



**SNS COLLEGE OF TECHNOLOGY**

(An Autonomous Institution)  
COIMBATORE – 35



**DEPARTMENT OF AGRICULTURE ENGINEERING**

**19ITP202-PYTHON PROGRAMMING LABORATORY**

**PRACTICAL RECORD**

**NAME** : \_\_\_\_\_

**REGISTER No** : \_\_\_\_\_

**CLASS** : \_\_\_\_\_

**SEMESTER** : \_\_\_\_\_



**SNS COLLEGE OF TECHNOLOGY**

(An Autonomous Institution)

COIMBATORE – 35



**DEPARTMENT OF AGRICULTURE ENGINEERING**

**19ITP202-PYTHON PROGRAMMING LABORATORY**

Name .....

Class.....

Branch.....

Roll No.....

Register No

Certified that this is the bonafide record of work done by the above student of the **19ITP202-PYTHON PROGRAMMING LABORATORY** during the year 2021-2022.

**Signature of Lab In-charge**

**Head of the Department**

Submitted for the practical examination held on .....

**Internal Examiner**

**External Examiner**

## INDEX

S.No	Date	Name of the Experiment	Page No	Performance (25)	Record (15)	Viva (10)	Total (50)	Sign
Average Marks								

## Unit 1

### INTRODUCTION TO PYTHON

#### EXPERIMENT :1

**Aim:**

To write a python program to print the given string “Hello world” using interactive shell.

**Algorithm:**

Step 1: Start the program.

Step 2: Use print() function to print the given “hello world” string.

Step 3: End the program.

**Program:**

```
print("Hello world")
```

**Output:**

Hello world

**Result:**

The above program is executed successfully and the required output is obtained.

## **EXPERIMENT :2**

### **Aim:**

To write a python program to print the given “Hello world” string using variable.

### **Algorithm:**

Step 1: Start the program

Step 2: Declare a variable and initiate the value “hello world” to the variable.

Step 3: Using print() function print the output.

Step 4: End the program.

### **Program:**

```
string1 = "Hello world"  
print(string1)
```

### **Output:**

Hello world

### **Result:**

The above program is executed successfully and the required output is obtained.

## **EXPERIMENT :3**

### **Aim:**

To write a python program to create a list and store some values and print them.

### **Algorithm:**

Step 1: Start the program.

Step 2: Declare a variable, create a list and initiate the values for the list.

Step 3: Using print() function print the output.

Step 4: End the program.

### **Program:**

```
list = [1,2,3,4,5,6]
```

```
print(list)
```

### **Output:**

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

### **Result:**

The above program is executed successfully and the required output is obtained.

## **EXPERIMENT: 4**

### **Aim:**

To write a python program to print the given string.

### **Algorithm:**

Step 1: Start the program

Step 2: Declare two variables of string type and initiate with the values.

Step 3: Using concatenation and print() function print the outputs.

Step 4: End the program

### **Program:**

```
string1= "HelloWorld,PythonisHighlevel,GeneralpurposeProgramminglanguage "  
  
string2="GuidoVanRossuminventedthePythonprogramminglanguagein1990s"  
  
print(string1+string2)
```

### **Output:**

```
HelloWorld,PythonisHighlevel,General purpose Programming language  
GuidoVanRossuminventedthePythonprogramminglanguagein1990s
```

### **Result:**

The above program is executed successfully and the required output is obtained.

## UNIT-2

### CONTROL FLOW AND ARRAYS

#### EXPERIMENT: 1

##### AIM:

To find SUM and MULTIPLY of any three numbers

##### Algorithm

Step 1: Start a program

Step 2: Declare the variables a,b,c and sum

Step 3: Add and multiple the given numbers and assign the result sum

Step 4: Print the output

Step 5: End the program

##### Program:

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
c = int(input("Enter third number: "))

sum = a+b+c

mul = a*b*c

print("The sum is:",sum,"\nThe product is: ",mul)
```

##### Output:

Enter first number: 1

Enter second number: 2

Enter third number: 3

The sum is: 6

The product is: 6

##### Result:

The given python programing is successfully executed.



## **EXPERIMENT : 2**

### **AIM**

Program to find the average of any five numbers.

### **Algorithm:**

Step 1: Start a program

Step 2: Declare the variable n, average and sum

Step 3: Using the FOR loop store the input in the form of list

Step 4: Add the values of sum

Step 5: Average = sum/5

Step 6: Display the average

Step 7: Print the output

Step 8: End the program

### **Program:**

```
n = 5
```

```
average = 0
```

```
sum = 0
```

```
for num in range(0,n+1,1):
```

```
    sum = sum+num;
```

```
average = sum / n
```

```
print("Average of first ", n, "natural number is: ", average)
```

### **Output:**

Average of first 5 natural number is: 3.0

### **Result:**

The given python programing is successfully executed.

## **EXPERIMENT : 3**

### **AIM:**

Program to find simple interest.

### **Algorithm:**

Step 1: Start a program

Step 2: Read principal amount, rate and period

Step 3: Calculate interest using the formula  $SI = (p * n * r) / 100$

Step 4: Print simple interest

Step 5: End the program

### **Program:**

