")
```

OUTPUT:

Enter a letter: mom

The letter is a palindrome

7(c). CHARACTER COUNT

ALGORITHM:

STEP1: Get the string from the user

STEP2: Get the character to be counted in the string from the user.

STEP3: For loop is executed until the last character is checked in the string

STEP4: Using the if condition, the character entered and the characters in the string is compared inside the for loop for each iteration.

STEP5: The number of times the entered character has been present in the string will be returned as the total count.

PROGRAM:

```
test_str = input("Enter the String:")
character=input("Enter the Character:")
count = 0
```

```
for i in test_str:
    if i == character:
        count = count + 1
 print ("Count is : "+ str(count))
```

OUTPUT:

Enter the String: expression

Enter the Character: s

Count is : 2

7(d). REPLACING CHARACTERS

ALGORITHM:

STEP1: Read the String

STEP2: Replace the character in the string using replace() method.

STEP3: Display the replaced string.

PROGRAM:

```
string = " HI! HI! HI! WELCOME TO PYTHON PROGRAMMING"
print ("Original String:",string)
replace1=string.replace("HI", "HELLO")
print ("Replaced String 1:",replace1)
replace2=replace1.replace("HELLO", "Hello", 3)
print ("Replaced String 2:",replace2)
```

OUTPUT:

Original String: HI! HI! HI! WELCOME TO PYTHON PROGRAMMING

Replaced String 1: HELLO! HELLO! HELLO! WELCOME TO PYTHON PROGRAMMING

Replaced String 2: Hello! Hello! Hello! WELCOME TO PYTHON PROGRAMMING

7(e). BASIC STRING OPERATIONS

ALGORITHM

STEP1: Declare a string

STEP2: Access a word from the string using index value

STEP3: Carry out the slicing operations either by using the forward slicing or backward slicing.

STEP4: Concatenation of two strings are done by using “+” operator.

STEP5: Repetition of a string is done by using “*” operator.

STEP6: Check whether the words are present in the string by using membership operator.

STEP7: Print the string in uppercase by using upper() method.

STEP8: Length of the string is calculated by using len() method.

PROGRAM:

```
a="hello"
print("String is: ",a)
#accessing a string
print("Accessed String is: ",a[3])
#Slicing Operation
print("Slicing Operation:",a[0:3])
print("Forward Slicing:", a[0:4])
print("Backward Slicing:", a[-3:-1])
print("Slicing using size:", a[1:4:2])
#concatenation
print("concatenation:", a+"Hi")
#repetition
print("Repetition:", a*3)
#membership
print("Membership e in a:", 'e' in a)
print("Membership v in a:", 'v' in a)
#methods
```

```
print("string in Uppercase:", a.upper())
print("Length of string:", len(a))
```

OUTPUT:

```
String is:  hello
Accessed String is:  l
Slicing Operation: hel
Forward Slicing: hell
Backward Slicing: ll
Slicing using size: el
concatenation: helloHi
Repetition: hellohellohello
Membership e in a: True
Membership v in a: False
string in Uppercase: HELLO
Length of string: 5
```

RESULT:

Thus the python program to implement the string was done successfully and the output was verified.

8. Implementing programs using written modules and Python Standard Libraries

Ex.No. 8(a) Simple Program using numpy

AIM:
To write a python program to plot a simple python graph for a straight line.

ALGORITHM:

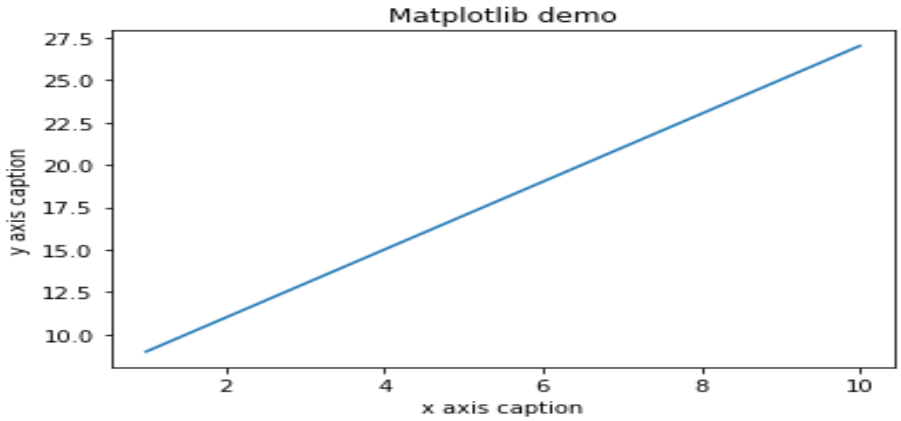
1. Start the Program
2. Import the modules numpy, pyplot
3. Store the set of values in x, equation in y
4. Assign title, axis label for the graph
5. Show the graph
6. Stop

PROGRAM:

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

x = np.arange(1,11)
y = 2 * x + 7
plt.title("Matplotlib demo")
plt.xlabel("x axis caption")
plt.ylabel("y axis caption")
plt.plot(x,y)
plt.show()
```

