# **1. Write a Python program to perform arithmetic operations on integers.**

**Code:**

import introJitendra

introJitendra.printIntro("Arithmetic Operators in python")

a = int(input("first number: "))

b = int(input("second number: "))

print("Sum: ", a+b)

print("Difference: ", a-b)

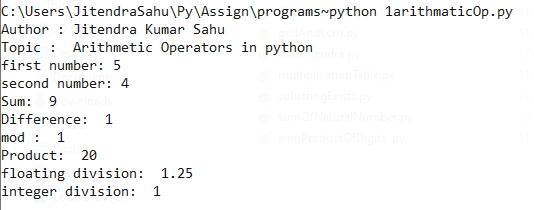
print("mod : ", a-b)

print("Product: ", a\*b)

print("floating division: ", a/b)

print("integer division: ", a//b)

**Output:**

****

# **2. Write a Python program to check and print the types of at least 5 different inbuilt objects.**

**Code:**

# check and print the types of at least 5 different inbuilt objects

import introJitendra

introJitendra.printIntro("check and print the types of at least 5 different inbuilt objects")

a = 5

print('a = ',a)

print('type of a = ',type(a))

a = 34.5

print('a = ',a)

print('type of a = ',type(a))

a = "jitendra sahu"

print('a = ',a)

print('type of a = ',type(a))

a = [6,4,6,6]

print('a = ',a)

print('type of a = ',type(a))

a = (5,67,8,"jitu")

print('a = ',a)

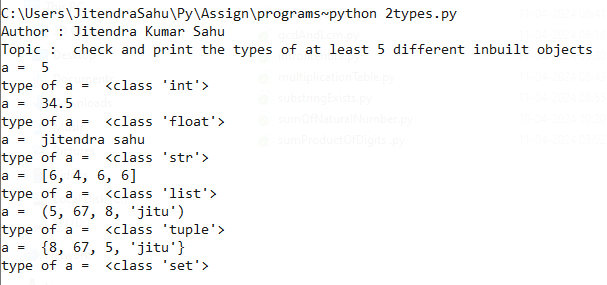
print('type of a = ',type(a))

a = {5,67,8,"jitu"}

print('a = ',a)

print('type of a = ',type(a))

**Output:**

****

# **3. Write a Python program to check if a number is EVEN or ODD.**

**Code:**

# check if a number is EVEN or ODD

import introJitendra

introJitendra.printIntro(" check number is even or odd")

num = int(input("Enter a number: "))

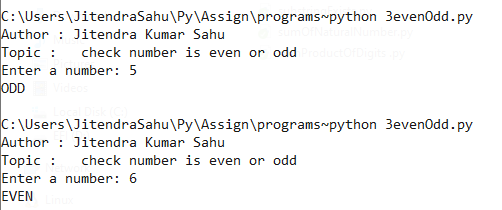
if num % 2 == 0:

print("EVEN")

else:

print("ODD")

**Output:**

****

4. Write a Python program to check if a number is Positive, Negative or Zero.

**Code:**# check if a number is Positive, Negative or Zero

import introJitendra

introJitendra.printIntro(" check if a number is Positive, Negative or Zero")

num = int(input("Enter a number: "))

if num > 0:

print("Positive")

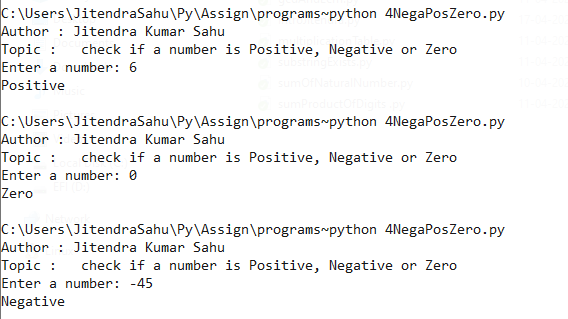
elif num < 0:

print("Negative")

else:

print("Zero")

**Output:**

**** 5. Write a Python program to check if a number is PRIME or NOT.

**Code:**

# check if a number is PRIME or NOT

import introJitendra

introJitendra.printIntro(" check if a number is PRIME or NOT")

num = int(input("Enter a number: "))

if num > 1:

for i in range(2, num):

if (num % i) == 0:

print(num, "is not a prime number")

break

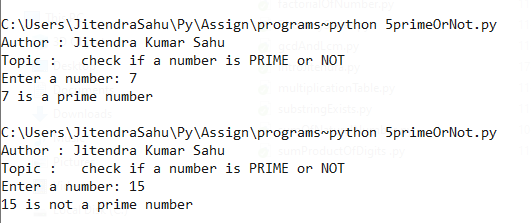
else:

print(num, "is a prime number")

else:

print(num, "is not a prime number")

**Output:**

****

# **6. Write a Python program to check whether a string entered by the user is a valid decimal number or not.**

**Code:**

# check whether a string entered by the user is a valid decimal number or not

import introJitendra

introJitendra.printIntro(" check whether a string entered by the user is a valid decimal number or not")

s = input("Enter a string: ")

isDecimal = True

for i in s :

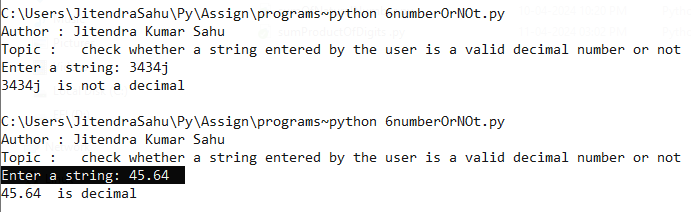
if not((i >= '0' and i <= '9') or i=='.'):

isDecimal = False

if isDecimal : print(s , " is decimal")

else : print(s , " is not a decimal")

**Output:**

****

# **7. Write a Python program to check if a year entered by the user is a Leap Year or NOT.**

**Code:**# check if a year entered by the user is a Leap Year or NOT

import introJitendra

introJitendra.printIntro(" check if a year entered by the user is a Leap Year or NOT")

year = int(input("Enter a year: "))

if (year % 4) == 0:

if (year % 100) == 0:

if (year % 400) == 0:

print(year," is a leap year")

else:

print(year," is not a leap year")

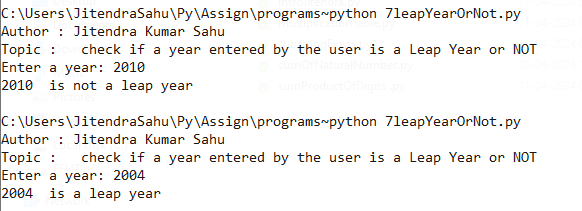
else:

print(year," is a leap year")

else:

print(year," is not a leap year")

**Output:**

****

# **8. Write a Python program to check whether a string entered by the user is a palindrome or not.**

**Code:**

# check and print if entered string is palindrome or not

import introJitendra

introJitendra.printIntro("check and print if entered string is palindrom or not")

s=input("enter the string to be checked : ") ;

isPalindrome=True

for i in range(len(s)//2):

if s[i] != s[-(i+1)] :