```
p = int(input("Enter the principle amount:"))  
r = int(input("Enter the rate: "))  
n = int(input("Enter the time period: "))  
SI = (p*n*r)/100  
print("The simple interest is: ",SI)
```

### **Output:**

Enter the principle amount:5000

Enter the rate: 2

Enter the time period: 3

The simple interest is: 300.0

### **Result:**

The given python programing is successfully executed.

## EXPERIMENT : 4

### AIM:

- a) program to store and display the above mentioned cities and corresponding items using arrays.
- b) Program to display name of cities where sales man has delivered maximum and minimum number of items.
- c) Program to search the number of items to be delivered of a user supplied city

### Algorithm:

- Step 1: Start a program
- Step 2: Declare the variable
- Step 3: Store the values using array
- Step 4: Display the list of cities using FOR loop
- Step 5: Print the name of cities where maximum and  
Minimum deliveries obtained
- Step 6: Find respective cities using if else
- Step 7: End the program

### Program:

```
from array import *

city = ["Aligarh", "Agra", "Baroda", "Banaras", "Chennai", "New Delhi", "New Jalparipur",
        "Howrah", "Kolkata", "Mumbai"]

items = array('i', [18,25,13,43,8,67,29,11,56,33])

#Item display

print("List of cities and no of items delivered\n")

for i in range(0,10):

    print(city[i],"\t",items[i])

print()

#Max and Min deliveries

max = (items.index(max(items)))

min = (items.index(min(items)))

print("Maximum delivery in: ",city[max])
```

```
print("Minimum delivery in: ",city[min])
```

```
#Search for respective city
```

```
src = input("Enter the city name to be found: ")
```

```
for i in range(0,10):
```

```
    if(city[i].lower()==src.lower()):
```

```
        print("City: ",city[i])
```

```
        print("Products Sold: ",items[i])
```

```
        break
```

```
    else:
```

```
        continue
```

```
else:
```

```
    print("No match found: ",src)
```

**Output:**

List of cities and no of items delivered

Aligarh	18
Agra	25
Baroda	13
Banaras	43
Chennai	8
New Delhi	67
New Jalparipur	29
Howrah	11
Kolkata	56
Mumbai	33

Maximum delivery in: New Delhi

Minimum delivery in: Chennai

Enter the city name to be found: chennai

City: Chennai

Products Sold: 8

**Result:**

The given python programing is successfully executed.

## **EXPERIMENT : 5**

### **AIM:**

Program to find the volume of a sphere with radius 5

### **Algorithm:**

Step 1: Start a program

Step 2: Define the radius of sphere

Step 3: Define the pie and assign

Step 4: Calculate the volume of the sphere as

$$(4/3)*\text{math.pi}*\text{pow}$$

Step 5: Assign the volume of sphere

Step 6: Print the volume of sphere

Step 7: End the program

### **Program:**

```
import math

r = float(input("Enter the radius: "))

vol = (4/3)*math.pi*pow(r,3)

print("The volume is: %.2f"%vol)
```

### **Output:**

Enter the radius: 4.5

The volume is: 381.70

### **Result:**

The given python programing is successfully executed.

## EXPERIMENT : 6

### AIM:

Program to find the total wholesale book cost for 60 copies.

### Algorithm:

Step 1: Start a program

Step 2: Declare the variable

Step 3: Calculate the discount using formula

Step 4: Print the discount

Step 5: Print the wholesale price of book

Step 6: End the program

### Program:

```
no_of_books = int(input("Enter no of books: "))
cov_price = float(input("Enter the cover price: "))
discount = float(input("Enter the discount percentage: "))

cost = (cov_price)*(discount/100)
cost_aft_disc = cov_price - cost

ship1 = float(input("Enter the shipping price for first copy: "))
rem_charge = float(input("Enter the shipping price for other copies: "))
ship2 = rem_charge*(no_of_books-1)

total = cost_aft_disc * no_of_books
total_amnt = total + ship1 + ship2

print("The total amount of books are: %.3f"%total_amnt)
```

**Output:**

Enter no of books: 60

Enter the cover price: 240.95

Enter the discount percentage: 40

Enter the shipping price for first copy: 30

Enter the shipping price for other copies: 0.75

The total amount of books are: 8748.450

**Result:**

The given python programing is successfully executed.



## Unit – 3

### CLASSES AND FUNCTION

#### EXPERIMENT :1

#### AIM

program to create Teacher , Program and Student Classes with above mentioned data member sandmember functions.

#### ALGORITHM

- Create a class teacher with name, department, hours, programsTaught as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class program with name, department, duration as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class student with name, department, rollNo, program as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create an object for the class teacher.
- Set the details for the object using setDetails() method.
- Get the details of the instance using getDetails() method and print it.
- Create an object for the class program.
- Set the details for the object using setDetails() method.
- Get the details of the instance using getDetails() method and print it.
- Create an object for the class student.
- Set the details for the object using setDetails() method.
- Get the details of the instance using getDetails() method and print it.
- End the program.

#### PROGRAM

```
# Class Teacher
class Teacher:
    name = ""
    department = ""
    hours = 0
    programsTaught = 0

# Set details
def setDetails(self, Name, Dept, Hours, programs):
    self.name = Name
    self.department = Dept
    self.hours = Hours
    self.programsTaught = programs
```

```
# Get details
def getDetails(self):
    details = []
    details.append(self.name)
    details.append(self.department)
    details.append(self.hours)
    details.append(self.programsTaught)
```

```
    return details
```

```
# Class Program
```

```
class Program:
    name = ""
    department = ""
    duration = 0
```

```
# Set details
```

```
def setDetails(self, Name, Dept, duration):
    self.name = Name
    self.department = Dept
    self.duration = duration
```

```
# Get details
```

```
def getDetails(self):
    details = []
    details.append(self.name)
    details.append(self.department)
    details.append(self.duration)
```

```
    return details
```

```
# Class student
```

```
class Student:
    name = ""
    rollNo = ""
    program = ""
    department = ""
```