OUTPUT:



Ex.No 8.(b) **Program using pandas**
Date:

AIM:

To create data frames using pandas in python.

ALGORITHM:

1.Start the Program.
2. Import numpy and pandas
3. Create a dictionary with student name, marks of three subjects
4. Convert the dictionary to a data frame
5. Print the data frame
6. Stop

PROGRAM:

```
import numpy as np

import pandas as pd


mydictionary = {'names': ['Somu', 'Kiku', 'Amol', 'Lini'],

  'physics': [68, 74, 77, 78],

  'chemistry': [84, 56, 73, 69],

  'algebra': [78, 88, 82, 87]}


#create dataframe using dictionary

df_marks = pd.DataFrame(mydictionary)


print(df_marks)
```

OUTPUT:

```
names  physics  chemistry  algebra

0  Somu       68         84       78

1  Kiku       74         56       88

2  Amol       77         73       82

3  Lini       78         69       87
```

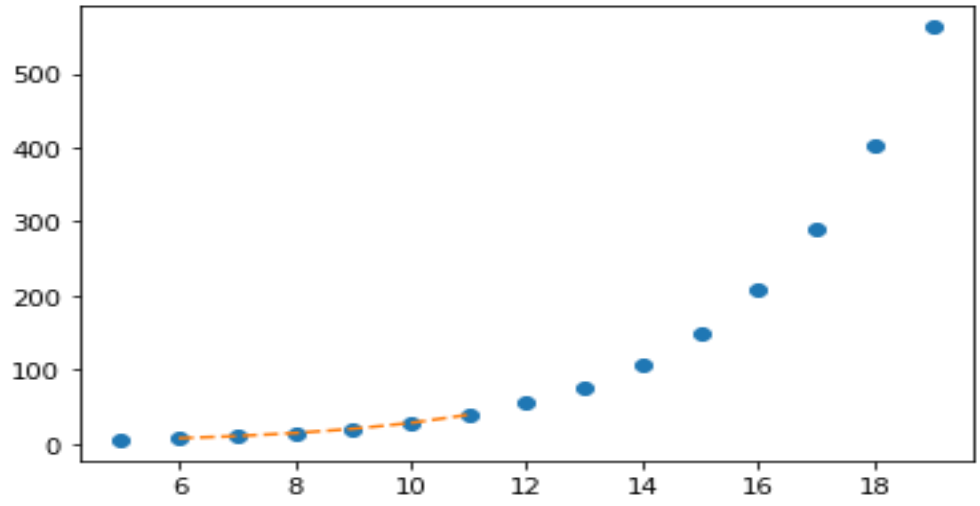
Ex.No.8(c) Using Scipy - Univariate Interpolation
Date:

AIM:
To implement a python Program to plot an univariate interpolation.

