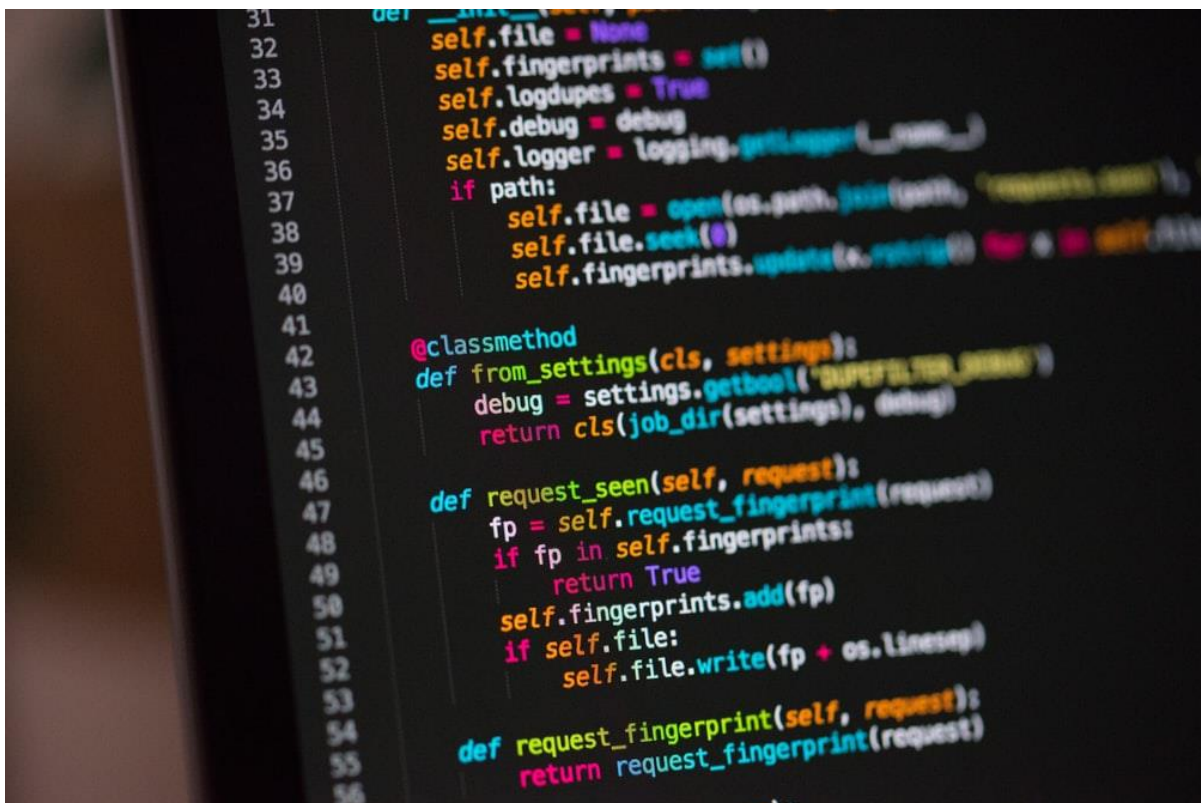




**SYMBIOSIS INSTITUTE OF TECHNOLOGY (SIT)**

# **Programming and Problem Solving – Lab**



## **Experiments Record - Journal**

**~ Mudit Garg**

**CSE B2**

**Batch 2021-25**

**PRN: 21070122098**

# Experiment No. 1-A

## Title:

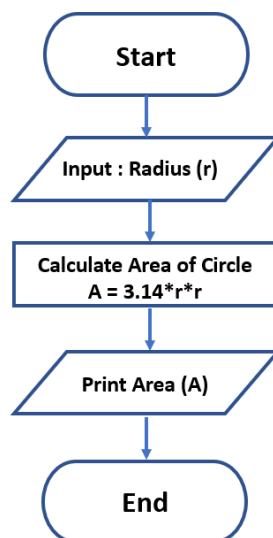
To design and develop a flowchart and an algorithm for finding the area of a circle, of a rectangle, of a square and of a triangle.

## Tool/Platform:

Microsoft Word / PowerPoint

## Flowcharts and Algorithms:

### Flowchart – Area of a Circle

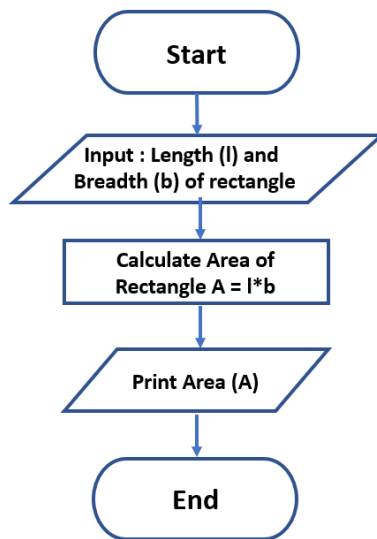


13/10/2021

Algorithm for Area of Circle:

1. Initialise the program
2. Initialise variable “r” and assign it the value as entered by user for radius of the circle
3. Initialise variable “a”
4. Calculate the area of circle using the formula, **area = 3.14\*r\*r** and assign it to the variable “a”
5. Print the area of circle, “a” as the output
6. End the program

## Flowchart – Area of a Rectangle

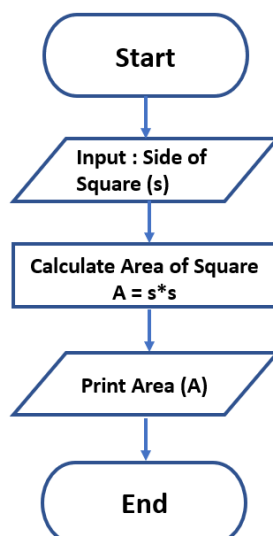


13/10/2021

Algorithm for Area of Rectangle:

1. Initialise the program
2. Initialise variable “l” and “b” and assign them a value as entered by user for length and breadth of the rectangle
3. Initialise variable “a”
4. Calculate the area of rectangle using the formula, **area = l\*b** and assign it to the variable “a”
5. Print the area of rectangle, “a” as the output
6. End the program

## Flowchart – Area of a Square

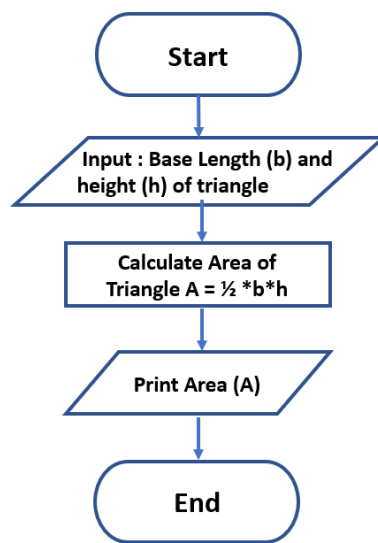


13/10/2021

Algorithm for Area of Square:

1. Initialise the program
2. Initialise variable “s” and assign it the value as entered by user for side of the square
3. Initialise variable “a”
4. Calculate the area of square using the formula, **area = s\*s** and assign it to the variable “a”
5. Print the area of square, “a” as the output
6. End the program

## Flowchart – Area of a Triangle



13/10/2021

Algorithm for Area of Triangle:

1. Initialise the program
2. Initialise variable “b” and “h” and assign it the value as entered by user for the base length and height of the triangle
3. Initialise variable “a”
4. Calculate the area of triangle using the formula, **area = 1/2\*b\*h** and assign it to the variable “a”
5. Print the area of triangle, “a” as the output
6. End the program

## Learning Outcome:

By performing this experiment, I was able to understand the concept of Computational Thinking and implement it. I was able to understand that why algorithms and flowcharts are important in solving a problem and how they help in achieving a better result faster.

## Experiment No. 1-B

### Title:

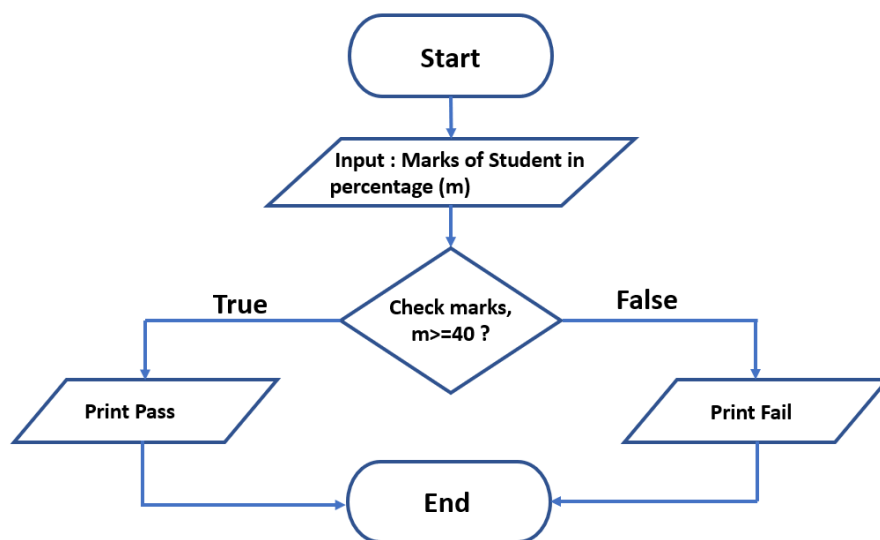
To design and develop a flowchart and an algorithm to determine whether a student has passed the exam or not

### Tool/Platform:

Microsoft Word / PowerPoint

### Flowchart:

#### Flowchart – Student Examination



13/10/2021

### Algorithm:

1. Initialise the program
2. Initialise variable “m” and assign it the value as entered by user for marks obtained out of 100
3. Check if marks are greater than 40 or not
4. If marks are greater than 40, Print “**Pass**” as the output
5. Else print “**Fail**” as the output
6. End the Program

### Learning Outcome:

By performing this experiment, I was able to understand the concept of Computational Thinking and implement it. I was able to understand that why algorithms and flowcharts are important in solving a problem and how they help in achieving a better result faster.

## Experiment No. 2

### Title:

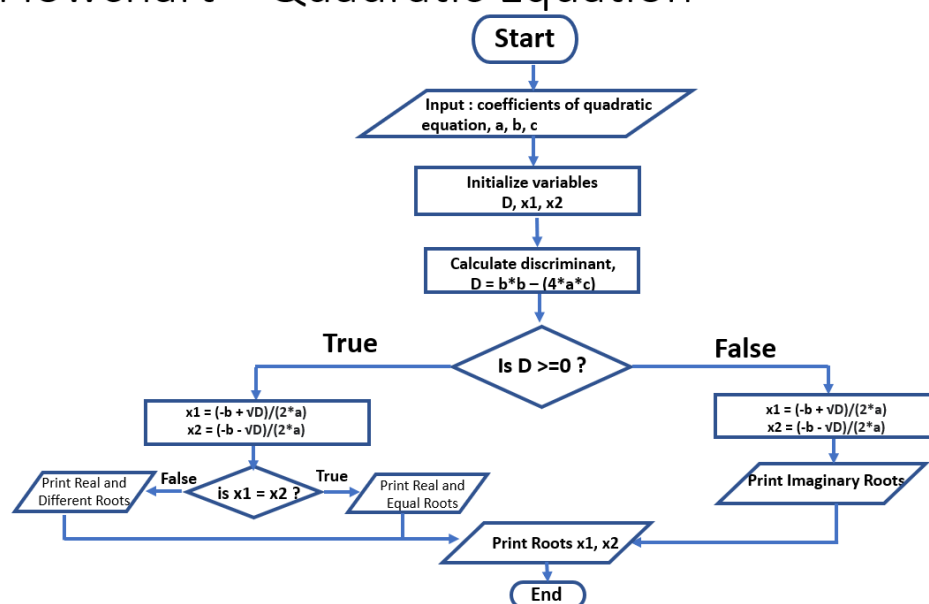
To design and develop a flowchart and an algorithm that takes three coefficients  $a$ ,  $b$  and  $c$  of a quadratic equation  $ax^2 + bx + c = 0$  as input and compute all possible roots of the equation

### Tool/Platform:

Microsoft Word / PowerPoint

### Flowchart:

#### Flowchart – Quadratic Equation



20/10/2021

### Algorithm:

1. Initialize Program
2. Initialize variables " $a$ ", " $b$ ", " $c$ " and assign the value as per input from user for coefficients of quadratic equation
3. Initialize variables " $D$ ", " $x1$ ", " $x2$ "
4. Calculate discriminant as  $b*b - (4*a*c)$  and assign to " $D$ "
5. If  $D \geq 0$ , calculate roots as " $x1 = \frac{-b + \sqrt{D}}{2*a}$ ", " $x2 = \frac{-b - \sqrt{D}}{2*a}$ "
6. If  $x1 = x2$ , print "Real and Equal Roots"
7. Else, if  $D < 0$ , calculate roots as  $x1 = \frac{-b + \sqrt{D}}{2*a}$ ,  $x2 = \frac{-b - \sqrt{D}}{2*a}$ , Print "Imaginary/ Complex Roots"
8. Print Roots " $x1$ ", " $x2$ "

**Learning Outcome:**

By performing this experiment, I was able to understand the concept of Computational Thinking and implement it. I was able to understand that why algorithms and flowcharts are important in solving a problem and how they help in achieving a better result faster and how to break a complex problem into simpler parts and solve to get the answer easily and efficiently.