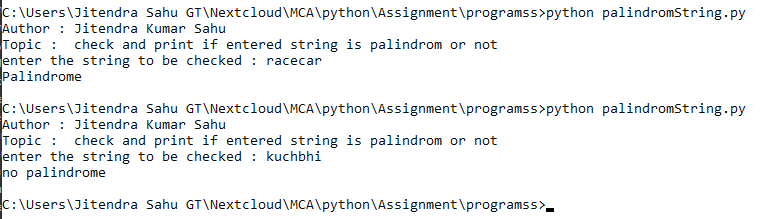
print("no palindrome")

isPalindrome=False ;

break ;

if isPalindrome : print("Palindrome")

**Output:**

****

# **9. Write a Python program to get a Decimal number from user and convert it into Binary, Octal and Hexadecimal.**

**Code:**

#Convert a Decimal number into Binary, Octal and Hexadecimal:

import introJitendra

introJitendra.printIntro("Convert a Decimal number into Binary, Octal and Hexadecimal")

def convert(num,base) :

bn = ''

while(num>0):

dig = num % base

if(dig<10):

bn = bn + str(dig)

else :

alpha = chr(65 + dig%10)

bn = bn + alpha

num //= base

# printing the number

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

print (bn[i],end='') ;

print('')

num = int(input("Enter number to be converted : "))

print("Binary : ",end='')

convert(num=num,base=2)

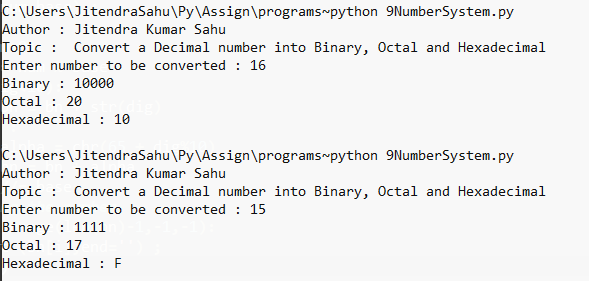
print("Octal : ",end='')

convert(num=num,base=8)

print("Hexadecimal : ",end='')

convert(num=num,base=16)

**Output:**

****

# **10. Write a Python program to find sum of natural numbers, up to N.**

**Code:**

#Find sum of natural numbers, up to N:

import introJitendra

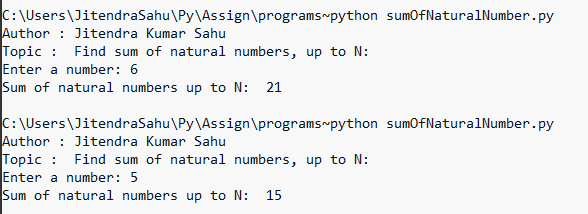
introJitendra.printIntro("Find sum of natural numbers, up to N:")

N = int(input("Enter a number: "))

sum = (N \* (N + 1)) // 2

print("Sum of natural numbers up to N: ", sum)

**Output:**

****

# **11. Write a Python program to get marks in five subjects from user and calculate average marks, percentage and grade of a student.**

**Code:**

# Calculate average marks, percentage and grade of a student:

import introJitendra

introJitendra.printIntro("Calculate average marks, percentage and grade of a student:")

n = 5

total = 0

for i in range(n):

m = int(input(f"Enter marks for subject {i+1}: "))

total += m

average = total / n

percentage = (total / (n \* 100)) \* 100

print("Average Marks: ", average)

print("Percentage: ", percentage)

# Grade

print("Grade : ", end="")

if percentage >= 90 : print("A+")

elif percentage >= 80 : print("A")

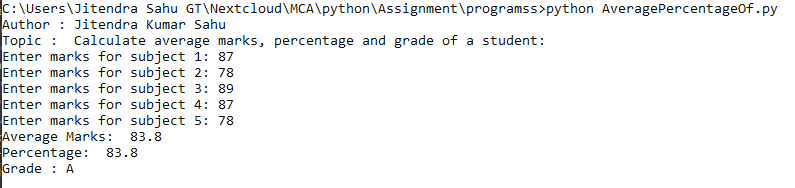
elif percentage >= 70 : print("B+")

elif percentage >= 60 : print("B")

elif percentage >= 50 : print("C")

else : print("D")

**Output:**

****

# **12. Write a Python program to get a number and find the sum and product of its digits.**

**Code:**#Find the sum and product of digits of a number:

import introJitendra

introJitendra.printIntro("Find the sum and product of digits of a number:")

num = input("Enter a number: ")

sum\_of\_digits = 0

for i in num :

sum\_of\_digits += int(i)

product\_of\_digits = 1

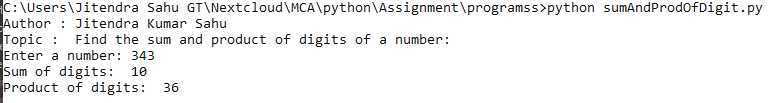
for digit in num:

product\_of\_digits \*= int(digit)

print("Sum of digits: ", sum\_of\_digits)

print("Product of digits: ", product\_of\_digits)

**Output:**

****

# **13. Write a Python program to get two integers and find their GCD and LCM.**

**Code:**

#Find the sum and product of digits of a number:

import introJitendra

introJitendra.printIntro("GCD and LCM")

def getGCD(a,b):

while(b!=0):

temp = a

a = b

b = temp % b

return a

def getLCM(a,b):

max = a if a>b else b

while True :

if(max%a==0) and (max%b==0): break

max+=1

return max ;

x=int(input("enter namber 1 : "))

y=int(input("enter namber 2 : "))

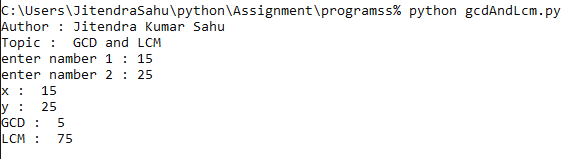
print("x : ",x)

print("y : ",y)

print("GCD : ",getGCD(x,y))

print("LCM : ",getLCM(x,y))

**Output:**

****

# **14. Write a Python program to find factorial of a number using while loop.**

**Code:**

# factorial of a number

import introJitendra

introJitendra.printIntro("factorial of a number")

num = int(input("enter the number : "))

t = num

fact = 1

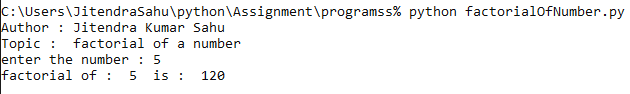
while num >= 1:

fact \*= num

num -= 1

print("factorial of : ", t, " is : ", fact)