ALGORITHM:
1.Start
2. Import interpolate , pyplot
3. Using x, y values arrange the graph area
4. Using pyplot , print the graph
5. Stop

PROGRAM:
```
import matplotlib.pyplot as plt
from scipy import interpolate
x = np.arange(5, 20)
y = np.exp(x/3.0)
f = interpolate.interp1d(x, y)
x1 = np.arange(6, 12)
y1 = f(x1)   # use interpolation function returned by `interp1d`
plt.plot(x, y, 'o', x1, y1, '--')
plt.show()
```

OUTPUT:



Result:
Thus the programs using modules were implemented in python.

Exp Number 9. Implementing real-time/technical applications using File handling.

A. Python program to copy the content of one file into another file

Aim :

To write a program to copy the content of one file into anoqther file.

Algorithm:

Step 1: Start

Step 2: Open one file called test.txt in read mode.

Step 3: Open another file out.txt in write mode.

Step 4: Read each line from the input file and write it into the output file.

Step 5: Stop.

Program:

```
print("Enter the Name of Source File: ")
sourcefile = input()
print("Enter the Name of Target File: ")
targetfile = input()
fileHandle = open(sourcefile, "r")
texts = fileHandle.readlines()
fileHandle.close()
fileHandle = open(targetfile, "w")
for s in texts:
   fileHandle.write(s)
fileHandle.close()
print("File Copied Successfully!")
```

Output:

```
Enter the Name of Source File: in.txt
Enter the Name of Target File: out.txt

File Copied Successfully
```

B . Python Program to Count the Number of Words in a Text File

Aim :

To write a Python Program to Count the Number of Words in a Text File

Algorithm:

Step 1: Start

Step 2: Open file “book.txt” in read mode and store contents of file in file object say fin

Step 3: Read each line from the file using read() function

Step 4: Split the line to form a list of words using split() function and store it in variable say l.

Step 5: Intitially Set the value of count_words variable to zero in which we will store the calculated result.

Step 6: Use for loop to read list of word stored in variable say l.

Step 7: Find the length of words in the list and print it.

Step 8: Close the file using close() function.**Program:**

Step 9: Stop

Program:

```
fin = open("book.txt","r")
str = fin.read()
l = str.split()
count_words = 0
for i in l:
   count_words = count_words + 1
print(count_words)
fin.close()
```

Output:

25

C . Python Program to find longest Word **in a Text File**

Aim :

To write a Python Program to find longest Word in a Text File

Algorithm:

Step 1: Start

Step 2: Open text file say ‘name.txt’ in read mode using open function

Step 3: Pass file name and access mode to open function

Step 4: Read the whole content of text file using read function and store it in another variable say ‘str’

Step 5: Use split function on str object and store words in variable say ‘words’

Step 6: Find maximum word from words using len method

Step 7: Iterate through word by word using for loop

Step 8: Use if loop within for loop to check the maximum length of word

Step 9: Store maximum length of word in variable say ‘longest_word’

Step 10: Display longst_word using print function

Step 11: Stop

Program:

```
fin = open("name.txt","r")
str = fin.read()
words = str.split()
max_len = len(max(words, key=len))
for word in words:
   if len(word)==max_len:
      longest_word =word

print(longest_word)
```

Output:

Encyclopedia

Result :

Thus the program for real-time/technical applications using File handling was implemented and executed successfully.

10. Implementing real-time / technical applications using Exception handling

AIM:

To implement python program for real-time / technical applications using Exception handling.

10(a). DIVIDE BY ZERO ERROR

ALGORITHM:

STEP1: Get the input for n, d and c.

STEP2: In the try block, calculate q=n/(d-c).

STEP3: If the denominator is non-zero, then the quotient gets printed.

STEP4: Otherwise, the exception Zero Division Error is executed.

PROGRAM:

```
n=int(input("Enter the value of n:"))
d=int(input("Enter the value of d:"))
c=int(input("Enter the value of c:"))
try:
    q=n/(d-c)
    print("Quotient:",q)
except ZeroDivisionError:
    print("Division by Zero!")
```

OUTPUT:

```
Enter the value of n: 10
Enter the value of d: 5
Enter the value of c: 5
Division by Zero!
```

10(b). VOTER'S AGE VALIDITY

ALGORITHM:

STEP1: Get the age from the user.