## Experiment No. 3-A

### Title:

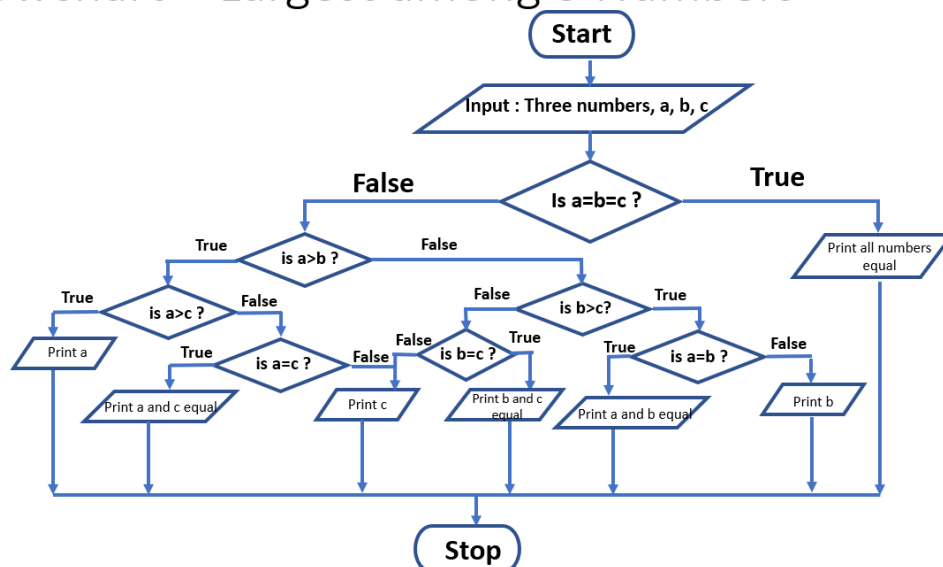
To design and develop a flowchart and an algorithm that takes three numbers a, b and c as input and prints the value of the largest number

### Tool/Platform:

Microsoft Word / PowerPoint

### Flowchart:

Flowchart – Largest among 3 Numbers



27/10/2021

### Algorithm:

1. Initialize Program
2. Initialize variables “a”, “b”, “c” and assign the value as per input from user for three numbers
3. Check if the all the numbers are **equal or not**
4. If all the numbers are equal, print “**All numbers equal**”, and their **value**
5. Else, check if **a>b**?
6. If a>b check for **a>c**
7. If a>c, print “a” as the output, else check for **a=c**
8. If a=c, print “**a and c equal**”, and their **value**, else print “c”
9. If a>b is false, check for **b>c**
10. If b>c is false, check for **b=c** and print “**b and c equal**” and their **value if true** or else print “c”
11. If b>c is true check for **b=a**
12. If b=a, print “**a and b equal**”, and their value, else print “b”
13. Stop the program



**Learning Outcome:**

By performing this experiment, I was able to understand the concept of Computational Thinking and implement it. I was able to understand that why algorithms and flowcharts are important in solving a problem and how they help in achieving a better result faster and how to break a complex problem into simpler parts and solve to get the answer easily and efficiently.

## Experiment No. 3-B

### Title:

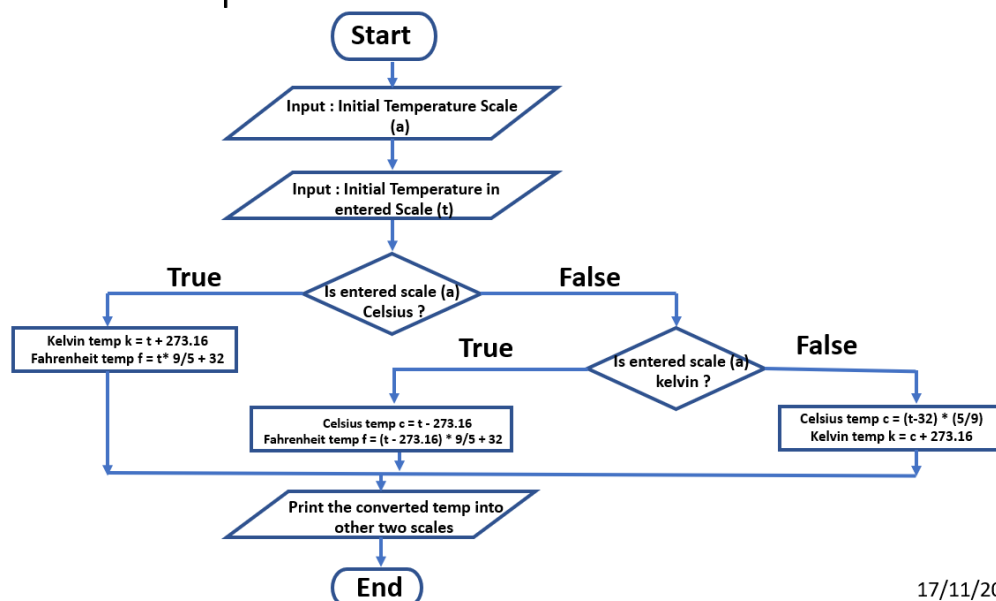
Write a Python Program to convert the entered temperature in Celsius into other two temperature scales.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Temperature Conversion



17/11/2021

### Algorithm:

1. Initialise the program
2. Initialise variable "a" and "t"
3. Input Initial Temperature Scale from User and assign it to "a"
4. Input Temperature and assign it to "t"
5. Check if entered scale "a" is Celsius
6. If yes, calculate temperature in Fahrenheit using  $f = t * (9/5) + 32$  and in Kelvin using  $t + 273.16$
7. Else if false, check for Scale being Kelvin
8. If true, calculate temperature in Fahrenheit using  $f = (t - 273.16) * (9/5) + 32$  and in Celsius using  $t - 273.16$

9. If the entered scale is not Kelvin, it is Fahrenheit
10. Calculate the temperature in Celsius as  $c = (t - 32) * (5/9)$  in Kelvin as  $k = c + 273.16$
11. Print the converted temperature with scales
12. End the program

### **Source Code:**

```
# Temperature Conversion
# Date 17/11/21
import sys

print("Temperature Conversion from One to another scale")

a = input("Select Default Temperature Scale from Celsius / Fahrenheit / Kelvin: ")
if a not in ["Celsius", "celsius", "Fahrenheit", "fahrenheit", "Kelvin", "kelvin"]:
    sys.exit("Invalid Input, Recheck !")

print("You have selected", a, "as your default temperature measurement scale!")
t = eval(input(" Enter the temperature in Selected Scale : "))

if a == "Celsius" or a == "celsius":
    k = t + 273.16
    f = t*(9/5) + 32
    print(f' The entered temperature {t} converted from {a} to Kelvin is', round(k,2),
          and in Fahrenheit is", round(f,2),' !')

elif a == "Fahrenheit" or a == "fahrenheit":
    c = (t - 32) * (5/9)
    k = c + 273.16
    print(f' The entered temperature {t} converted from {a} to Kelvin is', round(k,2),
          and in Celsius is", round(c,2),' !')

elif a == "Kelvin" or a == "kelvin":
    c = t - 273.16
    f = (t - 273.16) * (9/5) + 32
    print(f' The entered temperature {t} converted from {a} to Celsius is', round(c,2),
          and in Fahrenheit is", round(f,2),' !')

else:
    print("Invalid Operations, Recheck !")
```

```

# Temperature Conversion
# Date 17/11/21
import sys

print("Temperature Conversion from One to another scale")

a = input("Select Default Temperature Scale from Celsius / Fahrenheit / Kelvin: ")
if a not in ["Celsius", "celsius", "Fahrenheit", "fahrenheit", "Kelvin", "kelvin"]:
    sys.exit("Invalid Input, Recheck !")

print("You have selected", a, "as your default temperature measurement scale!")
t = eval(input(" Enter the temperature in Selected Scale : "))

if a == "Celsius" or a == "celsius":
    k = t + 273.16
    f = t*(9/5) + 32
    print(f' The entered temperature {t} converted from {a} to Kelvin is', round(k,2)," and in Fahrenheit is", round(f,2),' !')

elif a == "Fahrenheit" or a == "fahrenheit":
    c = (t - 32) * (5/9)
    k = c + 273.16
    print(f' The entered temperature {t} converted from {a} to Kelvin is', round(k,2)," and in Celsius is", round(c,2),' !')

elif a == "Kelvin" or a == "kelvin":
    c = t - 273.16
    f = (t - 273.16) * (9/5) + 32
    print(f' The entered temperature {t} converted from {a} to Celsius is', round(c,2)," and in Fahrenheit is", round(f,2),' !')

else:
    print("Invalid Operations, Recheck !")

```

## Output Screenshot:

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\3. Temperature Conversion.py
Temperature Conversion from One to another scale
Select Default Temperature Scale from Celsius / Fahrenheit / Kelvin: Kelvin
You have selected Kelvin as your default temperature measurement scale!
Enter the temperature in Selected Scale : 300
The entered temperature 300 converted from Kelvin to Celsius is 26.84 and in Fahrenheit is 80.31 !
>>>|

```

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\3. Temperature Conversion.py
Temperature Conversion from One to another scale
Select Default Temperature Scale from Celsius / Fahrenheit / Kelvin: celsius
You have selected celsius as your default temperature measurement scale!
Enter the temperature in Selected Scale : -40
The entered temperature -40 converted from celsius to Kelvin is 233.16 and in Fahrenheit is -40.0 !
>>>|

```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\3. Temperature Conversion.py
Temperature Conversion from One to another scale
Select Default Temperature Scale from Celsius / Fahrenheit / Kelvin: fahrenheit
You have selected fahrenheit as your default temperature measurement scale!
Enter the temperature in Selected Scale : 104
The entered temperature 104 converted from fahrenheit to Kelvin is 313.16 and in Celsius is 40.0 !
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\3. Temperature Conversion.py
Temperature Conversion from One to another scale
Select Default Temperature Scale from Celsius / Fahrenheit / Kelvin: cel
SystemExit: Invalid Input, Recheck !
>>>
```

## **Learning Outcome:**

By performing this experiment, I was able to understand the basics concepts and syntax of Python Programming Language and write a code in it. The use of built-in functions and loop was clear. The basic data types in Python were understood and used to get appropriate result. Functions, Control flow statements and different data types were used to represent data, process it and give the required output. Use of if-else or if-elif-else loop or the nested if-else loop was understood and the syntax was clear. Its function was understood and it was used to get the desired result.

## Experiment No. 4

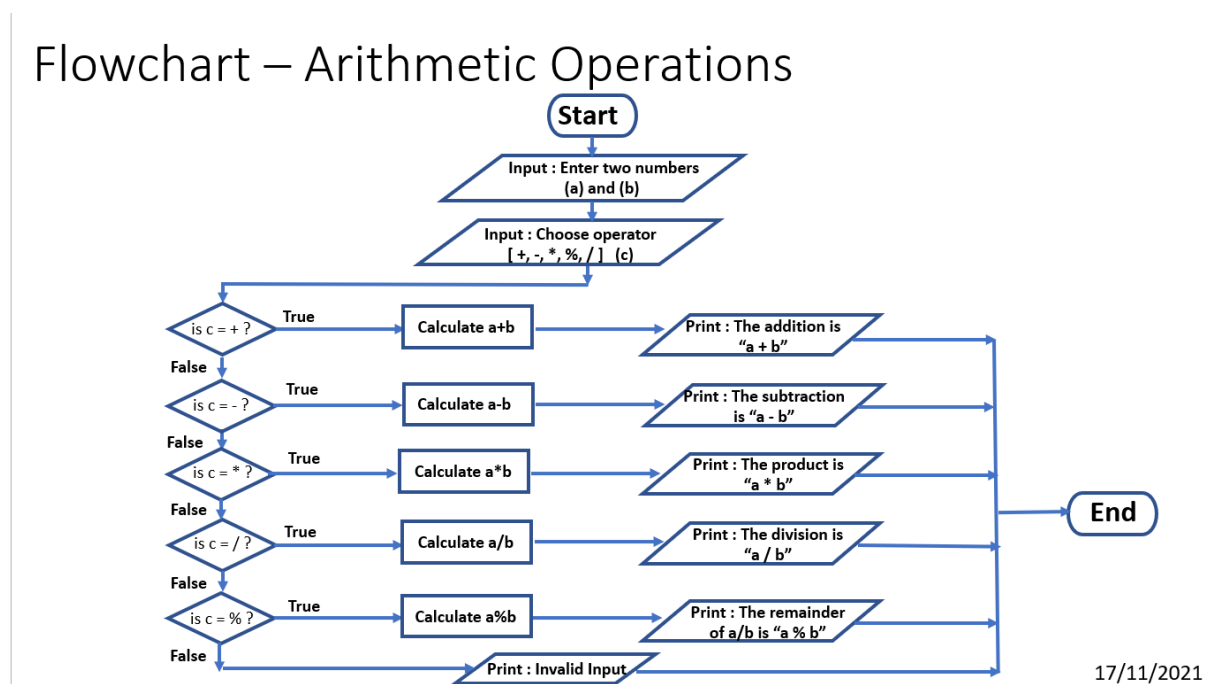
### Title:

Write a Python Program to perform different arithmetic operations like in mathematics.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:



### Algorithm:

1. Initialise the program
2. Initialise variable "**a**", "**b**" and "**c**"
3. Input two numbers from user and assign them to "**a**" and "**b**"
4. Input the Operation to be performed and assign it to "**c**"
5. Check if "**c**" is "+" and if true perform the operation "**a+b**" and print the output
6. If false, check if "**c**" is "-" and if this is true perform the operation "**a-b**" and print the output
7. Else, check if "**c**" is "\*" and if this is true perform the operation "**a\*b**" and print the output
8. If false, check if "**c**" is "/" and if this is true perform the operation "**a/b**" and print the output
9. Else, check if "**c**" is "%" and if this is true perform the operation "**a%b**" and print the output
10. Else, if false, Print "**Invalid Operation**"
11. End the program

## Source Code:

```
#if - else condition to perform simple arithmetic functions for entered two values
#Date 17/11/2021
a=eval(input("Enter a number: "))
b=eval(input("Enter second number with which to perform the simple arithmetic
operation: "))
c=input("Choose an operator [ + , - , / , * , % [To get remainder when a is divided by
b] ")
if c == '+' :
    print("The addition of two entered numbers is : " , a+b)
elif c == '-' :
    print("The subtraction of two entered numbers is : " , a-b)
elif c == '*' :
    print("The multiplication of two entered numbers is : " , a*b)
elif c == '/' :
    print("The division of two entered numbers is : " , a / b)
elif c == '%' :
    print("The remainder when a is divided by b is:" , a%b)
else:
    print("Entered Operator is INVALID !!!")
```