**Output:**

****

# **15. Write a Python program to print Fibonacci series up to N terms.**

**Code:**

# Fibonacci series up to N terms:

import introJitendra

introJitendra.printIntro("Fibonacci series up to N terms:")

def fibonacci(n):

a, b = -1, 1

while n > 0:

a, b = b, a + b

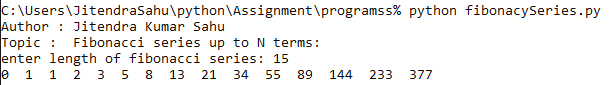
print(b, " ", end="")

n -= 1

n = int(input('enter length of fibonacci series: '))

fibonacci(n)

**Output:**

****

# **16. Write a Python program to print multiplication table.**

**Code:**

# Multiplication table:

import introJitendra

introJitendra.printIntro("Multiplication table:")

def multiplication\_table(n):

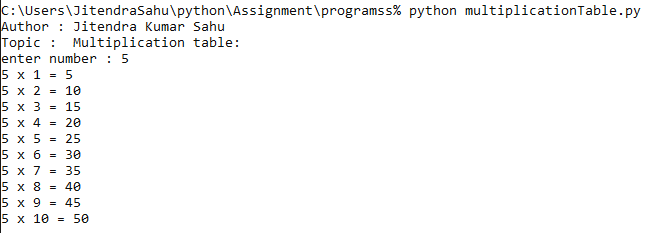
for i in range(1, 11):

print(n, 'x', i, '=', n\*i)

n = int(input('enter number : '))

multiplication\_table(n)

**Output:**

****

# **17. Write a Python program to access each element of a string in forward and backward orders using the ‘while’ loop.**

**Code:**

# Access each element of a string in forward and backward orders using the 'while' loop:

import introJitendra

introJitendra.printIntro("Access each element of a string in forward and backward orders using the 'while' loop:")

def access\_string(s):

i = 0

while i < len(s):

print(s[i],end='')

i += 1

i = len(s) - 1

print('')

while i >= 0:

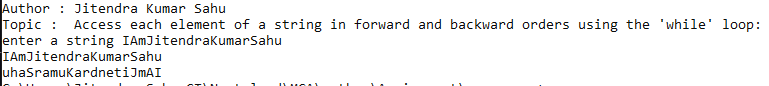
print(s[i],end='')

i -= 1

str1 = input("enter a string ")

access\_string(str1)

**Output:**

****

# **18. Write a Python program to access each element of a string in forward and backward orders using the ‘for’ loop.**

**Code:**

# Access each element of a string in forward and backward orders using the 'while' loop:

import introJitendra

introJitendra.printIntro("Access each element of a string in forward and backward orders using the 'while' loop:")

def access\_string(s):

i = 0

while i < len(s):

print(s[i],end='')

i += 1

i = len(s) - 1

print('')

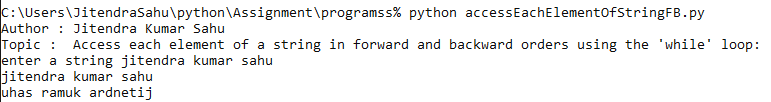
while i >= 0:

print(s[i],end='')

i -= 1

str1 = input("enter a string ")

access\_string(str1)

**Output:   
**

# **19. Write a Python program to find whether a substring exists in main string or not.**

**Code:**

# Find whether a substring exists in the main string or not:

import introJitendra

introJitendra.printIntro("Find whether a substring exists in the main string or not")

def substring\_exists(s, sub):

if sub in s:

return True

else:

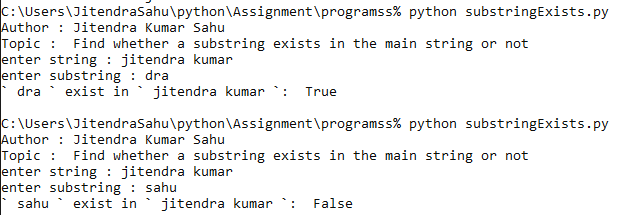
return False

s = input("enter string : ")

sb = input("enter substring : ")

print("`", sb, "` exist in `", s, "`: ", substring\_exists(s, sb))

**Output:**

****

# **20. Write a Python program to find the first occurrence of a substring in the main string.**

**Code:**

# program to count the number of times a substring appears in the main string

import introJitendra

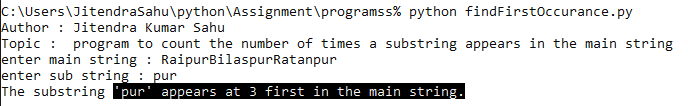
introJitendra.printIntro("program to count the number of times a substring appears in the main string")

mainString = input("enter main string : ")

substring = input("enter sub string : ")

occurrences = mainString.find(substring)

print(f"The substring '{substring}' appears at {occurrences} first in the main string.")

**Output:   
**

# **21. Write a Python program to count the number of times a substring appears in the main string.**

**Code:**

# program to count the number of times a substring appears in the main string

import introJitendra

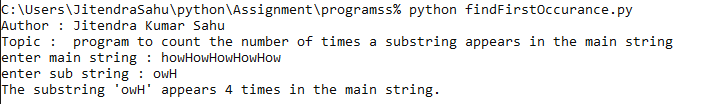
introJitendra.printIntro("program to count the number of times a substring appears in the main string")

mainString = input("enter main string : ")

substring = input("enter sub string : ")

occurrences = mainString.count(substring)

print(f"The substring '{substring}' appears {occurrences} times in the main string.")

**Output:   
**

# **22. Write a Python program to demonstrate the use of all “casing” methods and display a string in different cases.**

**Code:**

# demonstrate the use of all **‘**casing**’** methods and display a string in different cases.

import introJitendra

introJitendra.printIntro("demonstrate the use of all **‘**casing**’** methods and display a string in different cases.")

text = input("enter string : ")

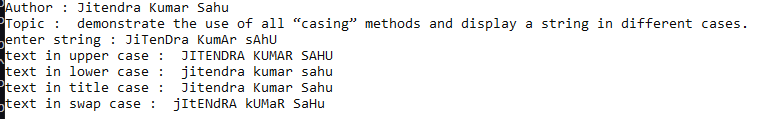
print("text in upper case : ",text.upper())

print("text in lower case : ",text.lower())

print("text in title case : ",text.title())

print("text in swap case : ",text.swapcase())

**Output:**

****

# **23. Write a Python program to demonstrate the use of all string testing {isXXX()} methods.**

**Code:**

# demonstrate the use of all “casing” methods and display a string in different cases.

import introJitendra

introJitendra.printIntro("demonstrate the use of all “casing” methods and display a string in different cases.")