STEP2: In the if block, check if the age is greater than 18, then print “Eligible to vote”.

STEP3: Otherwise, print “Not eligible to vote”

STEP4: If the age entered is a non-number, then the exception block is executed.

PROGRAM:

```
def main():
    try:
        age=int(input("Enter your age"))
        if age>18:
            print("Eligible to vote")
        else:
            print("Not eligible to vote")
    except:
        print("age must be a valid number")
main()
```

OUTPUT 1:

Enter your age 24

Eligible to vote

OUTPUT 2:

Enter your age 14

Not eligible to vote

10(c). STUDENT MARK RANGE VALIDATION

ALGORITHM:

STEP1: Get the number between the range(1 - 100)

STEP2: In the if block, check whether the number is between 1 to 100.

STEP3: Otherwise, print “Above 100, wrong”

STEP4: If the age entered is a non-number, then the exception block is executed.

PROGRAM:

```
def input_number(min,max):
    try:
        while True:
            n=int(input("Please enter a number between {} and {} :".format(min,max)))
            n=int(n)
            if(min<=n<=max):
                print(n)
               break
            else:
                print("Above 100, wrong")
                break
    except:
        print("mark must be a valid number")
input_number(1,100)
```

OUTPUT 1:

Please enter a number between 1 and 100 :4

4

OUTPUT 2:

Please enter a number between 1 and 100 :156

Above 100, wrong

OUTPUT 3:

Please enter a number between 1 and 100 : ab

mark must be a valid number

RESULT:

Thus the program for real-time / technical applications using Exception handling was implemented and executed successfully.

Ex. No. 11

Exploring Pygame tool

- Pygame is a cross-platform set of Python modules which is used to create video games.
- It consists of computer graphics and sound libraries designed to be used with the Python programming language.
- Pygame was officially written by Pete Shinners to replace PySDL.
- Pygame is suitable to create client-side applications that can be potentially wrapped in a standalone executable.

Pygame Installation

Install pygame in Windows

Before installing Pygame, Python should be installed in the system, and it is good to have 3.6.1 or above version because it is much friendlier to beginners, and additionally runs faster.

There are mainly two ways to install Pygame, which are given below:

1. Installing through pip: The good way to install Pygame is with the pip tool. The command is the following:

py -m pip install -U pygame --user

2. Installing through an IDE: The second way is to install it through an IDE and here we are using Pycharm IDE. Installation of pygame in the pycharm is straightforward. We can install it by running the above command in the terminal or use the following steps:

Open the File tab and click on the Settings option.

Select the Project Interpreter and click on the + icon.

It will display the search box. Search the pygame and click on the install package button.

To check whether the pygame is properly installed or not, in the IDLE interpreter, type the following command and press Enter:

```
import pygame
```

If the command runs successfully without throwing any errors, it means we have successfully installed Pygame and found the correct version of IDLE to use for pygame programming.

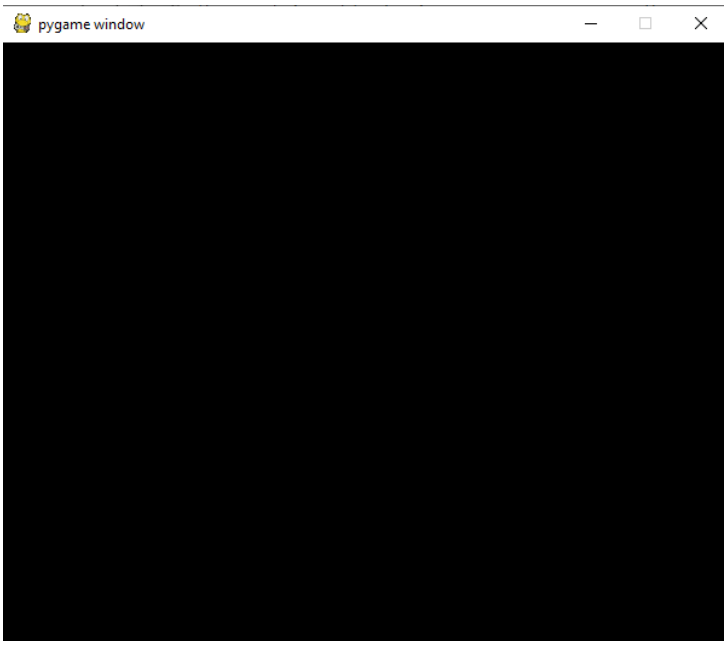
Simple pygame Example

```
import pygame

pygame.init()
screen = pygame.display.set_mode((400,500))
done = False

while not done:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
```

```
        done = True
    pygame.display.flip()
```