```
# Set details
```

```
def setDetails(self, Name, rollno, program, Dept):
    self.name = Name
    self.rollNo = rollno
    self.department = Dept
    self.program = program
```

```
# Get details
```

```
def getDetails(self):
    details = []
```

```
        details.append(self.name)
        details.append(self.rollNo)
        details.append(self.program)
        details.append(self.department)

    return details

# Driver code

# Teacher
staff1 = Teacher()
staff1.setDetails('ashok', 'IT', 9, 11)
teacherDetail = staff1.getDetails()
print(teacherDetail)

# Program
program = Program()
program.setDetails('Python programming', 'Computer Science', 9)
programDetail = program.getDetails()
print(programDetail)

# Student
stud1 = Student()
stud1.setDetails('Ramalingam', '19CECS031', 'Computer Science', 'Python Programming')
studentDetail = stud1.getDetails()
print(studentDetail)
```

## OUTPUT

```
['ashok', 'IT', 9, 11]
['Python programming', 'Computer Science', 9]
['Ramalingam', '19CECS031', 'Computer Science', 'Python Programming']
```

## RESULT

The given python programing is successfully executed.

## EXPERIMENT: 2

### AIM

program the default department of Teacher, Program and Student should be „ComputerScience“ ;however a different department could be assigned at runtime.

### ALGORITHM:

- Create a class teacher with name, department, hours, programsTaught as data members and setDetails() and getDetails() as member functions with a constructor to give a default value for the department as “Computer Science”. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class program with name, department, duration as data members and setDetails() and getDetails() as member functions with a constructor to give a default value for the department as “Computer Science”. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class student with name, department, rollNo, program as data members and setDetails() and getDetails() as member functions with a constructor to give a default value for the department as “Computer Science”. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create 2 objects for the class teacher staff1 and staff2.
- Set the details for the objects using setDetails() method.
- Get the details of the instances using getDetails() method and print it.
- End the program.

### Program Code:

```
# Class Teacher
class Teacher:
    name = ""
    department = ""
    hours = 0
    programsTaught = 0

    def __init__(self):
        self.department = "Computer Science"

# Set details
def setDetails(self, Name, Dept, Hours, programs):
    self.name = Name
    self.department = Dept
    self.hours = Hours
    self.programsTaught = programs

# Get details
def getDetails(self):
    details = []
    details.append(self.name)
    details.append(self.department)
    details.append(self.hours)
    details.append(self.programsTaught)
```

```
    return details
```

```
# Class Program
```

```
class Program:
```

```
    name = ""
```

```
    department = ""
```

```
    duration = 0
```

```
# Constructor
```

```
def __init__(self):
```

```
    department = "Computer Science"
```

```
# Set details
```

```
def setDetails(self, Name, Dept, duration):
```

```
    self.name = Name
```

```
    self.department = Dept
```

```
    self.duration = duration
```

```
# Get details
```

```
def getDetails(self):
```

```
    details = []
```

```
    details.append(self.name)
```

```
    details.append(self.department)
```

```
    details.append(self.duration)
```

```
    return details
```

```
# Class student
```

```
class Student:
```

```
    name = ""
```

```
    rollNo = ""
```

```
    program = ""
```

```
    department = ""
```

```
def __init__(self):
```

```
    self.department = "Computer Science"
```

```
# Set details
```

```
def setDetails(self, Name, rollno, program, Dept):
```

```
    self.name = Name
```

```
    self.rollNo = rollno
```

```
    self.department = Dept
```

```
    self.program = program
```

```
# Get details
```

```
def getDetails(self):
```

```
    details = []
```

```
    details.append(self.name)
```

```
details.append(self.rollNo)
details.append(self.program)
details.append(self.department)
```

```
return details
```

```
# Driver code
```

```
# Teacher
```

```
staff1 = Teacher()
staff1.setDetails('ashok', 'IT', 9, 11)
Staff2.setDetails('Dinesh', 'CSE', 12, 5)
teacherDetail = staff1.getDetails()
print(teacherDetail)
teacherDetail = staff2.getDetails()
print(teacherDetail)
```

**Output:**

```
['ashok', 'IT', 9, 11]
['Dinesh', 'CSE', 12, 5]
```

**RESULT**

The given python programing is successfully executed.

## EXPERIMENT :3

### AIM

program to over load “+” operator to add the hours of two teachers

### ALGORITHM

- Create a class teacher with name, department, hours, programsTaught as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list with \_\_add\_\_() with a parameter staff2. This function is used to overload the ‘+’ operator. This will return the sum of the hours of 2 teachers.
- Create a class program with name, department, duration as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class student with name, department, rollNo, program as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create an object for the class teacher.
- Set the details for the object using setDetails() method.
- Invoke the \_\_add\_\_() method and pass the 2 objects as parameters.
- Print the output.
- End the program.

### PROGRAM