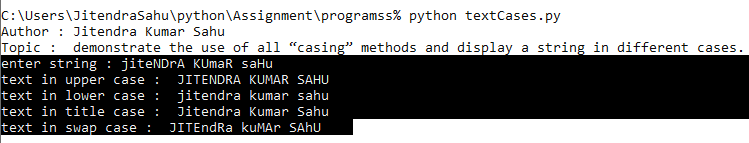
text = input("enter string : ")

print("text in upper case : ",text.upper())

print("text in lower case : ",text.lower())

print("text in title case : ",text.title())

print("text in swap case : ",text.swapcase())

**Output:   
**

# **24. Write a Python function to take a list of integers as input and return the average.**

**Code:**

#Python function to take a list of integers as input and return the averageimport introJitendra

import introJitendra

introJitendra.printIntro("Python function to take a list of integers as input and return the average")

def calculate\_average(numbers):

total = 0

for i in numbers:

total += i

return total / len(numbers)

num\_list = list()

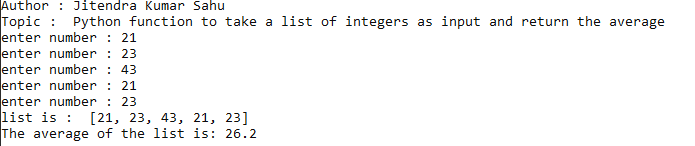
for i in range(5):

num\_list.append(int(input("enter number : ")))

print("list is : ", num\_list)

avg = calculate\_average(num\_list)

print(f"The average of the list is: {avg}")

**Output:   
**

# **25. Write a Python function to take two distinct integers as input and print all prime numbers between them.**

**Code:**

# Python function print all prime numbers between two distinct number.

import introJitendra

introJitendra.printIntro("Python function print all prime numbers between two distinct number.")

def is\_prime(num):

if num < 2:

return False

for i in range(2, int(num\*\*0.5) + 1):

if num % i == 0:

return False

return True

def print\_primes\_between(start, end):

print(f"prime between {start} and {end} are : ")

for num in range(start, end + 1):

if is\_prime(num):

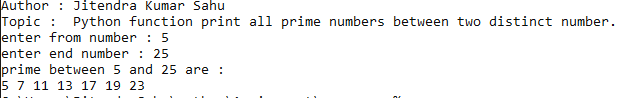
print(num, end=" ")

start\_num = int(input("enter from number : "))

end\_num = int(input("enter end number : "))

print\_primes\_between(start\_num, end\_num)

**Output:**

****

# **26. Write a Python function to take two integers as input and return both their sum and product.**

**Code:**

# Function to take two integers as input and return their sum and product

import introJitendra

introJitendra.printIntro("function takes two integers and return their sum and product.")

def sum\_and\_product(a, b):

return a + b, a \* b

x=int(input("enter num1 : ")) ;

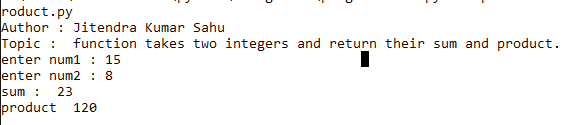
y=int(input("enter num2 : ")) ;

sum , prod = sum\_and\_product(x,y)

print("sum : " , sum)

print("product " , prod)

**Output:**

****

# **27. Write a Python program to demonstrate the positional arguments of a function.**

**Code:**

# Program to demonstrate positional arguments of a function

import introJitendra

introJitendra.printIntro("Python program to demonstrate the positional arguments of a function.")

def positional\_demo(x, y):

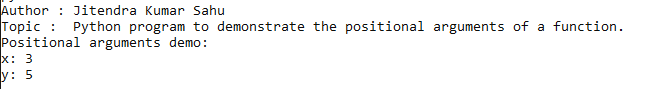
print("Positional arguments demo:")

print("x:", x)

print("y:", y)

positional\_demo(3, 5)

**Output:**

****

# **28. Write a Python program to demonstrate the keyword arguments of a function.**

**Code:**

# Program to demonstrate keyword arguments of a function

import introJitendra

introJitendra.printIntro("Python program to demonstrate the keyword arguments of a function.")

def keyword\_demo(x, y):

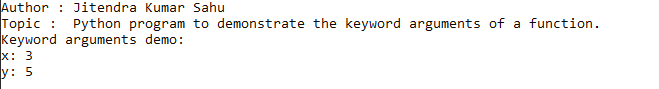
print("Keyword arguments demo:")

print("x:", x)

print("y:", y)

keyword\_demo(y=5, x=3)

**Output:**

****

# **29. Write a Python program to demonstrate the default arguments of a function.**

**Code:**

# Program to demonstrate default arguments of a function

import introJitendra

introJitendra.printIntro("Python program to demonstrate the default arguments of a function.")

def default\_demo(x=1, y=1):

print("Default arguments demo:")

print("x:", x)

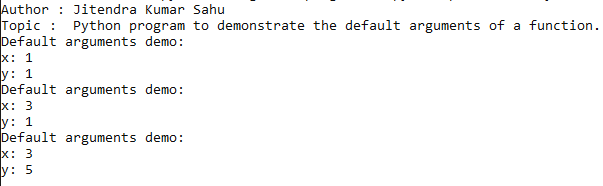
print("y:", y)

default\_demo()

default\_demo(3)

default\_demo(3, 5)

**Output:**

****

# **30. Write a Python function to demonstrate variable length arguments.**

**Code:**

# Function to demonstrate variable length arguments

import introJitendra

introJitendra.printIntro("Python function to demonstrate variable length arguments.")

def variable\_length\_args(\*args):

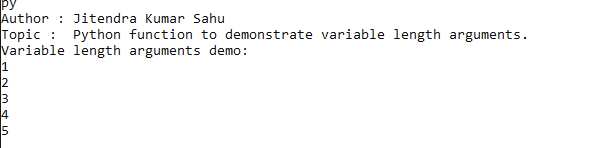
print("Variable length arguments demo:")

for arg in args:

print(arg)

variable\_length\_args(1, 2, 3, 4, 5)

**Output:**

****

# **31. Write a Python function to demonstrate keyword variable length arguments.**

**Code:**

# Function to demonstrate keyword variable length arguments

import introJitendra

introJitendra.printIntro("Python function to demonstrate keyword variable length arguments.")

def keyword\_variable\_length\_args(\*\*kwargs):

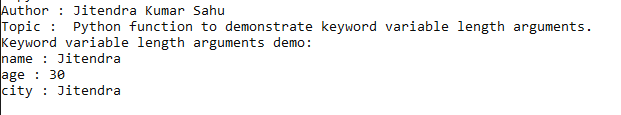
print("Keyword variable length arguments demo:")

for key, value in kwargs.items():

print(key, ":", value)

keyword\_variable\_length\_args(name="Jitendra", age=30, city="Jitendra")

**Output:**

****

# **32. Write a Python program to demonstrate global and local variables.**

**Code:**

# Program to demonstrate global and local variables

import introJitendra

introJitendra.printIntro("demonstrate global and local variables.")

global\_var = "I am a global variable"

def local\_demo():

local\_var = "I am a local variable"

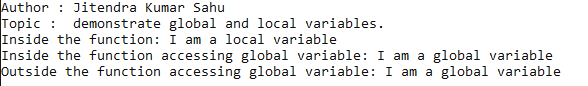
print("Inside the function:", local\_var)

print("Inside the function accessing global variable:", global\_var)

local\_demo()

print("Outside the function accessing global variable:", global\_var)