import pygame - This provides access to the pygame framework and imports all functions of pygame.

pygame.init() - This is used to initialize all the required module of the pygame.

pygame.display.set_mode((width, height)) - This is used to display a window of the desired size. The return value is a Surface object which is the object where we will perform graphical operations.

pygame.event.get()- This is used to empty the event queue.

pygame.QUIT - This is used to terminate the event when we click on the close button at the corner of the window.

pygame.display.flip() - Pygame is double-buffered, so this shifts the buffers. It is essential to call this function in order to make any updates that you make on the game screen to make visible.

Pygame Surface

The pygame Surface is used to display any image. The Surface has a pre-defined resolution and pixel format. The Surface color is by default black. Its size is defined by passing the size argument.

Pygame Clock

Times are represented in millisecond (1/1000 seconds) in pygame. Pygame clock is used to track the time. The time is essential to create motion, play a sound, or, react to any event.

Developing a game activity using Pygame

12. A. Bouncing ball game

Aim:

Write a python program to simulate bouncing ball using pygame

Algorithm:

1. Import pygamemodule
2. Call pygame.init() to initiate all imported pygamemodule
3. Set the screen size in terms of pixels using pygame.display.set_mode((400,300)
4. If there is any event in pygamequeue
    a. Get the event from the pygamequeue
    b. If event types is pygame.QUIT then setdone=true
5. Else, Draw the circle update the screen display with new circle to bring bouncingeffect
6. Call sys.exit() to uninitialized all the pygamemodules

Program:

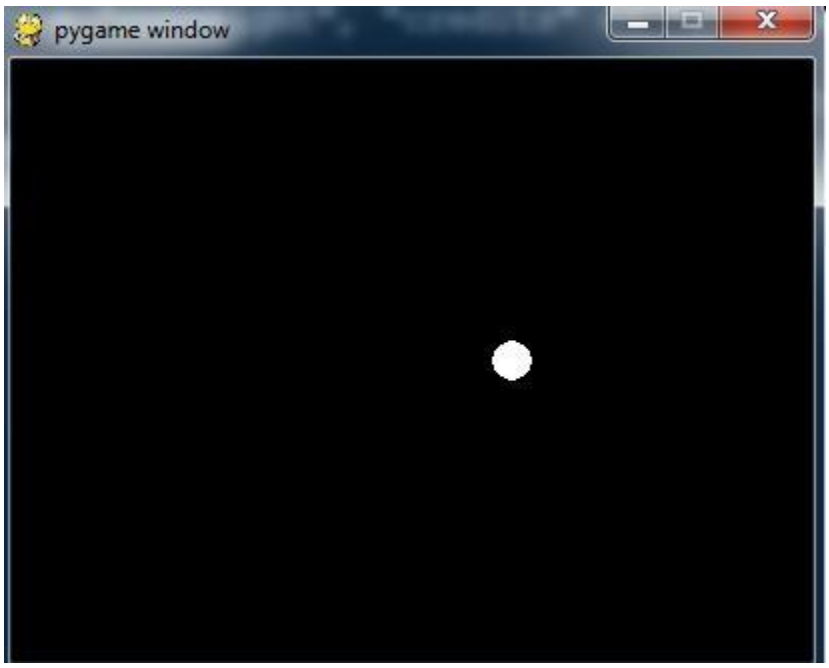
```
import pygame
from pygame.locals import *

pygame.init()
screen =
pygame.display.set_mode((40
0, 300)) done = False

while not done:
    for event in pygame.event.get():
        if event.type ==
            pygame.QUI
            T: done =
            True
```

```
pygame.draw.circle(screen, (255,255,255), [100, 80], 10,
0) pygame.display.update()
pygame.draw.circle(screen, (0,0,0), [100, 80],
10, 0) pygame.display.update()
pygame.draw.circle(screen, (255,255,255), [150, 95], 10,
0) pygame.display.update()
pygame.draw.circle(screen, (0,0,0), [150, 95], 10, 0)
pygame.display.update()
pygame.draw.circle(screen, (255,255,255), [200, 130], 10, 0)
pygame.display.update()pygame.draw.circle(screen, (0,0,0),
[200, 130], 10, 0) pygame.display.update()
pygame.draw.circle(screen, (255,255,255), [250, 150], 10, 0)
pygame.display.update()
pygame.display.update()
for event in
   pygame.event.get():
    if event.type ==
   QUIT:
       pygame.quit() sys.exit()
```