```
#if - else condition to perform simple arithmetic functions for entered two values
#Date 17/11/2021
a=eval(input("Enter a number: "))
b=eval(input("Enter second number with which to perform the simple arithmetic operation: "))
c=input("Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b] ")
if c == '+' :
    print("The addition of two entered numbers is : " , a+b)
elif c == '-' :
    print("The subtraction of two entered numbers is : " , a-b)
elif c == '*' :
    print("The multiplication of two entered numbers is : " , a*b)
elif c == '/' :
    print("The division of two entered numbers is : " , a / b)
elif c == '%' :
    print("The remainder when a is divided by b is:" , a%b)
else:
    print("Entered Operator is INVALID !!!")
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab1. WAP to perform all mathematical functions.py
Enter a number: 12
Enter second number with which to perform the simple arithmetic operation: 2
Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b] +
The addition of two entered numbers is : 14
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\1. WAP to perform all mathematical functions.py
Enter a number: 14
Enter second number with which to perform the simple arithmetic operation: 37
Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b ] -
The subtraction of two entered numbers is : -23
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\1. WAP to perform all mathematical functions.py
Enter a number: 5
Enter second number with which to perform the simple arithmetic operation: 13
Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b ] *
The multiplication of two entered numbers is : 65
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\1. WAP to perform all mathematical functions.py
Enter a number: 60
Enter second number with which to perform the simple arithmetic operation: 5
Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b ] /
The division of two entered numbers is : 12.0
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\1. WAP to perform all mathematical functions.py
Enter a number: 100
Enter second number with which to perform the simple arithmetic operation: 3
Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b ] %
The remainder when a is divided by b is: 1
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\1. WAP to perform all mathematical functions.py
Enter a number: 12
Enter second number with which to perform the simple arithmetic operation: 34
Choose an operator [ + , - , / , * , % [To get remainder when a is divided by b ] .
Entered Operator is INVALID !!!
>>>
```

## Learning Outcome:

By performing this experiment, I was able to understand the basics concepts and syntax of Python Programming Language and write a code in it. The use of built-in functions and loop was understood. The basic data types in Python were used to get appropriate result. Use of in-built arithmetic operators was understood. Inbuilt arithmetic functions and if-else loop were used to get required result.



## Experiment No. 5-A

### Title:

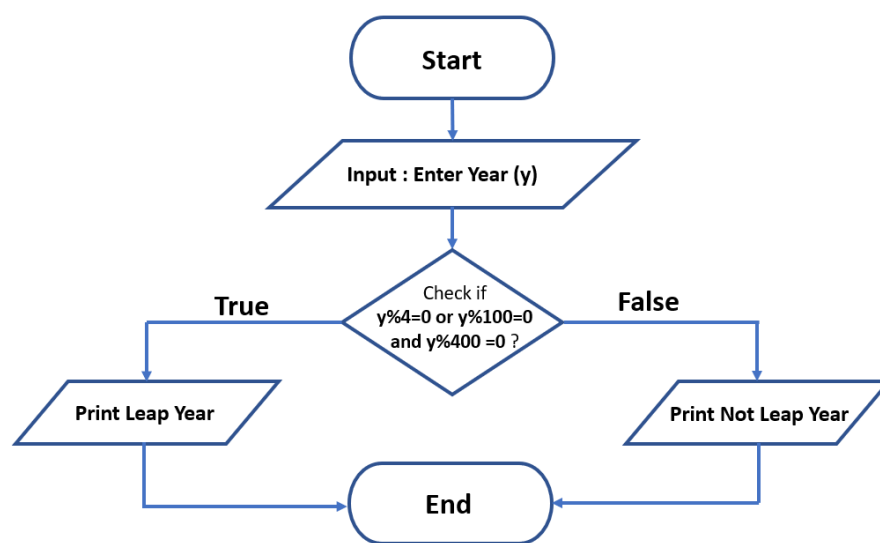
Write a Python Program to check if the given year is a leap year.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

#### Flowchart – Check if Leap Year



17/11/2021

### Algorithm:

1. Initialise the program
2. Initialise variable "y"
3. Input value of Year from user and assign it to "y"
4. Check if **y%4=0 or y%100=0 and y%400=0**
5. If True, Print "Leap Year"
6. Else, Print "Not a Leap Year"

### Source Code:

```
#Check if leap year or not  
# Date 17/11/21
```

```
year = int(input("Please Enter the Year Number you wish to check: "))
```

```
if (year%4 == 0) or (year%100 == 0) and (year%400 == 0):
```

```
    print(f' Entered year {year} is a Leap Year')
```

```
else:
```

```
    print(f' Entered year {year} is not a Leap Year')
```

```
#Check if leap year or not
# Date 17/11/21

year = int(input("Please Enter the Year Number you wish to check: "))

if (year%4 == 0) or (year%100 == 0) and (year%400 == 0):
    print(f'Entered year {year} is a Leap Year')
else:
    print(f'Entered year {year} is not a Leap Year')
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\4. Check if leap year or not.py
Please Enter the Year Number you wish to check: 2020
Entered year 2020 is a Leap Year
>>>|
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\4. Check if leap year or not.py
Please Enter the Year Number you wish to check: 2021
Entered year 2021 is not a Leap Year
>>>|
```

## Learning Outcome:

By performing this experiment, I was able to understand the basics concepts and syntax of Python Programming Language and write a code in it. The basic data types in Python were used to get appropriate result. Use of if-else loop was understood. Use of Arithmetic Operators and Logical Operators was made clear. Syntax of “f- string formatting” was understood and it was used to get required result.

## Experiment No. 5-B

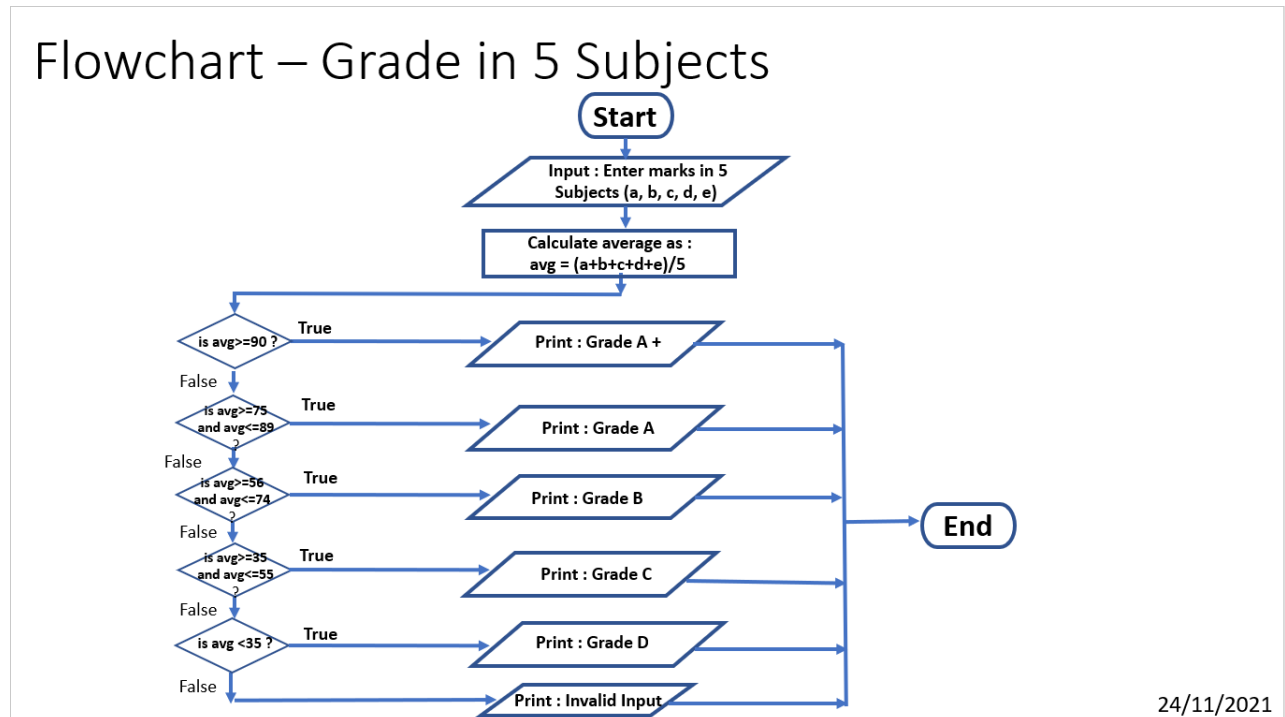
### Title:

Write a Python Program to input Marks in 5 Subjects and Display the Grade.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:



### Algorithm:

1. Initialise the program
2. Initialise variable “a”, “b”, “c”, “d”, “e” and “avg”
3. Input marks in 5 subjects from user and assign them to “a”, “b”, “c”, “d”, “e”
4. Calculate Average and assign it to “avg”
5. Check if “avg” is “>=90” and if true print “**Grade A+**”
6. If false, check if “avg” is “>=75 and <=89” and if this is true print “**Grade A**”
7. Else, check if “avg” is “>=56 and <=74” and if this is true print “**Grade B**”
8. If false, check if “avg” is “>=35 and <=55” and if this is true print “**Grade C**”
9. Else, check if “avg” is “<35” and if this is true print “**Grade D**”
10. Else, if false, Print “**Invalid Operation**”
11. Exit

## Source Code:

```
# Grade based on marks in 5 subjects
# Date : 24/11/2021
import sys

s1= float(input("Enter Marks Scored in Mathematics: "))
s2= float(input("Enter Marks Scored in Science: "))
s3= float(input("Enter Marks Scored in English: "))
s4= float(input("Enter Marks Scored in Hindi: "))
s5= float(input("Enter Marks Scored in Social Science: "))
print()
avg= (s1 + s2 + s3 + s4 + s5)/5
print("Scored Percentage:", avg)
if avg >= 90 :
    print(" Outstanding : Grade A* ")
elif avg>= 75 and avg <=89:
    print(" Excellent : Grade A ")
elif avg>= 56 and avg <=74:
    print(" Very Good : Grade B ")
elif avg>= 35 and avg <=55:
    print(" Good : Grade C ")
elif avg <35:
    print(" Scope for Improvement : Grade D ")
else:
    sys.exit("Invalid Input, Recheck !")
```

```
# Grade based on marks in 5 subjects
# Date : 24/11/2021
import sys

s1= float(input("Enter Marks Scored in Mathematics: "))
s2= float(input("Enter Marks Scored in Science: "))
s3= float(input("Enter Marks Scored in English: "))
s4= float(input("Enter Marks Scored in Hindi: "))
s5= float(input("Enter Marks Scored in Social Science: "))
print()
avg= (s1 + s2 + s3 + s4 + s5)/5
print("Scored Percentage:", avg)
if avg >= 90 :
    print(" Outstanding : Grade A* ")
elif avg>= 75 and avg <=89:
    print(" Excellent : Grade A ")
elif avg>= 56 and avg <=74:
    print(" Very Good : Grade B ")
elif avg>= 35 and avg <=55:
    print(" Good : Grade C ")
elif avg <35:
    print(" Scope for Improvement : Grade D ")
else:
    sys.exit("Invalid Input, Recheck !")
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\6. Marks in 5 subjects.py
Enter Marks Scored in Mathematics: 100
Enter Marks Scored in Science: 95
Enter Marks Scored in English: 96
Enter Marks Scored in Hindi: 90
Enter Marks Scored in Social Science: 99

Scored Percentage: 96.0
Outstanding : Grade A*
```

## Learning Outcome:

By performing this experiment, use of if-elif-else loop was understood. Use of Arithmetic Operators and Logical Operators was made clear. Appropriate data types were used to perform required operations and thus get required result.

## Experiment No. 6-A

### Title:

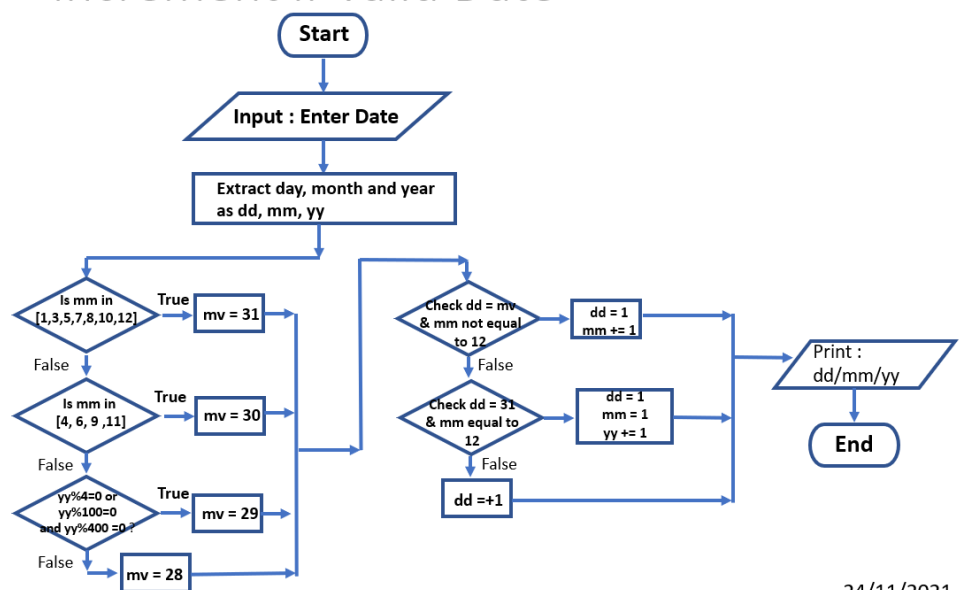
Write a Python Program to check if a date is valid and print the incremented date if it is.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Increment if Valid Date



24/11/2021

### Algorithm:

1. Initialise the program
2. Initialise variable “**date**”, “**dd**”, “**mm**”, “**yy**”, “**mv**”
3. Input date from user and assign to “**date**”
4. Extract day, month and year and assign to “**dd**”, “**mm**”, “**yy**”
5. Check for maximum number of days in given month and assign to “**mv**”
6. If “**dd**” is equal to “**mv**” and “**mm**” is not equal to 12, set “**dd**” as 1 and increment value of “**mm**” by 1
7. Else, check if “**dd**” is equal to 31 and “**mm**” is equal to 12, if true, set “**dd**” = 1, “**mm**” = 1 and increment value of “**yy**” by 1
8. If false, increment value of “**dd**” by 1
9. Print the incremented date
10. Exit

## **Source Code:**

#Check if entered date is in valid form

#Print Incremented Date

#Date : 24/11/2021

from datetime import \*

idate=input("Enter date in the format dd/mm/yyyy: ")

dd,mm,yyyy=idate.split('/')

dd=int(dd)

mm=int(mm)

yyyy=int/yyyy)

if mm in [1,3,5,7,8,10,12]:

    mv= 31

elif mm in [4,6,9,11]:

    mv= 30

elif (yyyy%4==0 and yyyy%100!=0 or yyyy%400==0):

    mv= 29

else:

    mv= 28

if mm < 1 or mm > 12 or dd < 1 or dd > mv :

    sys.exit("Invalid Date Entered, Recheck !")

elif dd==mv and mm!=12:

    dd= 1

    mm+= 1

elif dd== 31 and mm == 12:

    dd = 1

    mm = 1

    yyyy +=1

else:

    dd +=1

odate= date/yyyy,mm,dd)

fdate=odate.strftime("%d/%m/%Y")

print("Entered Date:" , idate)

print("Incremented Date: ",fdate)