**Output:**

****

# **33. Write a Python function that takes an integer as input and calculates its factorial using recursion.**

**Code:**

# Python function that takes an integer as input and calculates its factorial using recursion

import introJitendra

introJitendra.printIntro("calculates factorial using recursion.")

def factorial(n):

if n == 0:

return 1

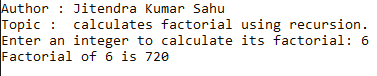
else:

return n \* factorial(n - 1)

num = int(input("Enter an integer to calculate its factorial: "))

print("Factorial of", num, "is", factorial(num))

**Output:**

****

# **34. Write a Python program to demonstrate the use of lambda functions.**

**Code:**

# Python program to demonstrate the use of lambda functions

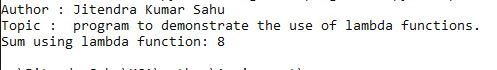
import introJitendra

introJitendra.printIntro("program to demonstrate the use of lambda functions.")

addition = lambda x, y : x + y

print("Sum using lambda function:", addition(3, 5))

**Output:**

****

# **35. Write a Python program to demonstrate the use of lambda functions and map.**

**Code:**

# Python program to demonstrate the use of lambda functions and map

import introJitendra

introJitendra.printIntro("program to demonstrate the use of lambda functions and map.")

nums = [1, 2, 3, 4, 5]

squared\_nums = list(map(lambda x: x\*\*2, nums))

print("Squared numbers:", squared\_nums)

**Output:**

****

# **36. Write a Python program to demonstrate the use of lambda functions and reduce.**

**Code:**

# Python program to demonstrate the use of lambda functions and reduce

import introJitendra

from functools import reduce

introJitendra.printIntro("program to demonstrate the use of lambda functions and reduce.")

nums = [1, 2, 3, 4, 5]

product = reduce(lambda x, y: x \* y, nums)

print("Product of numbers:", product)

**Output:**

****

# **37. Write a Python program to demonstrate the various list processing methods.**

**Code:**

# Python program to demonstrate the various list processing methods

import introJitendra

introJitendra.printIntro("program to demonstrate the various list processing methods.")

nums = [1, 2, 3, 4, 5]

print("Original list:", nums)

nums.append(6)

print("After appending 6:", nums)

nums.insert(2, 7)

print("After inserting 7 at index 2:", nums)

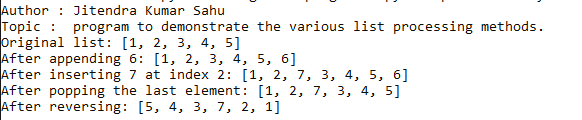
nums.pop()

print("After popping the last element:", nums)

nums.reverse()

print("After reversing:", nums)

**Output:**

****

# **38. Write a Python program to find the biggest and smallest numbers in a list of integers.**

**Code:**

# Python program to find the biggest and smallest numbers in a list of integers

import introJitendra

introJitendra.printIntro("program to find the biggest and smallest numbers in a list of integers.")

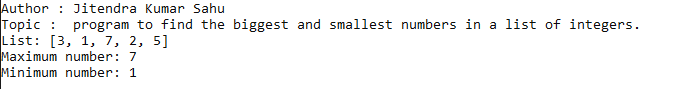
nums = [3, 1, 7, 2, 5]

print("List:", nums)

print("Maximum number:", max(nums))

print("Minimum number:", min(nums))

**Output:**

****

# **39. Write a Python program to find common elements in two lists.**

**Code:**

# Python program to find common elements in two lists

import introJitendra

introJitendra.printIntro("program to find common elements in two lists.")

list1 = list()

list2 = list()

print("enter 5 values for list 1 ")

for i in range(5):

list1.append(int(input()))

print("enter 5 values for list 2 ")

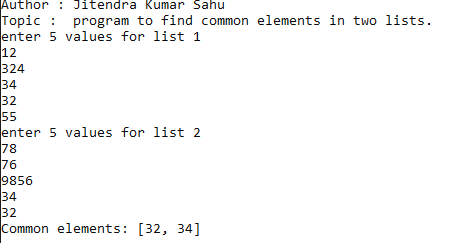
for i in range(5):

list2.append(int(input()))

common\_elements = list(set(list1) & set(list2))

print("Common elements:", common\_elements)

**Output:**

****

# **40. Write a Python program to demonstrate the various tuple processing methods.**

**Code:**

# Python program to demonstrate the various tuple processing methods

import introJitendra

introJitendra.printIntro("Python program to demonstrate the various tuple processing methods.")

my\_tuple = (1, 2, 3, 4, 5)

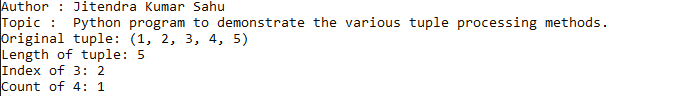
print("Original tuple:", my\_tuple)

print("Length of tuple:", len(my\_tuple))

print("Index of 3:", my\_tuple.index(3))

print("Count of 4:", my\_tuple.count(4))

**Output:**

****

# **41. Write a Python program to demonstrate the use of dictionaries.**

**Code:**

# Python program to demonstrate the use of dictionaries

import introJitendra

introJitendra.printIntro("Write a Python program to demonstrate the use of dictionaries.")

my\_dict = {'name': 'Alice', 'age': 30, 'city': 'New York'}

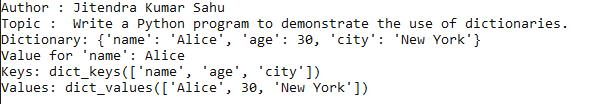
print("Dictionary:", my\_dict)

print("Value for 'name':", my\_dict['name'])

print("Keys:", my\_dict.keys())

print("Values:", my\_dict.values())

**Output:**

****

# **42. Write a Python program to find the number of occurrences of each letter in a string using dictionaries.**

**Code:**

# Python program to find the number of occurrences of each letter in a string using dictionaries

import introJitendra

introJitendra.printIntro("Write a Python program to find the number of occurrences of each letter in a string using dictionaries.")

string = input("Enter a string: ")

letter\_count = {}

for char in string:

if char in letter\_count:

letter\_count[char] += 1

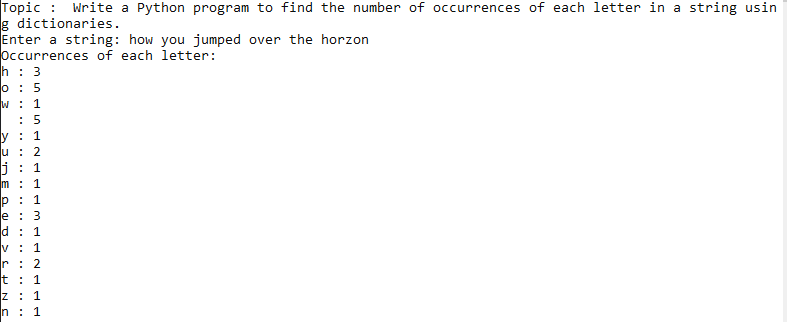
else:

letter\_count[char] = 1

print("Occurrences of each letter:")

for char, count in letter\_count.items():

print(char, ":", count)

**Output:   
**

# **43. Write a Python program to print the CWD and change the CWD.**

**Code:**

# Python program to print the CWD and change the CWD

import introJitendra

import os

introJitendra.printIntro("Write a Python program to print the CWD and change the CWD.")