Output:



Result:

Thus, the program to simulate elliptical orbits in pygame has been executed successfully.

12. B. Car Game

Aim:

Write a python program to simulate car game using pygame.

Algorithm:

1. Import pygamemodule
2. Call pygame.init() to initiate all imported pygamemodule
3. Set the screen size in terms of pixels using pygame.display.set_mode((798,600
4. Set the logo and the background for the game.
5. Assign the car images for the race and set its parameters
6. Set the key movement values
7. Update the screen display after every change in car position
8. Call the game loop until run
9. Stop

Program:

```
import pygame
pygame.init()
from pygame.locals import*
import random
screen = pygame.display.set_mode((798,600))
pygame.display.set_caption('Racing Beast')
logo = pygame.image.load('car game/logo.jpeg')
pygame.display.set_icon(logo)
def gameloop():
    bg = pygame.image.load('car game/bg.png')
```

```
maincar = pygame.image.load('car game\car.png')
maincarX = 350
maincarY = 495
maincarX_change = 0
maincarY_change = 0

car1 = pygame.image.load('car game\car1.jpeg')
car1X = random.randint(178,490)
car1Y = 100

car2 = pygame.image.load('car game\car2.png')
car2X = random.randint(178,490)
car2Y = 100
car3 = pygame.image.load('car game\car3.png')
car3X = random.randint(178,490)
car3Y = 100
run = True
while run:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_RIGHT:
                maincarX_change += 5
            if event.key == pygame.K_LEFT:
                maincarX_change -= 5
            if event.key == pygame.K_UP:
                maincarY_change -= 5
```

```
        if event.key == pygame.K_DOWN:
            maincarY_change += 5

    if event.type == pygame.KEYUP:
        if event.key == pygame.K_RIGHT:
            maincarX_change = 0

        if event.key == pygame.K_LEFT:
            maincarX_change = 0

        if event.key == pygame.K_UP:
            maincarY_change = 0

        if event.key == pygame.K_DOWN:
            maincarY_change = 0

if maincarX < 178:
    maincarX = 178
if maincarX > 490:
    maincarX = 490

if maincarY < 0:
    maincarY = 0
if maincarY > 495:
    maincarY = 495


screen.fill((0,0,0))

#displaying the background image
```

```
        screen.blit(bg,(0,0))

        #displaying our main car
        screen.blit(maincar,(maincarX,maincarY))

        #displaing other cars
        screen.blit(car1,(car1X,car1Y))
        screen.blit(car2,(car2X,car2Y))
        screen.blit(car3,(car3X,car3Y))

        #updating the values
        maincarX += maincarX_change
        maincarY +=maincarY_change

        car1Y += 10
        car2Y += 10
        car3Y += 10



        if car1Y > 670:
            car1Y = -100
        if car2Y > 670:
            car2Y = -150
        if car3Y > 670:
            car3Y = -200



        pygame.display.update()
gameloop()
```

Output:

Result:

Thus the game was developed using pygame module.