```
# Class Teacher
```

```
class Teacher:
```

```
    name = ""
```

```
    department = ""
```

```
    hours = 0
```

```
    programsTaught = 0
```

```
# Set details
```

```
def setDetails(self, Name, Dept, Hours, programs):
```

```
    self.name = Name
```

```
    self.department = Dept
```

```
    self.hours = Hours
```

```
    self.programsTaught = programs
```

```
# Get details
```

```
def getDetails(self):
```

```
    details = []
```

```
    details.append(self.name)
```

```
    details.append(self.department)
```

```
    details.append(self.hours)
```

```
    details.append(self.programsTaught)
```

```
    return details
```

```
def __add__(self, staff2):
```

```
total = self.hours + staff2.hours  
return total
```

```
# Class Program
```

```
class Program:
```

```
    name = ""
```

```
    department = ""
```

```
    duration = 0
```

```
# Set details
```

```
def setDetails(self, Name, Dept, duration):
```

```
    self.name = Name
```

```
    self.department = Dept
```

```
    self.duration = duration
```

```
# Get details
```

```
def getDetails(self):
```

```
    details = []
```

```
    details.append(self.name)
```

```
    details.append(self.department)
```

```
    details.append(self.duration)
```

```
    return details
```

```
# Class student
```

```
class Student:
```

```
    name = ""
```

```
    rollNo = ""
```

```
    program = ""
```

```
    department = ""
```

```
# Set details
```

```
def setDetails(self, Name, rollno, program, Dept):
```

```
    self.name = Name
```

```
    self.rollNo = rollno
```

```
    self.department = Dept
```

```
    self.program = program
```

```
# Get details
```

```
def getDetails(self):
```

```
    details = []
```

```
    details.append(self.name)
```

```
    details.append(self.rollNo)
```

```
    details.append(self.program)
```

```
    details.append(self.department)
```

```
    return details
```



```
# Driver code
```

```
# Teacher
```

```
staff1 = Teacher()
```

```
staff2 = Teacher()
```

```
staff1.setDetails('Venkat', 'IT', 9, 11)
```

```
staff2.setDetails('Senthil', 'CSE', 7, 15)
```

```
s3 = staff1 + staff2
```

```
print("Total hours:", s3)
```

## **OUTPUT**

Total hours: 16

## **RESULT**

The given python programing is successfully executed.

## EXPERIMENT :4

### AIM

Program to create two sub classes Residential Student and Non Residential Student inherited from Student class. Residential Student would have a data member Hall of Residence and Non-Residential Student would have Address as its data member.

### ALGORITHM

- Create a class teacher with name, department, hours, programsTaught as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class program with name, department, duration as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class student with name, department, rollNo, program as data members and setDetails() and getDetails() as member functions. setDetails() is used to set the data members and getDetails() is used to return the values of the data members in the form of list.
- Create a class ResidentialStudent inherited from the class student with HallOfResidence as data members and Print() as member function. Print() method is used to return the values of the data members from the base class and append the value of HallOfResidence in the form of list.
- Create a class teacher inherited from the class student with Address as data members and Print() as member function. Print() method is used to return the values of the data members from the base class and append the value of Address in the form of list.
- Create objects for the classes ResidentialStudent and NonResidentialStudent.
- Set the details for the objects using setDetails() method and set the values for the HallOfResidence & Address individually.
- Get the details of the instances using Print() method and print it.
- End the program.

### PROGRAM

```
# Class Teacher
class Teacher:
    name = ""
    department = ""
    hours = 0
    programsTaught = 0

# Set details
def setDetails(self, Name, Dept, Hours, programs):
    self.name = Name
    self.department = Dept
    self.hours = Hours
    self.programsTaught = programs

# Get details
```

```
def getDetails(self):
    details = []
    details.append(self.name)
    details.append(self.department)
    details.append(self.hours)
    details.append(self.programsTaught)
```

```
    return details
```

```
# Class Program
```

```
class Program:
```

```
    name = ""
```

```
    department = ""
```

```
    duration = 0
```

```
# Constructor
```

```
def __init__(self):
```

```
    department = "Computer Science"
```

```
# Set details
```

```
def setDetails(self, Name, Dept, duration):
```

```
    self.name = Name
```

```
    self.department = Dept
```

```
    self.duration = duration
```

```
# Get details
```

```
def getDetails(self):
```

```
    details = []
```

```
    details.append(self.name)
```

```
    details.append(self.department)
```

```
    details.append(self.duration)
```

```
    return details
```

```
# Class student
```

```
class Student:
```

```
    name = ""
```

```
    rollNo = ""
```

```
    program = ""
```

```
    department = ""
```

```
# Set details
```

```
def setDetails(self, Name, rollno, program, Dept):
```

```
    self.name = Name
```

```
    self.rollNo = rollno
```

```
    self.department = Dept
```

```
    self.program = program
```

```
# Get details
```

```
def getDetails(self):
    details = []
    details.append(self.name)
    details.append(self.rollNo)
    details.append(self.program)
    details.append(self.department)

    return details

# Class residential student
class ResidentialStudent(Student):
    HallOfResidence = 0

    def Print(self):
        details = self.getDetails()
        details.append(self.HallOfResidence)
        return details

class NonResidentialStudent(Student):
    Address = ""

    def Print(self):
        details = self.getDetails()
        details.append(self.Address)
        return details

# Driver code

# Student
stud1 = ResidentialStudent()
stud2 = NonResidentialStudent()

stud1.setDetails('Ram', '12', 'Computer Science', 'Python Programming')
stud1.HallOfResidence = 203
stud2.setDetails('Sam', '24', 'Computer Science', 'Python Programming')
stud2.Address = "Ram Nagar, Coimbatore, Tamilnadu - 641022"

stud1Det = stud1.Print()
stud2Det = stud2.Print()

print(stud1Det)
print(stud2Det)
```

**OUTPUT**

['Ram', '12', 'Computer Science', 'Python Programming', 203]

['Sam', '24', 'Computer Science', 'Python Programming', 'Ram Nagar, Coimbatore, Tamilnadu - 641022']

**RESULT**

The given python programing is successfully executed.

**UNIT - 4**  
**FILE HANDLING**

**EXPERIMENT:1**

**AIM**

Program to print number of days in a month

**ALGORITHM**

step 1: START

step 2: Declare the variable

step 3: Declare a function using if-else condition to calculate number of days in a month

step 4: Print number of days in a month

step 5: END

**PROGRAM**