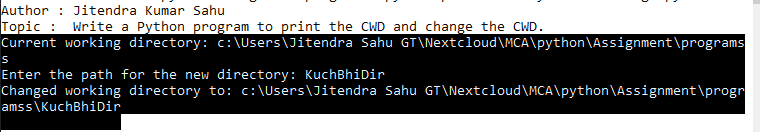
print("Current working directory:", os.getcwd())

new\_dir = input("Enter the path for the new directory: ")

os.chdir(new\_dir)

print("Changed working directory to:", os.getcwd())

**Output:**

****

# **44. Write a Python program that takes a list of words from the user and writes them into a file. The program should stop when the user enters the word ‘quit’.**

**Code:**

# Python program that takes a list of words from the user and writes them into a file

import introJitendra

introJitendra.printIntro("Write a Python program that takes a list of words from the user and writes them into a file. The program should stop when the user enters the word ΓÇÿquitΓÇÖ.")

words = []

while True:

word = input("Enter a word (or type 'quit' to stop): ")

if word == 'quit':

break

words.append(word)

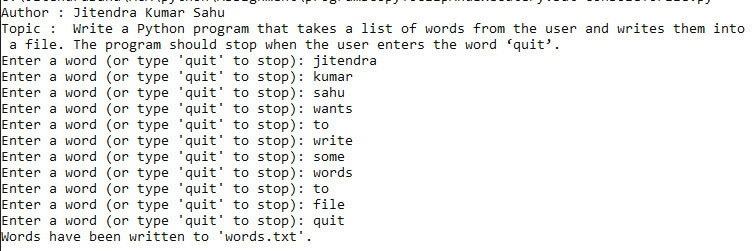
with open("words.txt", "w") as file:

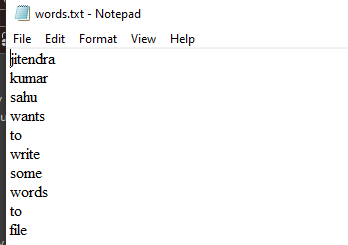
for word in words:

file.write(word + "\n")

print("Words have been written to 'words.txt'.")

**Output:**

****

****

# **45. Write a Python program that reads a file in text mode and counts the number of words that contain anyone of the letters [‘w’, ‘o’, ‘r’, ‘d’, ‘s’].**

**Code:**

# Python program that reads a file in text mode and counts the number of words that contain any one of the letters ['w', 'o', 'r', 'd', 's']

import introJitendra

introJitendra.printIntro("reads a file in text mode and \ncounts the number of words that\ncontain any one of the letters ['w', 'o', 'r', 'd', 's'].")

letters = set(['w', 'o', 'r', 'd', 's'])

word\_count = 0

with open("words.txt", "r") as file:

content = file.read();

print(content)

file.seek(0)

for line in file:

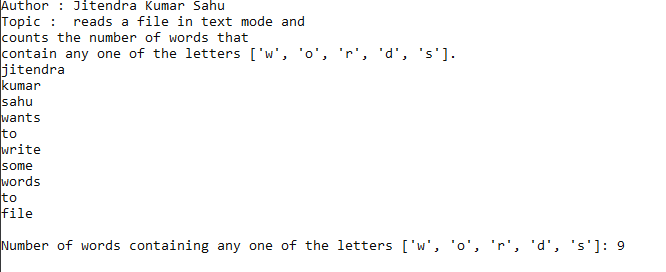
words = line.split()

for word in words:

if any(char in letters for char in word):

word\_count += 1

print("Number of words containing any one of the letters ['w', 'o', 'r', 'd', 's']:", word\_count)

**Output: **

# **46. Python programs to demonstrate the creation and use of “modules”.**

**Code:   
VEHICLE.PY**

class Vehicle :

def \_\_init\_\_(self,owner,numberOfWheels,type):

self.numberOfWheels=numberOfWheels

self.type=type

self.owner=owner

class Car(Vehicle):

def \_\_init\_\_(self,brand,model,owner):

super().\_\_init\_\_(type='CAR',numberOfWheels=4,owner=owner)

self.brand = brand

self.model = model

def print(self):

print(f"vehicle model : {self.model}")

print(f"vehicle type : {self.type}")

print(f"vehicle owner : {self.owner}")

print(f"number of wheels : {self.numberOfWheels}")

print(f"vehicle brand : {self.brand}")

**DRIVER.py**

import introJitendra

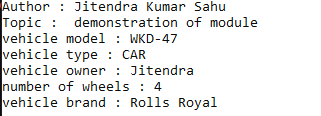
from Vehicle import Car

introJitendra.printIntro("demonstration of module")

c1 = Car("Rolls Royal","WKD-47","Jitendra") ;

c1.print()

**Output:**

****

# **47. Exception Handling Program that uses try and except.**

**Code:**

# Exception Handling Program that uses try and except

import introJitendra

introJitendra.printIntro("Exception Handling Program that uses try and except.")

try:

result = 10 / 0

except ZeroDivisionError:

print("Error: Division by zero occurred.")

**Output:**

****

# **48. Exception Handling Program that handles multiple types of exceptions.**

**Code:**

# Exception Handling Program that handles multiple types of exceptions

import introJitendra

introJitendra.printIntro("Exception Handling Program that handles multiple types of exceptions.")

try:

result = 10 / 'a'

except ZeroDivisionError:

print("Error: Division by zero occurred.")

except TypeError:

print("Error: Unsupported operation. Type mismatch.")

**Output:**

****

# **49. Exception Handling Program that uses try, except and else.**

**Code:**

# Exception Handling Program that uses try, except and else

import introJitendra

introJitendra.printIntro("Exception Handling Program that uses try, except and else.")

try:

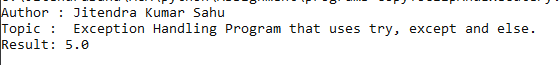
result = 10 / 2

except ZeroDivisionError:

print("Error: Division by zero occurred.")

else:

print("Result:", result)

**Output:   
**

# **50. Exception Handling Program that uses finally with try.**

**Code:**

# Exception Handling Program that uses finally with try

import introJitendra

introJitendra.printIntro("Exception Handling Program that uses finally with try.")

try:

result = 10 / 2

except ZeroDivisionError:

print("Error: Division by zero occurred.")

finally:

print("This will always execute, regardless of an exception.")