```

#Check if entered date is in valid form
#Print Incremented Date
#Date : 24/11/2021

from datetime import *

idate=input("Enter date in the format dd/mm/yyyy: ")
dd,mm,yyyy=idate.split("/")
dd=int(dd)
mm=int(mm)
yyyy=int(yyyy)
if mm in [1,3,5,7,8,10,12]:
    mv= 31
elif mm in [4,6,9,11]:
    mv= 30
elif (yyyy%4==0 and yyyy%100!=0 or yyyy%400==0):
    mv= 29
else:
    mv= 28

if mm < 1 or mm > 12 or dd < 1 or dd > mv :
    sys.exit("Invalid Date Entered, Recheck !")
elif dd==mv and mm!=12:
    dd= 1
    mm+= 1
elif dd== 31 and mm == 12:
    dd = 1
    mm = 1
    yyyy +=1
else:
    dd +=1

odate= date(yyyy,mm,dd)
fdate=odate.strftime("%d/%m/%Y")
print("Entered Date:" , idate)
print("Incremented Date: ",fdate)

```

## Output Screenshots:

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\7. if date valid print incremented.py
Enter date in the format dd/mm/yyyy: 27/5/2021
Entered Date: 27/5/2021
Incremented Date: 28/05/2021
>>>

```

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\7. if date valid print incremented.py
Enter date in the format dd/mm/yyyy: 31/12/2021
Entered Date: 31/12/2021
Incremented Date: 01/01/2022
>>>

```



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\7. if date valid print incremented.py
Enter date in the format dd/mm/yyyy: 29/2/2020
Entered Date: 29/2/2020
Incremented Date: 01/03/2020
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\7. if date valid print incremented.py
Enter date in the format dd/mm/yyyy: 29/2/2019
SystemExit: Invalid Date Entered, Recheck !
>>>
```

## **Learning Outcome:**

By performing this experiment, use of if-elif-else loop was understood. Use of Arithmetic Operators and Logical Operators was made clear. Appropriate data types were used to perform required operations and thus get required result. Use of appropriate assignment operators helped in achieving required result. This program helped in understanding the topics such as control flow statements, if-elif-else loop, use of inbuilt modules and functions, and in developing required logic for problem solving.

## Experiment No. 6-B

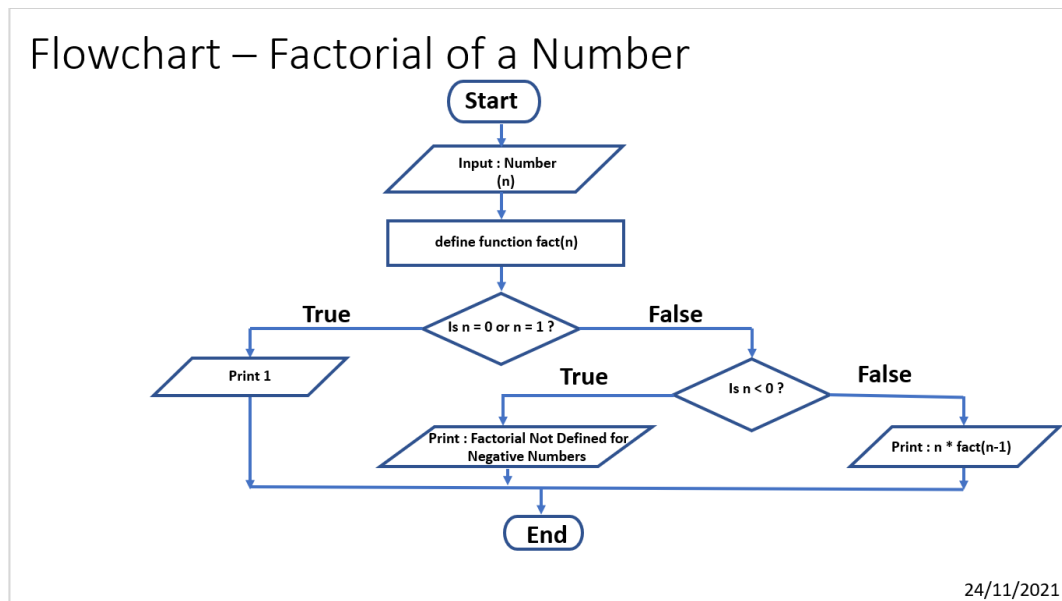
### Title:

Write a Python Program to find the factorial of a number

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:



### Algorithm:

1. Initialise the program
2. Initialise variable “n”
3. Define a function **fact(n)** (which is recursive in nature)
4. Check if input number n is 1 or if it is 0
5. If true, print 1
6. If false, check if  $n < 0$
7. If this is true, Print “**Factorial Not Defined for Negative Numbers**”
8. Else, return  $n * \text{fact}(n-1)$

## Source Code:

```
#Factorial of Number
# Date 24/11/2021
def fact(n):
    if n==1:
        return n
    else:
        return n * fact(n-1)

n=int(input("Enter Number: "))
if n<0:
    print("Factorial is not defined for Negative Numbers!")
elif n==0:
    print("Factorial = 1")
else:
    print("The factorial is:", fact(n))
```

```
#Factorial of Number
# Date 24/11/2021
def fact(n):
    if n==1:
        return n
    else:
        return n * fact(n-1)

n=int(input("Enter Number: "))
if n<0:
    print("Factorial is not defined for Negative Numbers!")
elif n==0:
    print("Factorial = 1")
else:
    print("The factorial is:", fact(n))
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\5. Factorial of Number.py
Enter Number: 0
Factorial = 1
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\5. Factorial of Number.py
Enter Number: 1
The factorial is: 1
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\5. Factorial of Number.py
Enter Number: 5
The factorial is: 120
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\5. Factorial of Number.py
Enter Number: -2
Factorial is not defined for Negative Numbers!
>>>
```

### **Learning Outcome:**

By performing this experiment, use of if-elif-else loop was understood. This program helped in understanding the topics such as control flow statements, if-elif-else loop, and in developing required logic for problem solving. By performing this experiment, use of user defined functions was understood and the how to define a recursive function was clear. Functions were used to structure the program and perform required operations to get desired results.

## Experiment No. 7-A

### Title:

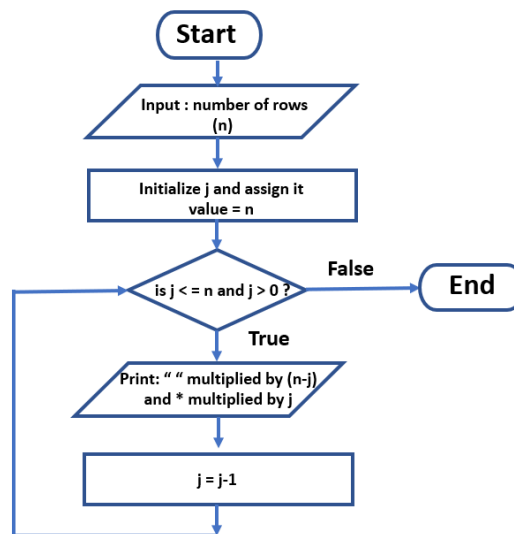
Write a Python Program to print an Inverted Star Pattern

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

#### Flowchart – Inverted Star Pattern



01/12/2021

### Algorithm:

1. Initialise the program
2. Initialise variable "n"
3. Store input of number of rows from user in "n"
4. Using for loop iterate j in range n to 0, with decrement of 1 with each iteration
5. For each iteration multiply " " with n-j and \* with j to manage spacing in rows
6. Print the concatenated result
7. End the program

## Source Code:

```
#Inverted Star Pattern
# Date: 01/12/2021
n=int(input("Enter number of rows of the pattern: "))
print()
print("Printing Star Pattern:")
print()
for i in range(n):
    print(' '*(n-i-1)+'* '*(i+1))
print()
print("Printing Inverted Star Pattern:")
print()
for j in range(n,0,-1):
    print(' '*(n-j)+'* '*(j))

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

```
#Inverted Star Pattern
# Date : 01/12/2021
n=int(input("Enter number of rows of the pattern: "))
print()
print("Printing Star Pattern:")
print()
for i in range(n):
    print(' '*(n-i-1)+'* '*(i+1))
print()
print("Printing Inverted Star Pattern:")
print()
for j in range(n,0,-1):
    print(' '*(n-j)+'* '*(j))

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

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\8. Inverted star pattern.py
Enter number of rows of the pattern: 10

Printing Star Pattern:

*
**
***
****
*****
*****
*****
*****
*****
*****

Printing Inverted Star Pattern:

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

## Learning Outcome:

By performing this experiment, it was understood that to solve a problem, one should look for the easiest and simplest approach possible. The syntax of for loop was understood and a basic program was written to solve the problem easily and efficiently.

## Experiment No. 7-B

### Title:

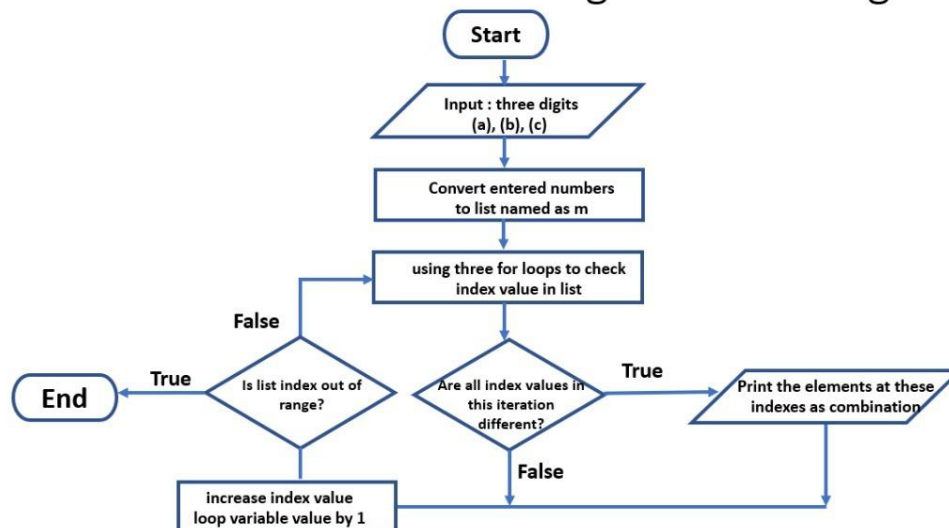
Write a Python Program to accept three digits and print all possible combinations of those digits

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Combinations using 3 entered Digits



01/12/2021

### Algorithm:

1. Initialise the program
2. Initialise variable "a", "b" and "c"
3. Store input of digits in "a", "b" and "c"
4. Create list "m" and input "a", "b" and "c" as its elements
5. Initialize empty list "n"
6. Using three for loops running from 0 to 3, to access the indexes of the elements of the list
7. The elements are printed if the values at the above indexes in the list are not equal
8. The above combination is appended into list n if not already existing in the list and then printed as output
9. End the program



## **Source Code:**

# Input 3 digits, print max possible combination

# Date: 01/12/2021

```
import sys
```

```
a = int(input("Enter First Digit : "))
```

```
b = int(input("Enter Second Digit : "))
```

```
c = int(input("Enter Third Digit : "))
```

```
if a > 9 or b > 9 or c > 9:
```

```
    sys.exit("Invalid Input, Enter only Single Digits, Recheck !")
```

```
m=[a,b,c]
```

```
n=[]
```

```
print()
```

```
print("Printing all possible unique combinations using entered digits:", a,b,c)
```

```
print()
```

```
for i in range(0,3):
```

```
    for j in range(0,3):
```

```
        for k in range(0,3):
```

```
            if (i != j and j != k and k != i):
```

```
                x= [m[i] , m[j] , m[k]]
```

```
                if x not in n: # to find unique combinations
```

```
                    n.append(x)
```

```
d=len(n)
```

```
for i in range(0,d):
```

```
    print(*n[i])
```

```
print()
```

```
print("Printed Number of Unique Combinations: ",d)
```

```

# Input 3 digits, print max possible combination
# Date: 01/12/2021

import sys

a = int(input("Enter First Digit : "))
b = int(input("Enter Second Digit : "))
c = int(input("Enter Third Digit : "))
if a > 9 or b > 9 or c > 9:
    sys.exit("Invalid Input, Enter only Single Digits, Recheck !")

m=[a,b,c]
n=[]
print()
print("Printing all possible unique combinations using entered digits:", a,b,c)
print()
for i in range(0,3):
    for j in range(0,3):
        for k in range(0,3):
            if (i != j and j != k and k != i):
                x= [m[i] , m[j] , m[k]]
                if x not in n: # to find unique combinations
                    n.append(x)
d=len(n)
for i in range(0,d):
    print(*n[i])

print()
print("Printed Number of Unique Combinations: ",d)