```
class monthdays:
    def no_of_days(self, year, month):
        leap = 0
        if year % 4 == 0:
            leap = 1
        elif year % 100 == 0:
            leap = 0
        if month == 2:
            return 28 + leap
        list = [1,3,5,7,8,10,12]
        if month in list:
            return 31
        return 30

ob = monthdays()
```

```
year = int(input("Enter the year: "))  
month = int(input("Enter the month number ranges 1 - 12: "))  
print(ob.no_of_days(year,month))
```

## **OUTPUT**

```
Enter the year: 2017  
Enter the month number ranges 1 - 12: 4  
30
```

## **RESULT**

The given python programing is successfully executed.

## **EXPERIMENT:2**

### **AIM**

Program to find the area of triangle and show its type

### **ALGORITHM**

step 1: START

step 2: Declare the variable

step 3: Declare a function to find a valid triangle and then define the type.

Step 4 : calculate the area of triangle

step 5 : Print the value

step 6 : END

### **PROGRAM**

#Function to define whether given is a valid triangle or not

```
def side_check(a, b, c):  
    if a+b>=c and b+c>=a and c+a>=b:  
        return True  
    else:  
        return False
```

#Function to define the type

```
def type_of_tri(a, b, c):  
    if a==b and b==c:  
        print("Triangle is Equilateral.")  
    elif a==b or b==c or a==c:  
        print("Triangle is Isosceles.")  
    else:  
        print("Triangle is Scalane.")
```

#Getting three sides

```
side_a = float(input('Enter length of side a: '))
```



```

side_b = float(input('Enter length of side b: '))

side_c = float(input('Enter length of side c: '))


#Function calling and decision making
if side_check(side_a, side_b, side_c):

    type_of_tri(side_a, side_b, side_c)

#Calculate the semi-perimeter

    semi = (side_a+side_b+side_c)/2

    #Calculate the area

    area = (semi*(semi-side_a)*(semi-side_b)*(semi-side_c)) ** 0.5

    print("The area of the triangle is %0.2f" %area)


else:

    print("Tringle is not possible from given sides.")

```

## OUTPUT

```

Enter length of side a: 10
Enter length of side b: 10
Enter length of side c: 10
Triangle is Equilateral.
The area of the triangle is 43.30

```

## RESULT

The given python programing is successfully executed.

## **EXPERIMENT:3**

### **AIM**

Program to convert the capital letter if it is a small letter and Vice-Versa.

### **ALGORITHM**

step 1: START

step 2: Declare the char data type to get the value from input

step 3: call the function swapcase to change the character from upper case to lower case vice versa.

Step 4 : print the value.

Step 5 : END

### **PROGRAM**

```
char = input("Enter the character to be case changed: ")  
  
print(char.swapcase())
```

### **OUTPUT**

Enter the character to be case changed: G

g

### **RESULT**

The given python programing is successfully executed.

## **EXPERIMENT:4**

### **AIM**

Program to find those numbers which are divisible by 7 and multiple of 5,between 1500 and 2700

### **ALGORITHM**

step 1: START

step 2: Declare an empty list

step 3: create a FOR loop to check the condition

Step 4 : print the value.

Step 5 : END

### **PROGRAM**

```
for x in range(1500 , 2701):  
    if (x%7 == 0) and (x%5 == 0):  
        emp_list.append(str(x))  
  
#Printing the result  
  
print (','.join(emp_list))
```

### **OUTPUT**

1505,1540,1575,1610,1645,1680,1715,1750,1785,1820,1855,1890,1925,1960,1995,2030,2065,2100,  
2135,2170,2205,2240,2275,2310,2345,2380,2415,2450,2485,2520,2555,2590,2625,2660,2695

### **RESULT**

The given python programing is successfully executed.

## **EXPERIMENT:5**

### **AIM**

Program to ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user

### **ALGORITHM**

step 1: START

step 2: Declare the variable to get input from user

step 3: using if-else condition check the value is odd or even

Step 4 : print the value.

Step 5 : END

### **PROGRAM**

```
num = int(input("Enter the integer to be checked whether odd or even: "))  
if num%2 == 0:  
    print("The given number is even")  
else:  
    print("The given number is odd")
```

### **OUTPUT**

```
Enter the integer to be checked whether odd or even: 17  
The given number is odd
```

### **RESULT**

The given python programing is successfully executed.

## UNIT V

### TEMPLATES

#### EXPERIMENT :1

#### AIM

A function to swap two numbers using functional template. The number could be Integer or float that depends on the user inputs.

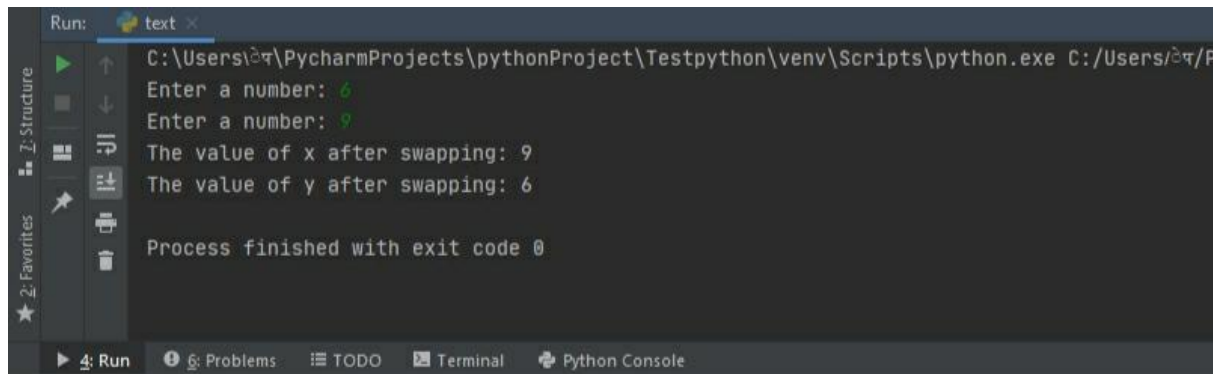
#### ALGORITHM

- Step 1: Start a program
- Step 2: Declare the variable x,y
- Step 3: Swap two function using functional template
- Step 4: Print the output
- Step 5: End the program

#### PROGRAM