**Output:   
**

# **51. Write a Python program that creates a class “Person”, with attributes [aadhar, name, DoB]**

**Code:**# Python program that creates a class “Person”, with attributes [aadhar, name, DoB]

import introJitendra

introJitendra.printIntro("Write a Python program that creates a class “Person”, with attributes [aadhar, name, DoB].")

class Person:

def \_\_init\_\_(self, aadhar, name, dob):

self.aadhar = aadhar

self.name = name

self.dob = dob

def print(self):

print(f"name :{self.name}\ndob : {self.dob}\naadhar : {self.aadhar}\n")

name=input("enter name : ")

dob=input("enter dob : ")

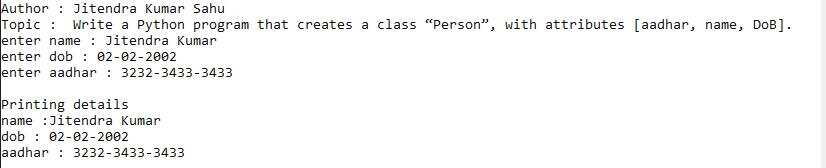
aadhar=input("enter aadhar : ")

p1 = Person(name=name,dob=dob,aadhar=aadhar)

print("\nPrinting details")

p1.print()

**Output:**

****

# **52. Write a Python program that creates classes “Point” and “Rectangle” where the Rectangle class has a Point object as its attribute.**

**Code:**

# Python program that creates classes “Point” and “Rectangle” where the Rectangle class has a Point object as its attribute

import introJitendra

introJitendra.printIntro("Write a Python program that creates classes “Point” and “Rectangle” where the Rectangle class has a Point object as its attribute.")

class Point:

def \_\_init\_\_(self, x, y):

self.x = x

self.y = y

def \_\_eq\_\_(self, other) -> bool:

return (self.x == other.x) and (self.y == other.y)

def printPoint(self):

print("x = ", self.x)

print("y = ", self.y)

class Rectangle:

def \_\_init\_\_(self, point1, point2):

self.point1 = point1

self.point2 = point2

def print(self):

self.point1.printPoint()

self.point2.printPoint()

x = float(input("enter x for point 1 : "))

y = float(input("enter y for point 1 : "))

p1 = Point(x, y)

x = float(input("enter x for point 2 : "))

y = float(input("enter y for point 2 : "))

p2 = Point(x, y)

if (not(p1==p2)):

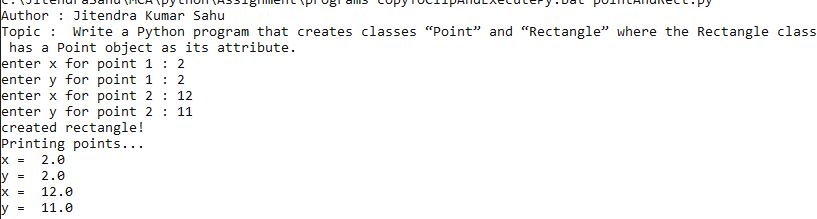
rect1 = Rectangle(p1,p2)

print("created rectangle!\nPrinting points...")

rect1.print()

else : print("both point should not be same")

**Output:**

****

# **53. Write a Python program that creates a class Students which inherits the properties of the “Person” class; add attributes [roll\_no, class].**

**Code:**

#Python program that creates a class Students which inherits the properties of the **‘**Person**’** class; add attributes [roll\_no, class]

import introJitendra

introJitendra.printIntro("Write a Python program that \ncreates a class Students which inherits the properties of \nthe **‘**Person**’** class; add attributes [roll\_no, class].")

class Person:

def \_\_init\_\_(self, name, age, city):

self.name = name

self.age = age

self.city = city

class Student(Person):

def \_\_init\_\_(self, name, age, city, roll\_no, class\_name):

super().\_\_init\_\_(name,age,city)

self.roll\_no = roll\_no

self.class\_name = class\_name

def print(self):

print("\nDetails of the student:")

print("Name:", self.name)

print("Roll:", self.roll\_no)

print("class:", self.class\_name)

print("Age:", self.age)

print("City:", self.city)

name = input("Enter the name of the student : ")

roll = input("Enter the roll of the student : ")

class\_name = input("Enter the class of the student : ")

age = int(input("Enter the age of the student : "))

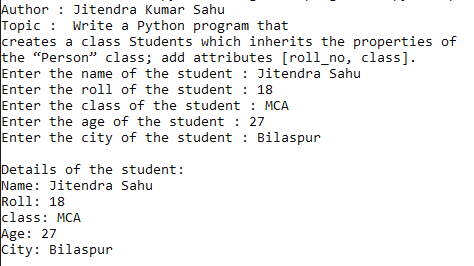
city = input("Enter the city of the student : ")

s1 = Student(name=name, age=age,city=city,roll\_no=roll,class\_name=class\_name)

# Print the details of the person

s1.print()

**Output:**

****

# **54. Write a Python program to demonstrate “Multiple Inheritance”.**

**Code:**

# Python program to demonstrate **‘**Multiple Inheritance**’**

import introJitendra

introJitendra.printIntro("Write a Python program to demonstrate **‘**Multiple Inheritance**’**.")

class A:

def show\_parent1(self):

print("Parent Father")

class B:

def show\_parent2(self):

print("Parent Mother")

class C(A, B):

def show\_child(self):

print("Am child\n")

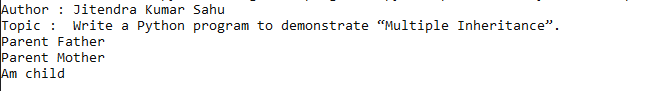
obj = C()

obj.show\_parent1()

obj.show\_parent2()

obj.show\_child()

**Output:**

****

# **55. Write a Python program to demonstrate “Method Overriding”.**

**Code:**

# Python program to demonstrate **‘**Method Overriding**’**

import introJitendra

introJitendra.printIntro("Write a Python program to demonstrate **‘**Method Overriding**’**.")

class Parent:

def show(self):

print("Parent's show method")

class Child(Parent):

def show(self):

print("Child's show method")

obj = Child()

obj.show()

**Output:**

****

# **56. Write a Python program to demonstrate “Method Overloading”.**

**Code:**

# Python program to demonstrate **‘**Method Overloading**’**

import introJitendra

introJitendra.printIntro("Write a Python program to demonstrate **‘**Method Overloading**’**.")

class Adder:

def total(self, a=None, b=None, c=None):

if c != None:

return a + b + c

if b != None:

return a + b

if a != None:

return a

# Create an instance of the class

obj = Adder()

# Call the sum method with different numbers of arguments

print("enter two numbers : ")

x = int(input())

y = int(input())

print(f"Sum of {x} and {y} = {obj.total(x, y)}")

print("enter three numbers : ")

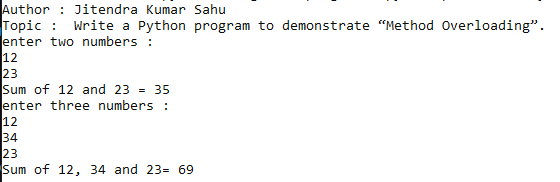
x = int(input())

y = int(input())

z = int(input())

print(f"Sum of {x}, {y} and {z}= {obj.total(x, y, z)}")