```

## Output Screenshots:

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\9. max combination using 3 digits.py
Enter First Digit : 2
Enter Second Digit : 3
Enter Third Digit : 5

Printing all possible unique combinations using entered digits: 2 3 5

2 3 5
2 5 3
3 2 5
3 5 2
5 2 3
5 3 2

Printed Number of Unique Combinations: 6
>>>

```

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\9. max combination using 3 digits.py
Enter First Digit : 2
Enter Second Digit : 6
Enter Third Digit : 2

Printing all possible unique combinations using entered digits: 2 6 2

2 6 2
2 2 6
6 2 2

Printed Number of Unique Combinations: 3
>>>

```

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\9. max combination using 3 digits.py
Enter First Digit : 1
Enter Second Digit : 9
Enter Third Digit : 12
SystemExit: Invalid Input, Enter only Single Digits, Recheck !
>>>

```

### **Learning Outcome:**

By performing this experiment, use of nested loops was understood. A code snippet was designed to filter out repeated output values from printing and thus only printing the necessary ones. It helped to understand the syntax of nested loops, and the use of lists to get required output in required format. The benefits of using lists and the operations/ functionalities they provide were made clear.

## Experiment No. 8-A

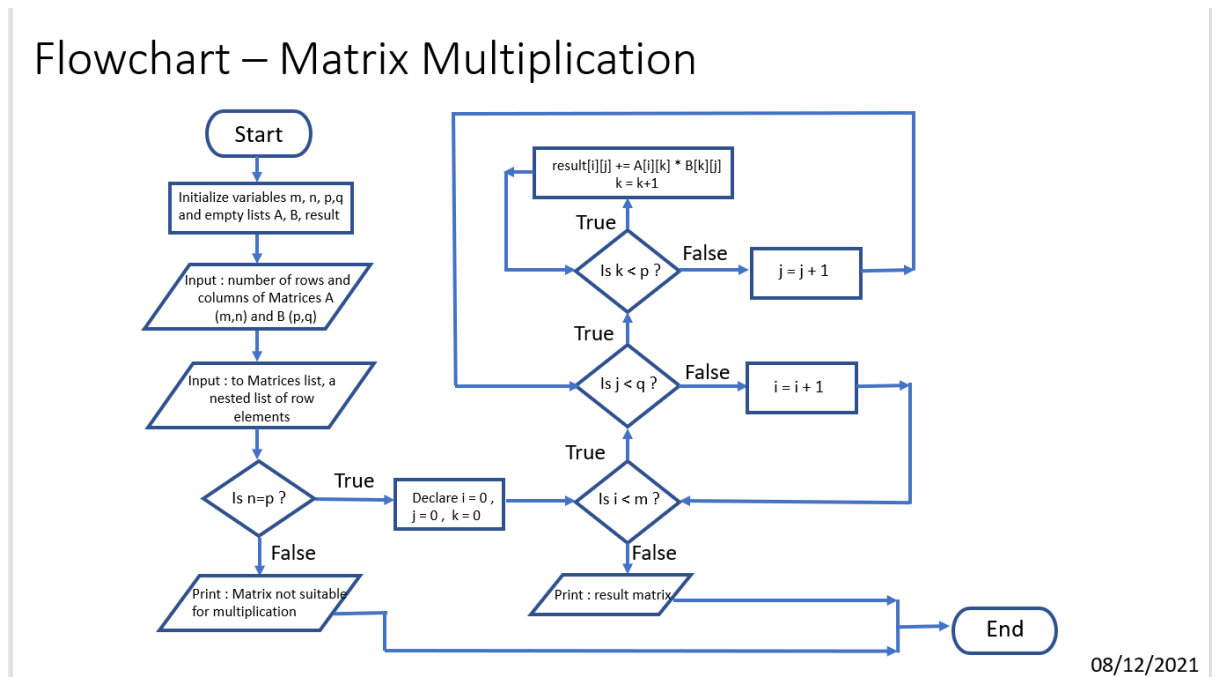
### Title:

Write a Python Program to multiply two matrices using nested loops.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:



### Algorithm:

1. Initialise the program
2. Initialise variables “m”, “n”, “p”, “q”
3. Create two empty lists “A” and “B”
4. Input number of rows and columns of both the matrices and store them in “m”, “n” and “p”, “q”.
5. Input the entries for matrix elements with each row as one list hence, making matrix list a nested list of row elements
6. Check if **n=p**. If false, exit the program with error “Matrices not suitable for Multiplication”
7. Else, define an empty matrix “**result**” with number of rows = “m” and columns = “q”, with each element being 0
8. Next, start a for loop (with variable i) which goes up to “m” giving row elements of “A”
9. Then, a nested for loop (with variable j) is created which goes up to “q” giving column elements of “B”.
10. And a final nested loop (with variable k) which goes up to “p” giving rows of “B”
11. Append the “**result**” matrix list as:  $result[i][j] += A[i][k] * B[k][j]$

12. Print the required matrix as **“result”**

13. End the program

### **Source Code:**

```
#Matrix Multiplication
```

```
#Date: 08/12/2021
```

```
import sys
```

```
print("Please input two matrices of order m X n and n X p ")
print()
```

```
print("For Matrix 1:")
m=int(input("Enter number of rows: "))
n=int(input("Enter number of column: "))
A = [ ]
print("Entries to be entered row - wise")
for i in range(m):
    a=[ ]
    for j in range(n):
        a.append(int(input("Enter Element: ")))
    A.append(a)
print()
```

```
print("For Matrix 2:")
p=int(input("Enter number of rows: "))
q=int(input("Enter number of columns: "))
B = [ ]
print("Entries to be entered row - wise")
for i in range(p):
    b=[ ]
    for j in range(q):
        b.append(int(input("Enter Element: ")))
    B.append(b)
print()
```

```
print("Printing Entered Matrices:")
for r in A:
    print(r)
print()
for r in B:
    print(r)
print()
```

```
if n != p:
    print("As per fundamentals of Matrix Multiplication, number of columns of first
matrix should be equal to number of rows of second matrix ... ")
```

```
sys.exit("Hence, Matrices entered not applicable for multiplication!")
```

```
rows,column=m,q
```

```
result = [[0]*column for i in range(rows)]
```

```
print("Performing Matrix Multiplication !")
```

```
for i in range(m): #rows of A
```

```
    for j in range(q): #columns of B
```

```
        for k in range(p): #rows of B
```

```
            result[i][j] += A[i][k] * B[k][j]
```

```
for r in result:
```

```
    print(r)
```

```
#Matrix Multiplication
#Date : 08/12/2021
import sys

print("Please input two matrices of order m X n and n X p ")
print()

print("For Matrix 1:")
m=int(input("Enter number of rows: "))
n=int(input("Enter number of column: "))
A = []
print("Entries to be entered row - wise")
for i in range(m):
    a=[]
    for j in range(n):
        a.append(int(input("Enter Element: ")))
    A.append(a)
print()

print("For Matrix 2:")
p=int(input("Enter number of rows: "))
q=int(input("Enter number of columns: "))
B = []
print("Entries to be entered row - wise")
for i in range(p):
    b=[]
    for j in range(q):
        b.append(int(input("Enter Element: ")))
    B.append(b)
print()

print("Printing Entered Matrices:")
for r in A:
    print(r)
print()
for r in B:
    print(r)
print()

if n != p:
    print("As per fundamentals of Matrix Multiplication, number of columns of first matrix should be equal to number of rows of second matrix ... ")
    sys.exit("Hence, Matrices entered not applicable for multiplication!")

rows,column=m,q
result = [[0]*column for i in range(rows)]

print("Performing Matrix Multiplication !")
for i in range(m): #rows of A
    for j in range(q): #columns of B
        for k in range(p): #rows of B
            result[i][j] += A[i][k] * B[k][j]
for r in result:
    print(r)
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\10. Matrix multiplication.py
Please input two matrices of order m X n and n X p

For Matrix 1:
Enter number of rows: 2
Enter number of column: 2
Entries to be entered row - wise
Enter Element: 1
Enter Element: 2
Enter Element: 3
Enter Element: 4

For Matrix 2:
Enter number of rows: 3
Enter number of columns: 2
Entries to be entered row - wise
Enter Element: 1
Enter Element: 2
Enter Element: 3
Enter Element: 4
Enter Element: 5
Enter Element: 6

Printing Entered Matrices:
[1, 2]
[3, 4]

[1, 2]
[3, 4]
[5, 6]

As per fundamentals of Matrix Multiplication, number of columns of first matrix should be equal to number of rows of second matrix ...
SystemExit: Hence, Matrices entered not applicable for multiplication!
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\10. Matrix multiplication.py
Please input two matrices of order m X n and n X p

For Matrix 1:
Enter number of rows: 2
Enter number of column: 3
Entries to be entered row - wise
Enter Element: 1
Enter Element: 2
Enter Element: 3
Enter Element: 4
Enter Element: 5
Enter Element: 6

For Matrix 2:
Enter number of rows: 3
Enter number of columns: 1
Entries to be entered row - wise
Enter Element: 1
Enter Element: 2
Enter Element: 4

Printing Entered Matrices:
[1, 2, 3]
[4, 5, 6]

[1]
[2]
[4]

Performing Matrix Multiplication !
[17]
[38]
>>> |
```



```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\10. Matrix multiplication.py
Please input two matrices of order m X n and n X p

For Matrix 1:
Enter number of rows: 2
Enter number of column: 2
Entries to be entered row - wise
Enter Element: 1
Enter Element: 2
Enter Element: 3
Enter Element: 4

For Matrix 2:
Enter number of rows: 2
Enter number of columns: 2
Entries to be entered row - wise
Enter Element: 1
Enter Element: 2
Enter Element: 3
Enter Element: 4

Printing Entered Matrices:
[1, 2]
[3, 4]

[1, 2]
[3, 4]

Performing Matrix Multiplication !
[7, 10]
[15, 22]
>>>

```

### **Learning Outcome:**

By performing this experiment, use of nested loops was understood. Using the functionalities provided by lists, matrices were entered as nested list where elements of each row were entered as a list.

This program, helped to understand the basic difference between lists and dictionaries and about where one should use list and where dictionaries are needed. It was understood that lists help in simplifying input, output and processes in between by providing properties and functions such as append, slicing and mutability. This program helped in solving a complex program by building a proper approach towards the problem statement and solving it by breaking into smaller parts and building a required logic.

## Experiment No. 8-B

### Title:

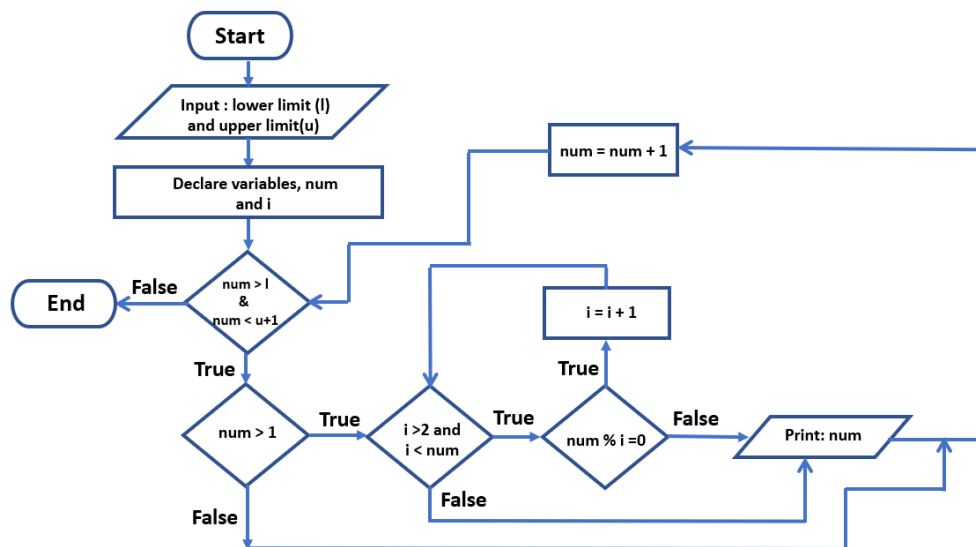
Write a Python Program to print all the prime numbers in a given range

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Prime Numbers in Given Range



15/12/2021

### Algorithm:

1. Initialise the program
2. Input the lower and upper limit from user and assign it to "l", "u"
3. Using for loop, a variable "num", is made to iterate through "l" to "u"
4. It is checked that "**num > 1**"
5. Then, using another for loop, with "i" iterating through **2** to "**num**", for each iteration it is checked if any remainder exists when "**num / i**"
6. If true, the number is not prime and the current iteration breaks and hence, it goes to next iteration with value of **i = i+1**
7. Else, if false, the number is prime and hence is printed
8. End the program

### Source Code:

#Prime Numbers in given range

# Date: 15/12/2021

```
l = int(input("Enter Lower Limit of the Range: "))
u = int(input("Enter Upper Limit of the Range: "))
print()
c=0
print(f'Printing Prime Numbers Between Entered Range {l} to {u}')
for num in range(l, u+1):
    if num>1:
        for i in range(2, num):
            if (num%i==0):
                break
        else:
            print(num)
            c+=1
print()
print("Total Prime Numbers Printed = ",c)
```