```
x = int(input("Enter a number: "))
y = int(input("Enter a number: "))
temp = x
x = y
y = temp
print('The value of x after snapping: {}'.format(x))
print('The value of y after snapping: {}'.format(y))
```

#### OUTPUT



```
Run: text x
C:\Users\ԶԴ\PycharmProjects\pythonProject\Testpython\venv\Scripts\python.exe C:/Users/ԶԴ/P
Enter a number: 6
Enter a number: 9
The value of x after swapping: 9
The value of y after swapping: 6

Process finished with exit code 0
```

## RESULT:

The given python programming is successfully executed

## EXPERIMENT: 2

### AIM

Create a class matrix that has matrix data members and getvalue(), setvalue() as member function. Write a program in python to perform Matrix operation. Add and multiplication using class template.

### ALGORITHM

Step 1: Start a program

Step 2: Declare the variable x,y and result

Step 3: Add the two matrix and print the output

Step 4: Multiply the two matrix and print the output

Step 5: End the program

### PROGRAM

```
X = [[12,7,3],
```

```
      [4 ,5,6],
```

```
      [7 ,8,9]]
```

```
Y = [[5,8,1],
```

```
      [6,7,3],
```

```
      [4,5,9]]
```

```
result = [[0,0,0]
```

```
          [0,0,0],
```

```
          [0,0,0]]
```

```
Print('Add of two matrix')
```

```
# iterate through rows
```

```
for i in range(len(X)):
```

```
    # iterate through columns
```

```
    for j in range(len(X[0])):
```

```
        result[i][j] = X[i][j] + Y[i][j]
```

```
for r in result:
```

```

print(r)

Print('multiply of two matrix')

# iterate through rows of X

for i in range(len(X)):

    # iterate through columns of Y

    for j in range(len(Y[0])):

        # iterate through rows of Y

        for k in range(len(Y)):

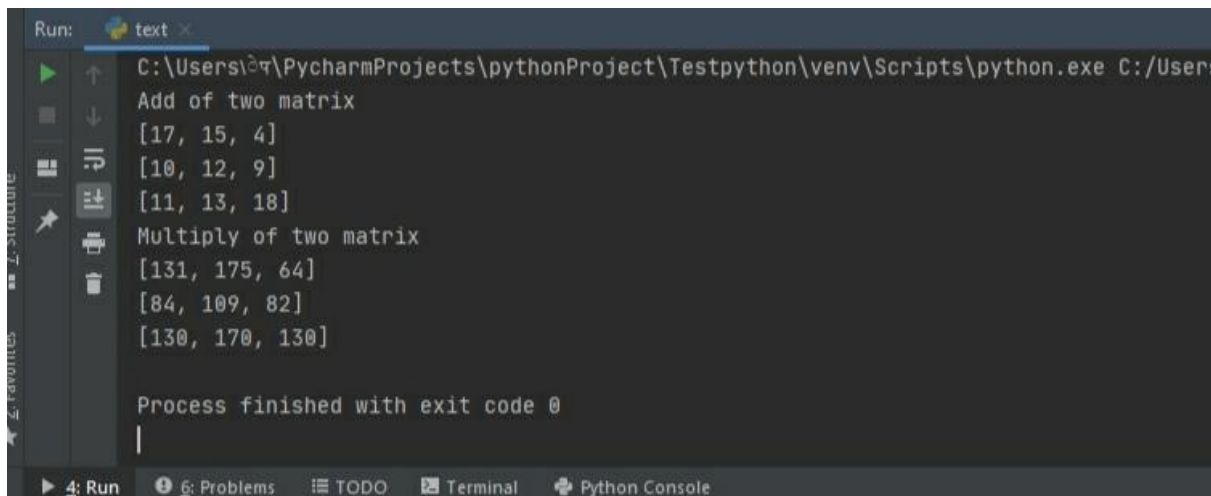
            result[i][j] += X[i][k] * Y[k][j]

for r in result:

    print(r)

```

## OUTPUT



```

Run: text x
C:\Users\B̄\PycharmProjects\pythonProject\Testpython\venv\Scripts\python.exe C:/Users
Add of two matrix
[17, 15, 4]
[10, 12, 9]
[11, 13, 18]
Multiply of two matrix
[131, 175, 64]
[84, 109, 82]
[130, 170, 130]