**Output:**

****

# **57. Write a Python program to demonstrate “Operator Overloading” [+ and -] using a class “Book”.**

**Code:**

# Python program to demonstrate **‘**Operator Overloading**’** [+ and -] using a class **‘**Book**’**

import introJitendra

introJitendra.printIntro("Write a Python program to demonstrate\n**’**Operator Overloading**’** [+ and -] using a class **‘**Book**’**.")

class Book:

def \_\_init\_\_(self, pages):

self.pages = pages

def \_\_add\_\_(self, other):

return self.pages + other.pages

def \_\_sub\_\_(self, other):

return self.pages - other.pages

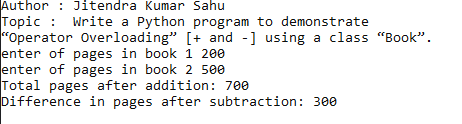
book1 = Book(int(input("enter of pages in book 1 ")))

book2 = Book(int(input("enter of pages in book 2 ")))

print("Total pages after addition:", book1 + book2)

print("Difference in pages after subtraction:", book2 - book1)

**Output:**

****

# **58. Use the “turtle” module to draw concentric circles with different colours.**

**Code:**

# Use the **‘**turtle**’** module to draw concentric circles with different colours.

import turtle

import introJitendra

introJitendra.printIntro("Use the **‘**turtle**’** module to draw concentric circles with different colours.")

colors = ['red', 'green', 'blue', 'orange', 'purple']

turtle.pensize(4)

for i in range(5):

turtle.color(colors[i])

turtle.penup()

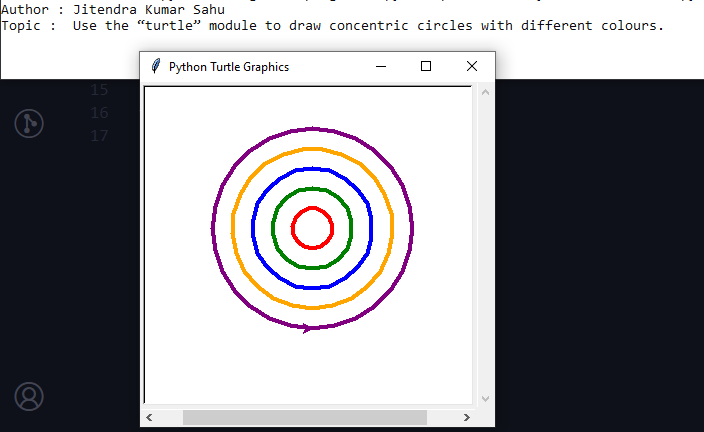
turtle.goto(0, -i \* 20)

turtle.pendown()

turtle.circle(20 + i \* 20)

turtle.done()

**Output:**

****

# **59. Use the “turtle” module to print the multiplication table.**

**Code:**

import turtle

# Create a turtle object

pen = turtle.Turtle()

# pen.speed(0) # Set the drawing speed to fastest

# Function to write text at a given position

def write\_text(x, y, text):

pen.penup()

pen.goto(x, y)

pen.pendown()

pen.write(text)

# Function to draw the multiplication table

def draw\_table(rows, cols):

for i in range(1, rows + 1):

for j in range(1, cols + 1):

# Calculate position

x = -200 + j \* 40

y = 200 - i \* 20

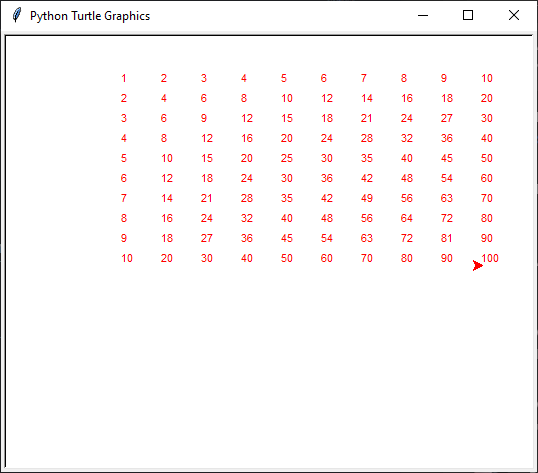
pen.color("red")

write\_text(x + 10, y - 10, str(i \* j))

draw\_table(10, 10)

turtle.mainloop()

**Output:**

****

# **60. Use the “turtle” module to draw (not write) your name.**

**Code:**

import turtle

screen = turtle.Screen()

screen.bgcolor("white")

pen = turtle.Turtle()

pen.pensize(5) # Set the pen size

def draw\_J():

pen.up()

pen.setheading(180)

pen.fd(200)

pen.fd(100)

pen.setheading(0)

pen.down()

pen.fd(80)

pen.rt(90)

pen.fd(160)

pen.circle(-40,180)

pen.fd(20)

pen.up()

pen.setheading(0)

pen.fd(120)

def draw\_I():

pen.up()

pen.fd(20)

pen.setheading(90)

pen.fd(140)

pen.setheading(180)

pen.down()

pen.fd(30)

pen.backward(90)

pen.fd(45)

pen.setheading(270)

pen.fd(200)

pen.setheading(180)

pen.fd(45)

pen.bk(90)

pen.up()

pen.bk(30)

def draw\_T():

pen.up()

pen.bk(35)

pen.setheading(90)

pen.fd(200)

pen.setheading(180)

pen.fd(45)

pen.down()

pen.bk(90)

pen.fd(45)

pen.setheading(270)

pen.fd(200)

pen.up()

pen.setheading(0)

pen.up()

pen.fd(60)

def draw\_U():

pen.setheading(90)

pen.fd(200)

pen.down()

pen.bk(150)

pen.down()

pen.circle(60,-180)

pen.undo()

pen.setheading(270)

pen.circle(60,-180)

pen.undo()

pen.circle(60,180)

pen.fd(150)

def drawSmiley():

pen.pensize(3)

pen.color('orange')

pen.setheading(90)

pen.up()

pen.setheading(0)

pen.fd(50)

pen.down()

pen.circle(80)

pen.up()

pen.setheading(90)

pen.fd(10)

pen.fd(30)

pen.fd(60)

pen.setheading(180)

pen.fd(30)

pen.fd(10)

pen.circle(5)

pen.down()

pen.circle(5)

pen.up()

pen.bk(60)

pen.bk(20)

pen.down()

pen.circle(5)

pen.setheading(270)

pen.up()

pen.fd(80)

pen.undo()

pen.fd(60)

pen.circle(-100,40)

pen.undo()

pen.setheading(180)

pen.down()

pen.circle(-100,40)

pen.undo()

pen.setheading(210)

pen.circle(-100,40)

pen.undo()

pen.circle(-80,50)

draw\_J()

pen.color('blue')

draw\_I()

pen.pensize(2)

pen.color('red')

draw\_T()

pen.pensize(5)

pen.color("green")

draw\_U()

drawSmiley()

pen.hideturtle()

screen.mainloop()

**Output:**