```
#Prime Numbers in given range
# Date: 15/12/2021

l = int(input("Enter Lower Limit of the Range: "))
u = int(input("Enter Upper Limit of the Range: "))
print()
c=0
print(f'Printing Prime Numbers Between Entered Range {l} to {u}')
for num in range(l, u+1):
    if num>1:
        for i in range(2, num):
            if (num%i==0):
                break
        else:
            print(num)
            c+=1
print()
print("Total Prime Numbers Printed = ",c)
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\11. Prime Number in Given Range.py
Enter Lower Limit of the Range: 1
Enter Upper Limit of the Range: 10

Printing Prime Numbers Between Entered Range 1 to 10
2
3
5
7

Total Prime Numbers Printed = 4
>>>
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\11. Prime Number in Given Range.py
Enter Lower Limit of the Range: -5
Enter Upper Limit of the Range: 20

Printing Prime Numbers Between Entered Range -5 to 20
2
3
5
7
11
13
17
19

Total Prime Numbers Printed = 8
>>>
```

## Learning Outcome:

The following program helped in understanding the use of nested loops. It helped in understanding the use and syntax of break statements and the use of if-else statements. It helped in realising the impact indentation makes and thus understand that even if logic is correct, because of indentation syntax errors, the program may not display correct output as per need.

The program helped in understanding control flow statements and the functioning of nested loops.

## Experiment No. 9-A

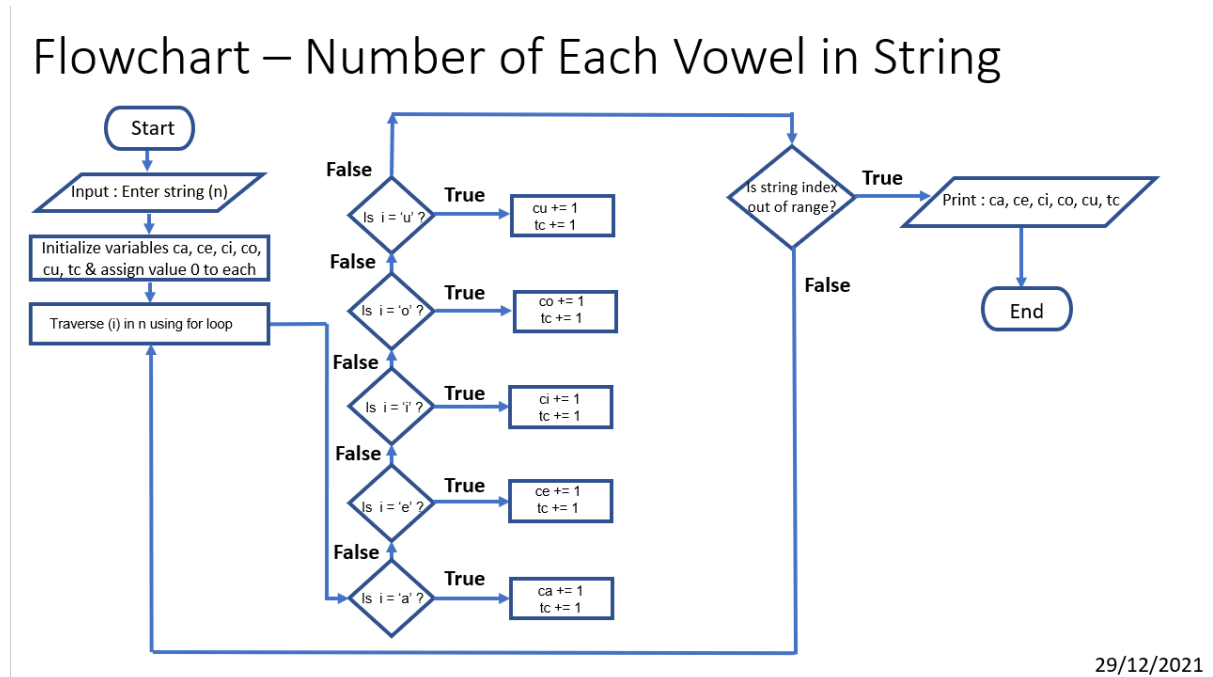
**Title:**

Write a Python Program to count the number of each vowel in the entered string.

**Tool/Platform:**

## Microsoft Word / PowerPoint and Python IDLE

### Flowchart:



### Algorithm:

1. Initialise the program
2. Initialise variables **"ca"**, **"ce"**, **"ci"**, **"co"**, **"cu"**, **"tc"** as counters for vowels and assign them value = 0
3. Input string from user and assign it to **"n"**
4. Traverse **"i"** through the string **"n"**
5. At each iteration, check if the value of **"i"** is equal to any of the vowel, that is **"a"**, **"e"**, **"i"**, **"o"**, **"u"**.
6. If true, increment the counter for that respective vowel and the counter for total vowels by 1
7. If false, move on to next iteration
8. Print the values of each counter including those of 5 vowels and of total vowel count
9. End the program

### **Source Code:**

```
#Number of each vowel
#Date : 29/12/2021
n = input("Enter a String: ")
ca, ce, ci, co, cu, tc = 0, 0, 0, 0, 0, 0
for i in n:
    if i.lower() == 'a' :
        ca += 1
        tc += 1
    elif i.lower() == 'e' :
        ce += 1
        tc += 1
    elif i.lower() == 'i' :
        ci += 1
        tc += 1
    elif i.lower() == 'o' :
        co += 1
        tc += 1
    elif i.lower() == 'u' :
        cu += 1
        tc += 1
print(f'Number of Vowels in the entered string "{n}" are as follows:')
print("Vowel Count for 'a' or 'A' = ", ca)
print("Vowel Count for 'e' or 'E' = ", ce)
print("Vowel Count for 'i' or 'I' = ", ci)
print("Vowel Count for 'o' or 'O' = ", co)
print("Vowel Count for 'u' or 'U' = ", cu)
print()
print("Total Vowel Count = ", tc)
```

```

#Number of each vowel
#Date : 29/12/2021

n = input("Enter a String: ")
ca, ce, ci, co, cu, tc = 0, 0, 0, 0, 0, 0

for i in n:
    if i.lower() == 'a' :
        ca += 1
        tc += 1
    elif i.lower() == 'e' :
        ce += 1
        tc += 1
    elif i.lower() == 'i' :
        ci += 1
        tc += 1
    elif i.lower() == 'o' :
        co += 1
        tc += 1
    elif i.lower() == 'u' :
        cu += 1
        tc += 1

print(f'Number of Vowels in the entered string "{n}" are as follows:')
print("Vowel Count for 'a' or 'A' = ", ca)
print("Vowel Count for 'e' or 'E' = ", ce)
print("Vowel Count for 'i' or 'I' = ", ci)
print("Vowel Count for 'o' or 'O' = ", co)
print("Vowel Count for 'u' or 'U' = ", cu)
print()
print("Total Vowel Count = ", tc)

```

### Output Screenshot:

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/gargm/Desktop/COLLEGE/Programming and Problem Solving Lab/17. number of each vowel.py
Enter a String: Checking for number of each vowels. Test String Entered
Number of Vowels in the entered string "Checking for number of each vowels. Test String Entered" are as follows:
Vowel Count for 'a' or 'A' = 1
Vowel Count for 'e' or 'E' = 8
Vowel Count for 'i' or 'I' = 2
Vowel Count for 'o' or 'O' = 3
Vowel Count for 'u' or 'U' = 1

Total Vowel Count = 15
>>>

```

### **Learning Outcomes:**

This program helped in understanding the use of if-elif-else loop. The following program helped to understand that while solving a problem statement, by building a proper approach and logic, a problem can be solved easily. The program also helped to understand the use of for loop to traverse a string and equate it with the required condition. It also helped in realising the difference between assignment (=) and equality (==) operators. The program also helped to practice the use and syntax of f-string formatting.



## Experiment No. 9-B

### Title:

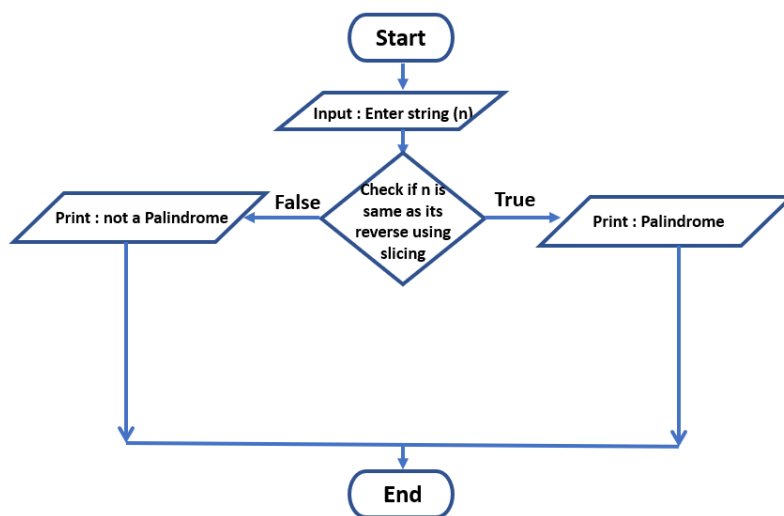
Write a Python Program to check if entered string is a palindrome or not.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – String Palindrome



29/12/2021

### Algorithm:

1. Initialise the program
2. Input string from user and assign it to "n"
3. Check if n is equal to its reverse using slicing as `n[::-1]`
4. If true, Print "Palindrome"
5. Else, print "Not a Palindrome"
6. End the program

### Source Code:

```
#String Palindrome
#Date : 29/12/2021
n=input("Enter a string: ")
print()
s=n.lower()
if s == s[::-1]:
    print(f'Entered String "{n}" is a palindrome !')
else:
    print(f'Entered String "{n}" is not a palindrome !')
```

```

#String Palindrome
#Date : 29/12/2021

n=input("Enter a string: ")
print()
s=n.lower()
if s == s[::-1]:
    print(f'Entered String "{n}" is a palindrome !')
else:
    print(f'Entered String "{n}" is not a palindrome !')

```

### Output Screenshots:

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\16. String Palindrome.py
Enter a string: redivider

Entered String "redivider" is a palindrome !
>>>

```

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\16. String Palindrome.py
Enter a string: use the radar

Entered String "use the radar" is not a palindrome !
>>>

```

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\16. String Palindrome.py
Enter a string: ruler

Entered String "ruler" is not a palindrome !
>>>

```

### Learning Outcomes:

The following program helped in understanding slicing operations on a string and how slicing can help to reverse a string. The following program also helped to realise that the problem should be approached in a way that it can be solved easily, faster and in less steps.

The program also helped to understand use of functions such as lower() on strings.

## Experiment No. 10-A

### Title:

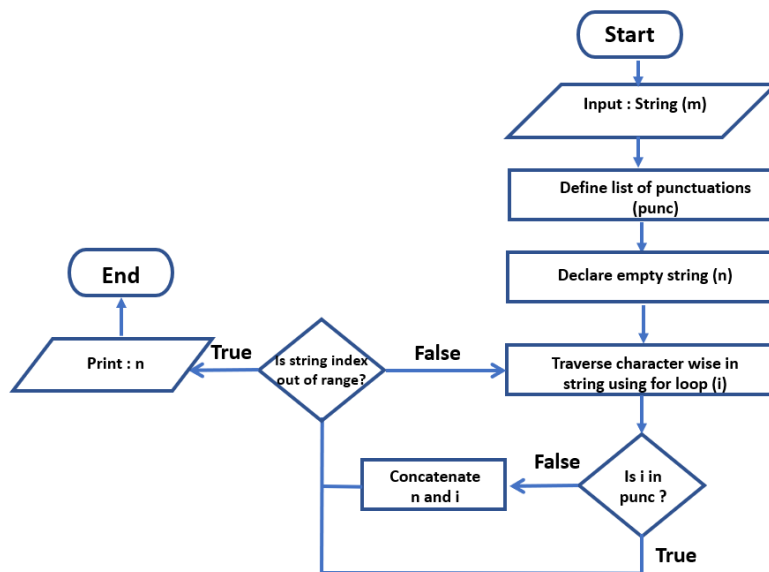
Write a Python Program to remove punctuations from a string

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

#### Flowchart – Remove Punctuations from String



22/12/2021

### Algorithm:

1. Initialise the program
2. Declare a list of punctuations marks as “**punc**”
3. Input string from user and assign to “**m**”
4. Initialise an empty string “**n**”
5. Using for loop iterate “**i**” through the string “**m**” assigning it each character of the string at a time in each iteration
6. Check if the current value of “**i**” matches any element in the list “**punc**”
7. If false, concatenate that value of “**i**” with existing value of “**n**”
8. Else, omit the value of “**i**” and move on to next iteration
9. Print the value of “**n**” as filtered string
10. End the program

## Source Code:

```
#remove punctuations
#Date: 22/12/2021
punc = [ '!', '"', '?', '!', '-', '_', '\\', '|', ';', ':', ' ', '(', ')', "...", "/" ]
m = input("Enter String: ")
print()
n = ""
for i in m:
    if i not in punc:
        n += i
print("Entered String is: \n", m)
print()
print("String after Punctuation Filter is: \n", n)
```

```
#remove punctuations
#Date: 22/12/2021
punc = [ '!', '"', '?', '!', '-', '_', '\\', '|', ';', ':', ' ', '(', ')', "...", "/" ]
m = input("Enter String: ")
print()
n = ""
for i in m:
    if i not in punc:
        n += i
print("Entered String is: \n", m)
print()
print("String after Punctuation Filter is: \n", n)
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\13. remove punctuations.py
Enter String: Test Case !!, Checking Python Program... ??language