Process finished with exit code 0
|

```

## RESULT

The given python programming is successfully executed.



## **EXPERIMENT: 3**

### **AIM**

Write a program in Python to check a number for Armstrong?

### **ALGORITHM**

Step 1: Start a program

Step 2: Declare the variable num and sum

Step 3: The digits is raised to the power power of the number of digits and stored

Step 4: Chech if the sum obtained is same as the original number

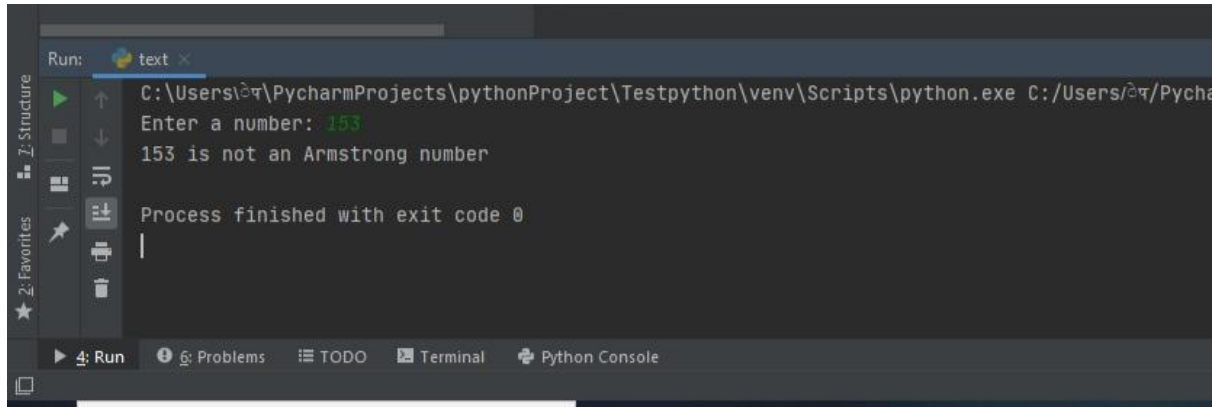
Step 5: Print the result

Step 6: End the program

### **PROGRAM**

```
# input from the user
num = int(input("Enter a number: "))
sum = 0
# find the sum of the cube of each digit
temp = num
while temp > 0:
    digit = temp % 10
    sum += digit ** 8    #upto eight digit only
    temp //= 10
# showing result
if num == sum:
    print(num,"is an Armstrong number")
else:
    print(num,"is not an Armstrong number")
```

### **OUTPUT**



```
Run: text x
C:\Users\...\PycharmProjects\pythonProject\Testpython\venv\Scripts\python.exe C:/Users/.../Pycha
Enter a number: 153
153 is not an Armstrong number

Process finished with exit code 0
```

## RESULT

The given python programming is successfully executed

## **EXPERIMENT:4**

### **AIM**

Write a program in Python to print factorial of a number ?

### **ALGORITHM**

Step 1: Start the program

Step 2: Declare the variable of num and factorial

Step 3: Iterate from 1 to n using for loop

Step 4: Using the formula calculate the factorial

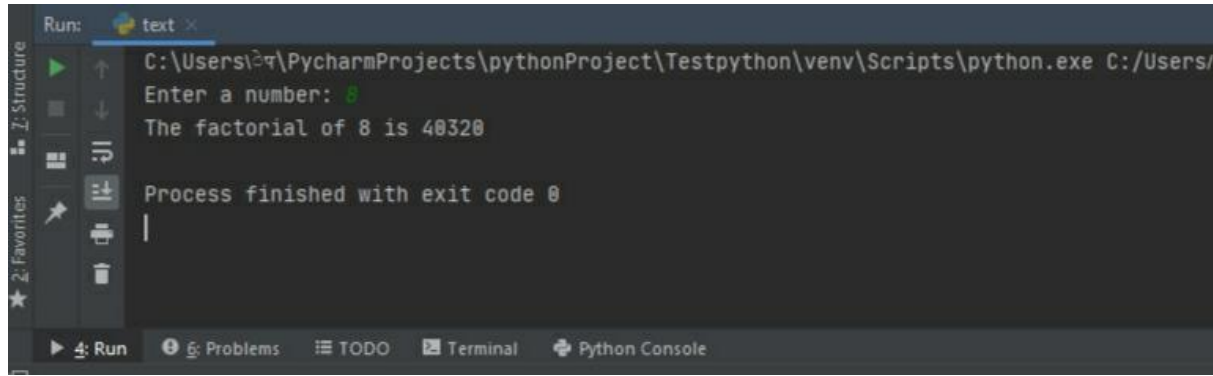
Step 5: Print the output

Step 6: End the program

### **PROGRAM**

```
# To take input from the user
num = int(input("Enter a number: "))
factorial = 1
if num < 0:      #if negative number given by user
    print("Sorry, factorial does not exist for negative numbers")
elif num == 0:   #if the number is equal to zero
    print("The factorial of 0 is 1")
else:           #if the numbers are positive
    for i in range(1,num + 1):
        factorial = factorial*i
    print("The factorial of",num,"is",factorial)
```

### **OUTPUT**



The screenshot shows the 'Run' window in PyCharm. The title bar indicates the file 'text'. The output text is as follows:

```
C:\Users\...PycharmProjects\pythonProject\Testpython\venv\Scripts\python.exe C:/Users/...
Enter a number: 8
The factorial of 8 is 40320
Process finished with exit code 0
```

On the left side of the window, there is a vertical toolbar with icons for Run (a green play button), Step Into (a magnifying glass), Step Over (a right arrow), Step Out (a left arrow), Toggle Breakpoint (a circle with a dot), and a trash can icon. The bottom status bar shows tabs for '4: Run', '6: Problems', 'TODO', 'Terminal', and 'Python Console'.

## RESULT

The given python programming is successfully executed

## **EXPERIMENT :5**

### **AIM**

Write a program in Python to generate first n Fibonacci terms recursively ?

### **ALGORITHM**

Step 1: Start the program

Step 2: Declare the variable of nterms and count,n1,n2

Step 3: check whether the nterms are positive number

Step 4: Make nterms in Fibonacci number

Step 5: Print the output

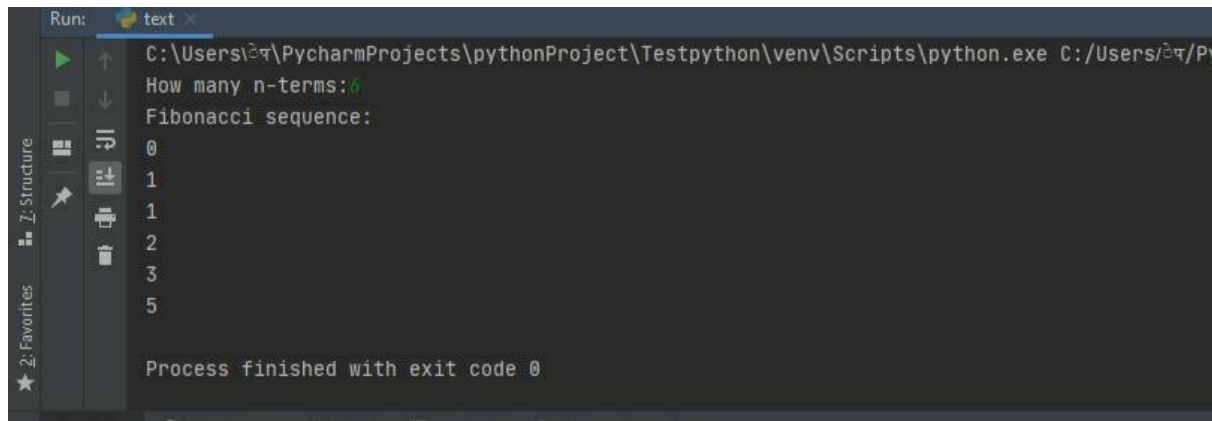
Step 6: End the program

### **PROGRAM**

```
nterms = int(input("How many n-terms:"))
# first two terms
n1, n2 = 0, 1
count = 0

# check if the number of terms is valid
if nterms <= 0:
    print("Please enter a positive integer")
elif nterms == 1:
    print("Fibonacci sequence upto",nterms,":")
    print(n1)
else:
    print("Fibonacci sequence:")
    while count < nterms:
        print(n1)
        nth = n1 + n2
        n1 = n2
        n2 = nth
        count += 1
```

### **OUTPUT**



```
Run: text x
C:\Users\देव\PycharmProjects\pythonProject\Testpython\venv\Scripts\python.exe C:/Users/देव/P
How many n-terms:6
Fibonacci sequence:
0
1
1
2
3
5

Process finished with exit code 0
```

## RESULT

The given python programming is successfully executed

## **EXPERIMENT:6**

### **AIM**

Write a program in Python to compute factorial of an integer n recursively?

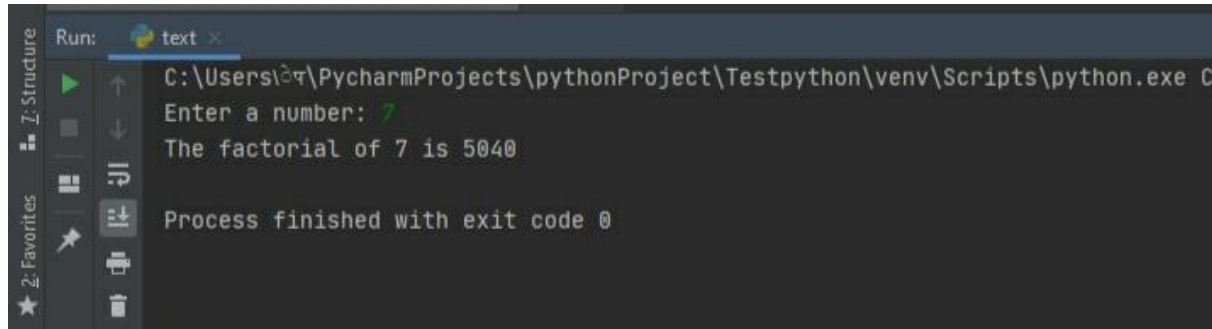
### **ALGORITHM**

- Step 1: Start a program
- Step 2: Declare the variable num
- Step 3: Check whether the num is greater than 1
- Step 4: Multiply the n with recur factorial
- Step 5: Print the output
- Step 6: End the program

### **PROGRAM**

```
def recur_factorial(n):  
    if n == 1:  
        return n  
    else:  
        return n*recur_factorial(n-1)  
  
num = int(input("Enter a number: "))  
  
# check if the number is negative  
if num < 0:  
    print("Sorry, factorial does not exist for negative numbers")  
elif num == 0:  
    print("The factorial of 0 is 1")  
else:  
    print("The factorial of", num, "is", recur_factorial(num))
```

### **OUTPUT**

A screenshot of the PyCharm Run console. The console title bar shows 'Run: text x'. The output text is as follows:

```
C:\Users\dev\PycharmProjects\pythonProject\Testpython\venv\Scripts\python.exe C
Enter a number: 7
The factorial of 7 is 5040

Process finished with exit code 0
```

The left sidebar of the IDE is visible, showing 'Structure' and 'Favorites' tabs.

## RESULT

The given python programming is successfully executed