Entered String is:
Test Case !!, Checking Python Program... ??language

String after Punctuation Filter is:
Test Case Checking Python Program language
>>>
```

## Learning Outcome:

The following program helped in understanding the use of lists and string. It helped to understand how for loop can be used to traverse through a string. The use and function of membership operators was understood.

## Experiment No. 10-B

### Title:

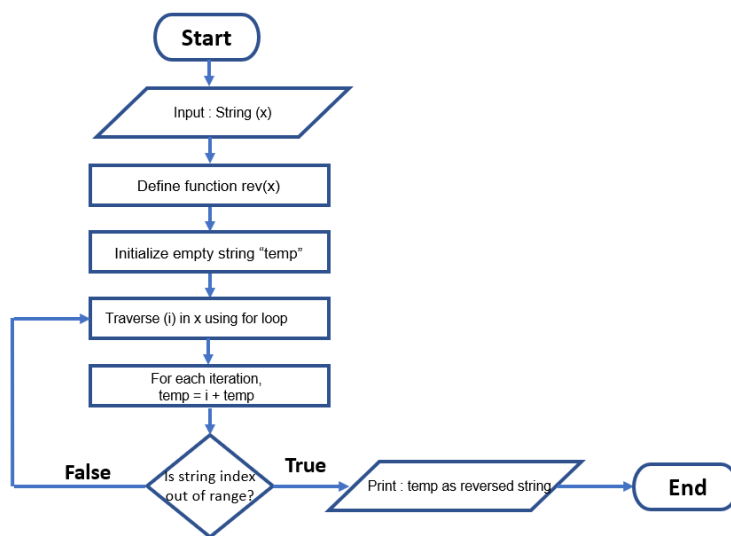
Write a Python Program to reverse a string using user defined function

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

#### Flowchart – Reverse a String



29/12/2021

### Algorithm:

1. Initialise program
2. Input string from user and assign to "x"
3. Define a function rev(x)
4. Initialize empty string "temp"
5. Traverse in x (using i) sing for loop
6. For each iteration of loop, temp = i + temp
7. Print the value of temp at the end of loop as the reversed string
8. End the program

## Source Code:

```
#Reverse string
#Functions
#Date : 29/12/2021
```

```
def rev(x):
    temp=""
    for i in x:
        temp=i+temp
    return temp
```

```
n=input("Enter a string: ")
```

```
print('Entered String:', n)
print("Reversed String:", rev(n))
```

```
"""
```

```
def reverse(x):
    s = x[ : :-1]
    print(s)
reverse(n)
"""
```

```
#Reverse string
#Functions
#Date : 29/12/2021

def rev(x):
    temp=""
    for i in x:
        temp=i+temp
    return temp

n=input("Enter a string: ")

print('Entered String:', n)
print("Reversed String:", rev(n))

"""

def reverse(x):
    s = x[ : :-1]
    print(s)
reverse(n)
"""
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\15. Reverse String using function.py
Enter a string: Reverse a string using user defined function
Entered String: Reverse a string using user defined function
Reversed String: noitcnuf denifed resu gnisu gnirts a esreveR
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\15. Reverse String using function.py
Enter a string: reverse
Entered String: reverse
Reversed String: esrever
>>>
```

## Learning Outcome:

The following program helped in understanding the use of string. It helped to understand how for loop can be used to traverse through a string. It helped to understand how the result of concatenation can change based on the position of the + operator. It helped to understand how to define, utilise and make programs simpler with functions.

## Experiment No. 11-A

### Title:

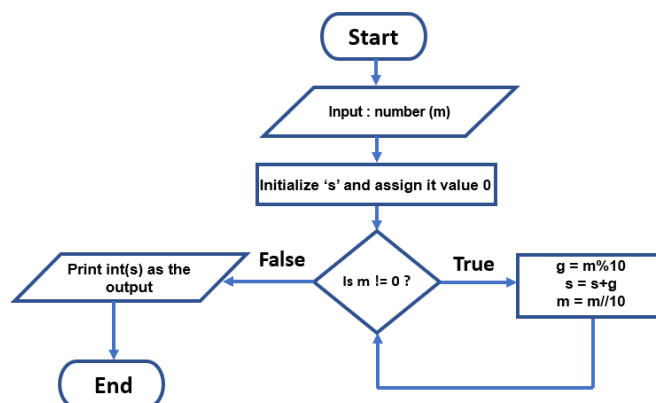
Write a Python Program to find the sum of digits of a number

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Sum of digits of a number



29/12/2021

### Algorithm:

1. Initialise the program
2. Input the number and assign it to “**m**”
3. Initialise “**s**” and assign it value = 0
4. Create while loop with condition “**m!=0**”
5. When true, perform the operations as  $g = m \% 10$  (to get the last digit of the number)
6. Increment the value of s as  $s = s + g$
7. Perform  $m = m // 10$  (to remove the last digit)
8. Continue with loop iterations
9. When loop ends, print the output as int(s)
10. End the program



## Source Code:

```
#sum of all digits of a given number
#date : 29/12/2021
m=int(input("Enter a number : "))
m=abs(m)
s=0
while m!=0:
    g=m%10
    s+=g
    m=int(m/10)
ans=int(s)
print("The sum of the digits of given number is : ",ans)
```

```
#sum of all digits of a given number
#date : 29/12/2021
m=int(input("Enter a number : "))
m=abs(m)
s=0
while m!=0:
    g=m%10
    s+=g
    m=int(m/10)
ans=int(s)
print("The sum of the digits of given number is : ",ans)
```

## Output Screenshot:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\18. Sum of digits.py
Enter a number : 12345
The sum of the digits of given number is : 15
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\18. Sum of digits.py
Enter a number : -12345
The sum of the digits of given number is : 15
>>>
```

## Learning Outcome:

The following program helped to understand the use and syntax of while loop. It helped to release the difference between syntax and usage of while and for loops.

The program helped to understand the usage of different arithmetic operators and of absolute (abs()) function.

## Experiment No. 11-B

### Title:

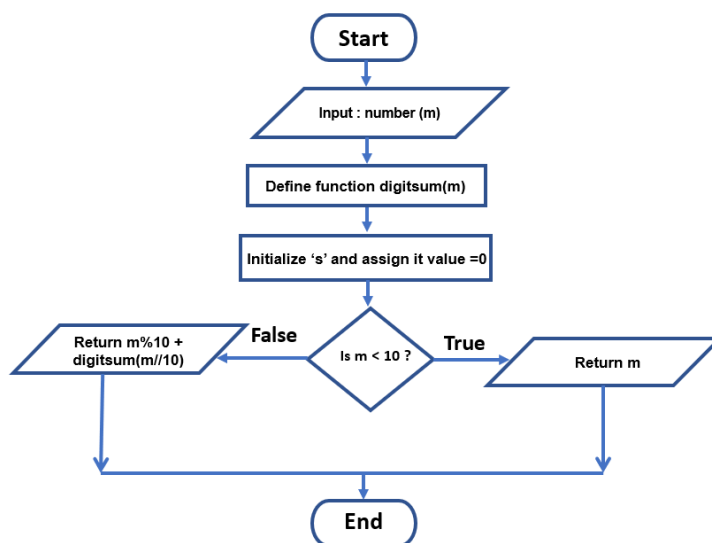
Write a Python Program to find the sum of digits of a number using recursive function

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Sum of digits of a number – Recursion



29/12/2021

### Algorithm:

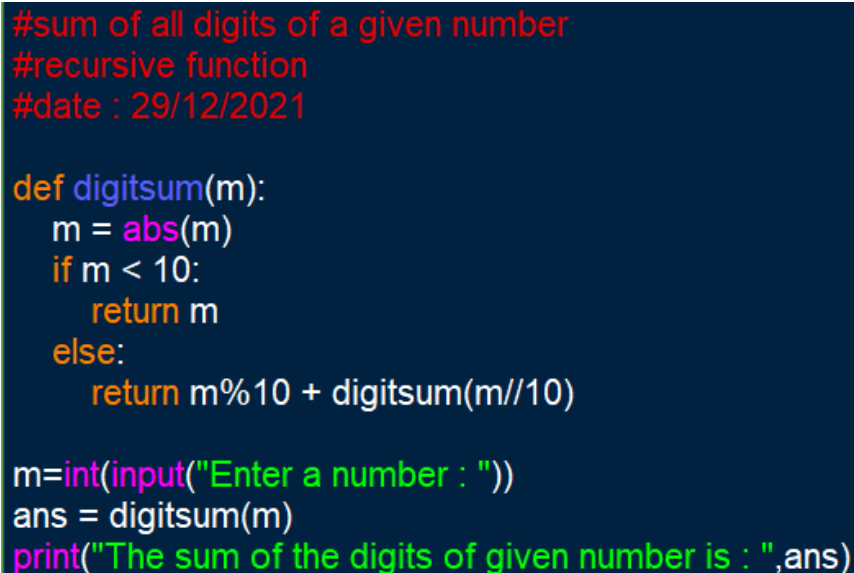
1. Initialise the program
2. Input the number and assign it to "m"
3. Define a recursive function "**digitsum(m)**"
4. Check if  $m < 10$
5. If true, return m as the output
6. Else, return  $m \% 10 + \text{digitsum}(m // 10)$  (using recursive function)
7. End the program

## Source Code:

```
#sum of all digits of a given number
#recursive function
#date : 29/12/2021

def digitsum(m):
    m = abs(m)
    if m < 10:
        return m
    else:
        return m%10 + digitsum(m//10)

m=int(input("Enter a number : "))
ans = digitsum(m)
print("The sum of the digits of given number is : ",ans)
```

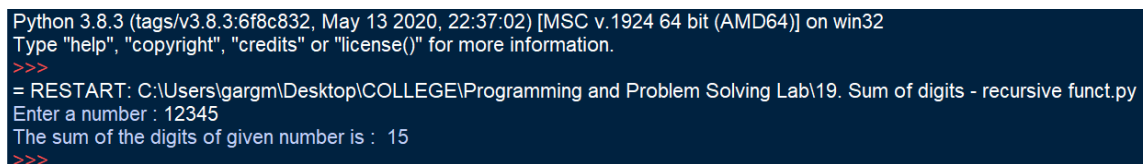


```
#sum of all digits of a given number
#recursive function
#date : 29/12/2021

def digitsum(m):
    m = abs(m)
    if m < 10:
        return m
    else:
        return m%10 + digitsum(m//10)

m=int(input("Enter a number : "))
ans = digitsum(m)
print("The sum of the digits of given number is : ",ans)
```

## Output Screenshot:



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\19. Sum of digits - recursive funct.py
Enter a number : 12345
The sum of the digits of given number is : 15
>>>
```

## Learning Outcome:

The following program helped to understand the use and syntax of recursive function. It helped to release the need of it to simplify the program.

The program helped to understand the usage of different arithmetic operators and of absolute (abs()) function.

## Experiment No. 12

### Title:

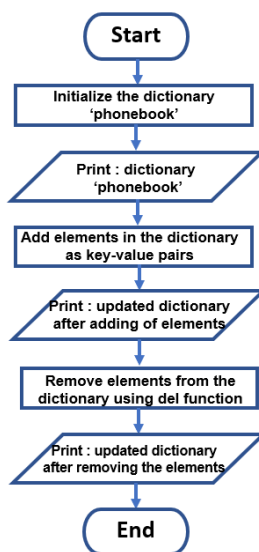
Write a Python Program to create a database of phone numbers using a dictionary and perform Initialise, Adding and Removing Operations.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Database of Phone numbers – Dictionary



03/01/2022

### Algorithm:

1. Initialise the program
2. Initialise a dictionary named as “**phonebook**”
3. Print the dictionary as output
4. Add new elements in the dictionary as key value pairs by assigning name of the person as key and the phone number as value
5. New elements in the dictionary can also be added using **update()** function
6. Print the updated dictionary “**phonebook**” after the elements are added
7. Remove few elements from the list by using **del method** where the key is mentioned in the statement. Elements can also be removed using **pop()** function
8. Print the updated dictionary “**phonebook**” after the elements are removed
9. End the program

## **Source Code:**

#Dictionary : Phonebook

#Date: 03/01/22

```
phonebook = { }  
phonebook['Mohan'] = 9978445512  
phonebook['Mahesh'] = 9765432980  
phonebook['Suresh'] = 9955643218  
phonebook['Ayush'] = 6353634318  
phonebook['Jasprit'] = 8769120542  
print(phonebook)  
print()
```

#initialise

```
pb = {"Mohan" : 9978445512, "Mahesh" : 9765432980, "Suresh" : 9955643218,  
      "Ayush" : 6353634318, "Jasprit" : 8769120542}  
print("Initialised Dictionary Phonebook:")  
print(pb)  
print()
```

#adding elements

```
pb['Ramesh'] = 9875436671  
pb.update(Samir = 7698634518)  
print("Updated Dictionary after adding elements is:")  
print(pb)  
print()
```

#removing elements

```
del pb['Mohan']  
pb.pop('Suresh')  
print("Updated Dictionary after removing elements is:")  
print(pb)
```

```

#Dictionary : Phonebook
#Date: 03/01/22

phonebook = { }
phonebook['Mohan'] = 9978445512
phonebook['Mahesh'] = 9765432980
phonebook['Suresh'] = 9955643218
phonebook['Ayush'] = 6353634318
phonebook['Jasprit'] = 8769120542
print(phonebook)
print()

#initialise
pb = {"Mohan" : 9978445512, "Mahesh" : 9765432980, "Suresh" : 9955643218, "Ayush" : 6353634318, "Jasprit" : 8769120542}
print("Initialised Dictionary Phonebook:")
print(pb)
print()

#adding elements
pb['Ramesh'] = 9875436671
pb.update(Samir = 7698634518)
print("Updated Dictionary after adding elements is:")
print(pb)
print()

#removing elements
del pb['Mohan']
pb.pop('Suresh')
print("Updated Dictionary after removing elements is:")
print(pb)

```

## Output Screenshot:

```

Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\SEM 1\Programming and Problem Solving Lab\20. Dictionary of phonebook.py
{'Mohan': 9978445512, 'Mahesh': 9765432980, 'Suresh': 9955643218, 'Ayush': 6353634318, 'Jasprit': 8769120542}

Initialised Dictionary Phonebook:
{'Mohan': 9978445512, 'Mahesh': 9765432980, 'Suresh': 9955643218, 'Ayush': 6353634318, 'Jasprit': 8769120542}

Updated Dictionary after adding elements is:
{'Mohan': 9978445512, 'Mahesh': 9765432980, 'Suresh': 9955643218, 'Ayush': 6353634318, 'Jasprit': 8769120542, 'Ramesh': 9875436671, 'Samir': 7698634518}

Updated Dictionary after removing elements is:
{'Mahesh': 9765432980, 'Ayush': 6353634318, 'Jasprit': 8769120542, 'Ramesh': 9875436671, 'Samir': 7698634518}
>>>

```

## Learning Outcome:

The following program helped to understand the working and functions of dictionary. It helped to release how a dictionary is different from other data types including lists and tuples. It helped to understand the practical use and applications of a dictionary. The following program also helped to understand how to define a dictionary and perform basic operations on it including adding of new elements and removing the unwanted ones.

## Experiment No. 13

### Title:

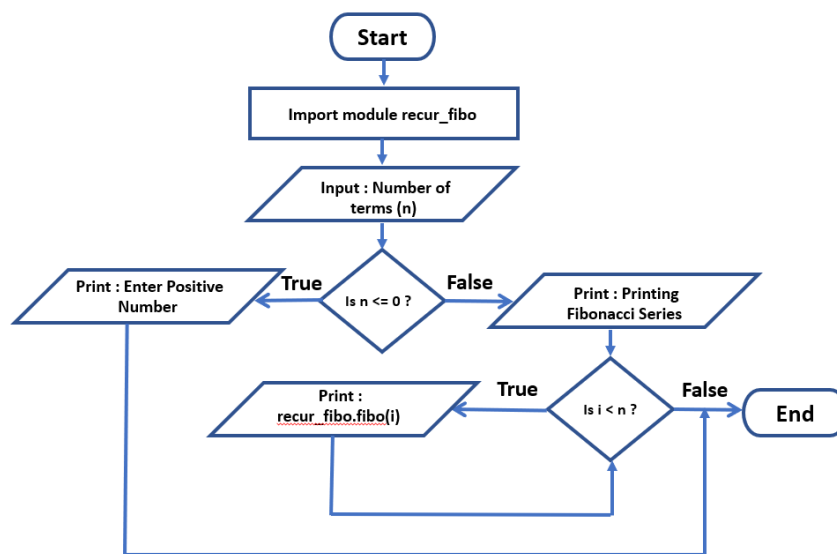
Write a Python Program to find Fibonacci Series using modules

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Fibonacci Series using Modules



22/12/2021

### Algorithm:

10. Initialise the program
11. Import created module "**recur\_fibo**"
12. Input the number of terms to be printed and assign it to "**n**"
13. Check if  $n \leq 0$ , if true, Print: "Enter Positive Number" and end the program
14. Else, Print "Printing Fibonacci Series"
15. Create for loop which iterates the value of "**i**" till "**n**"
16. For each iteration, print the output of "**recur\_fibo.fibo(i)**"
17. End the program

## Source Code:

```
import recur_fibo

n = int(input("Enter number of terms: "))
if n <= 0:
    print("Plese enter a positive integer")
else:
    print("Printing Fibonacci Series:")
    for i in range(n):
        print(recur_fibo.fibo(i))
```

## imported module: (recur\_fibo.py)

```
def fibo(n):
    if n <= 1:
        return n
    else:
        return(fibo(n-1) + fibo(n-2))
```

```
#importing modules
#fibo
#22/12/2021

import recur_fibo

n = int(input("Enter number of terms: "))
if n <= 0:
    print("Plese enter a positive integer")
else:
    print("Printing Fibonacci Series:")
    for i in range(n):
        print(recur_fibo.fibo(i))
```

```
#Fibonacci Series
#modules

def fibo(n):
    if n <= 1:
        return n
    else:
        return(fibo(n-1) + fibo(n-2))
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\12. importing fibonacci series as module.py
Enter number of terms: 10
Printing Fibonacci Series:
0
1
1
2
3
5
8
13
21
34
>>>
```



**Learning Outcome:**

The following program helped in understanding the use of functions and modules. It helped to understand how to define and import modules to make programs simpler.

The program also helped to understand the utility of modules in long programs. It helped to understand and deploy modules and functions.

## Experiment No. 14

### Title:

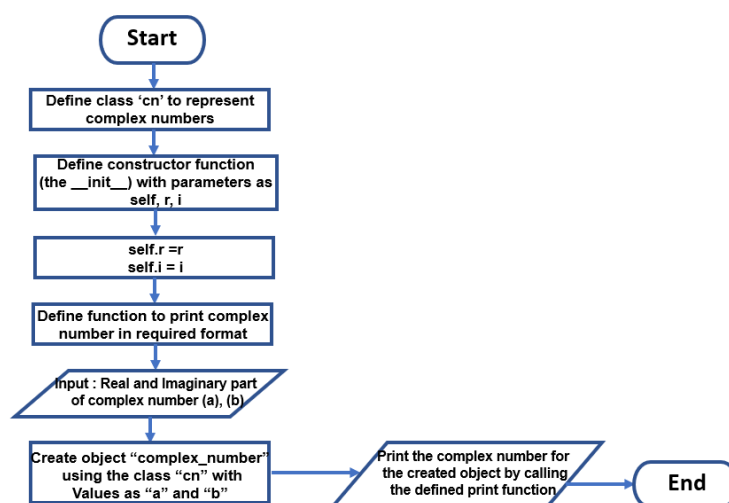
Write a Python Program to represent a complex number using classes

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Complex Number - Classes



04/01/2022

### Algorithm:

1. Initialise the program
2. Define a class "**cn**".
3. Define the constructor function – the **\_\_init\_\_** function with parameters : "**self**", "**r**", "**i**"
4. Assign values as **self.r = r** and **self.i = i**
5. Define a function "**print\_cn**" with parameter as **self** and prints the complex number in required format using the values **self.r** and **self.i**
6. Input the real part of the complex number and assign it to "**a**", also input the imaginary part of the complex number and assign it to "**b**"
7. Create object "**complex\_number**" using the class **cn** and values **a**, **b**
8. Call the print function defined in the class, for the created object
9. The complex number gets printed in the required format
10. End the program

## Source Code:

#Classes : Complex Number

#Date : 04/01/2022

```
class cn:
    def __init__(self, r, i):
        self.r = r
        self.i = i

    def print_cn(self):
        if self.i == 0:
            print(self.r, "+", self.i, "i", end=' ')
            print('=' , self.r)
        else:
            print(self.r, "+", self.i, "i", end = ' ')
            print('=' , self.i, "i")

a = int(input("Enter Real Part of the Complex Number: "))
b = int(input("Enter Imaginary Part of the Complex Number: "))
complex_number = cn(a,b)
complex_number.print_cn()
```

```
#Classes : Complex Number
#Date : 04/01/2022

class cn:
    def __init__(self, r, i):
        self.r = r
        self.i = i

    def print_cn(self):
        if self.i == 0:
            print(self.r, "+", self.i, "i", end=' ')
            print('=' , self.r)
        else:
            print(self.r, "+", self.i, "i", end = ' ')
            print('=' , self.i, "i")

a = int(input("Enter Real Part of the Complex Number: "))
b = int(input("Enter Imaginary Part of the Complex Number: "))
complex_number = cn(a,b)
complex_number.print_cn()
```

## Output Screenshots:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\SEM 1\Programming and Problem Solving Lab\21. Class - Complex Numbers.py
Enter Real Part of the Complex Number: 19
Enter Imaginary Part of the Complex Number: 3
19 + 3 i = 3 i
>>>
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\SEM 1\Programming and Problem Solving Lab\21. Class - Complex Numbers.py
Enter Real Part of the Complex Number: 10
Enter Imaginary Part of the Complex Number: 0
10 + 0 i = 10
>>> |
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\SEM 1\Programming and Problem Solving Lab\21. Class - Complex Numbers.py
Enter Real Part of the Complex Number: 0
Enter Imaginary Part of the Complex Number: 13
0 + 13 i = 13 i
>>>
```

## Learning Outcome:

The following helped in understanding the syntax used in OOPs and how to implement it. It helped in learning the class, object and function definitions and calls.

Using the following program, the functioning of `__init__` function was understood. It also helped to understand the use of `self` as a parameter.

The following program was useful to understand the need, applications and benefits of using OOPs in python programming.

## Experiment No. 15

### Title:

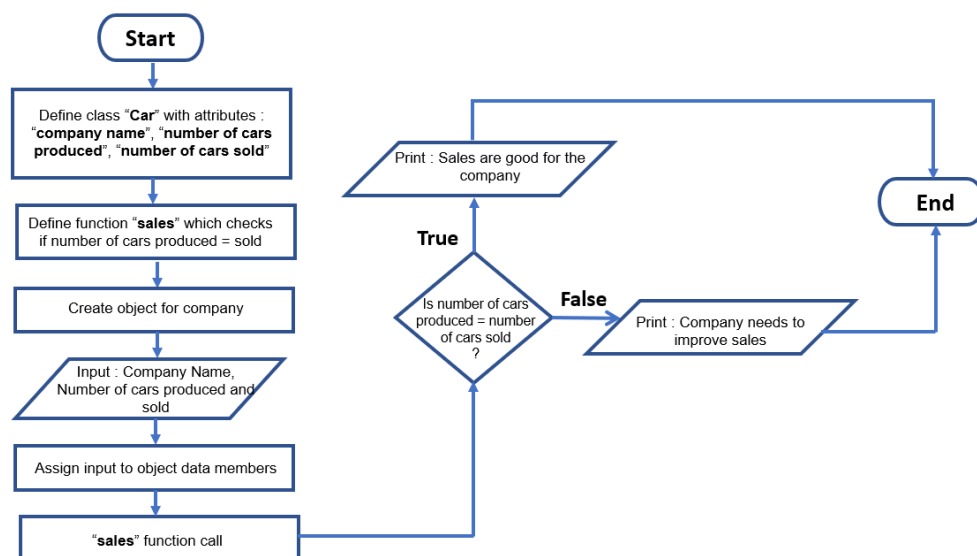
Write a Python Program to create a sales system for car dealerships using Object-Oriented techniques.

### Tool/Platform:

Microsoft Word / PowerPoint and Python IDLE

### Flowchart:

Flowchart – Car Dealership Sales System - OOPs



28/12/2021

### Algorithm:

11. Initialise program
12. Define a class **"Car"** with attributes/data members as **"company name"**, **"number of cars produced"**, **"number of cars sold"**
13. Define a function **"Sales"** in which it is checked if **"number of cars produced = number of cars sold"**
14. If true, " the sales are good", else, "sales need to be improved"
15. Create objects for each company with attributes as **company name**, **"number of cars produced"**, **"number of cars sold"**
16. Perform function call (the Sales function defined in the class) for the object created
17. End the program

## **Source Code:**

#Class - Sales System for Cars

#Date 28/12/2021

```
class Car():
    company = " "
    produced_no = 0
    sold_no = 0

    def sales(self):
        if self.sold_no == self.produced_no:
            print(f'Sales of {self.sold_no} out of {self.produced_no} manufactured
cars by {self.company}. The company is performing well.')
        else:
            print(f'Sales of {self.sold_no} out of {self.produced_no} manufactured
cars are not enough. Sales should be improved by {self.company}')
```

```
tata = Car()
tata.company = "TATA"
tata.produced_no = 500
tata.sold_no = 400
tata.sales()
print()
```

```
mahindra = Car()
mahindra.company = "MAHINDRA"
mahindra.produced_no = 500
mahindra.sold_no = 500
mahindra.sales()
print()
```

```
honda = Car()
honda.company = "HONDA"
honda.produced_no = 500
honda.sold_no = 500
honda.sales()
print()
```

```
kia = Car()
kia.company = "KIA"
kia.produced_no = 400
kia.sold_no = 400
kia.sales()
print()
```

```
toyota = Car()
toyota.company = "TOYOTA"
```

```
toyota.produced_no = 400
toyota.sold_no = 350
toyota.sales()
print()
```

```
#Class - Sales System for Cars
#Date 28/12/2021
class Car():
    company = ""
    produced_no = 0
    sold_no = 0

    def sales(self):
        if self.sold_no == self.produced_no:
            print(f'Sales of {self.sold_no} out of {self.produced_no} manufactured cars by {self.company}. The company is performing well.')
        else:
            print(f'Sales of {self.sold_no} out of {self.produced_no} manufactured cars are not enough. Sales should be improved by {self.company}.')

tata = Car()
tata.company = "TATA"
tata.produced_no = 500
tata.sold_no = 400
tata.sales()
print()

mahindra = Car()
mahindra.company = "MAHINDRA"
mahindra.produced_no = 500
mahindra.sold_no = 500
mahindra.sales()
print()

honda = Car()
honda.company = "HONDA"
honda.produced_no = 500
honda.sold_no = 500
honda.sales()
print()

kia = Car()
kia.company = "KIA"
kia.produced_no = 400
kia.sold_no = 400
kia.sales()
print()

toyota = Car()
toyota.company = "TOYOTA"
toyota.produced_no = 400
toyota.sold_no = 350
toyota.sales()
print()
```

## Output Screenshot:

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\gargm\Desktop\COLLEGE\Programming and Problem Solving Lab\14. Class - Sales System for Cars.py
Sales of 400 out of 500 manufactured cars are not enough. Sales should be improved by TATA

Sales of 500 out of 500 manufactured cars by MAHINDRA. The company is performing well.

Sales of 500 out of 500 manufactured cars by HONDA. The company is performing well.

Sales of 400 out of 400 manufactured cars by KIA. The company is performing well.

Sales of 350 out of 400 manufactured cars are not enough. Sales should be improved by TOYOTA
```

### **Learning Outcome:**

The following program helped in understanding the object-oriented techniques used in python programming. It helped in understanding the syntax used in OOPs and how to implement it. It helped in learning the class, object and function definitions and calls.

The following program was useful to understand the need, applications and benefits of using OOPs in python programming.