**Program: 1**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a program to input roll no, student name, marks of physics, chemistry and maths out of 100. (0-100). Calculate total, percentage, calculate STATUS (pass, fail) if students scores above 40 in all the 3 subjects the STATUS should be pass otherwise fail. Calculate GRADE if STATUS is pass. Grade must be based on percentage value.**

**if percentage is above 70, then grade must be DISTINCTION**

**if percentage is above 60, then grade must be FIRST CLASS**

**if percentage is above 50, then grade must be SECOND CLASS**

**if percentage is above 40, then grade must be PASS CLASS**

**Input :-**

print('Please enter you Enrollment number and Name: ')

while True:

try:

id = int(input('Enrollment id: \n'))

except:

print("Alphabets or Special characters are not allowed in Enrollment Id.")

else:

break

while (len(str(id)) != 8):

print('Please enter a valid Enrolment Id: ')

id = int(input('Enrollment id: \n'))

name = input('Name: \n')

def validate(subject):

while (subject not in range(0, 101)):

print("Please enter valid marks\n")

subject = int(input())

validate.condition = True

validate.export = subject

return subject

print("Please enter your marks for following subjects: \n")

try:

maths = int(input('Maths: \n'))

except:

print("Alphabets or Special characters are not allowed.")

maths = int(input('Maths: \n'))

validate(maths)

if validate.condition == True:

maths = validate.export

try:

physics = int(input('Physics: \n'))

except:

print("Alphabets or Special characters are not allowed.")

physics = int(input('Physics: \n'))

validate(physics)

if validate.condition == True:

physics = validate.export

try:

chemistry = int(input('Chemistry: \n'))

except:

print("Alphabets or Special characters are not allowed.")

chemistry = int(input('Chemistry: \n'))

validate(chemistry)

if validate.condition == True:

chemistry = validate.export

total = maths + physics + chemistry

percentage = (total \* 100) / 300

def calstatus():

if (maths > 40 and physics > 40 and chemistry > 40):

calstatus.status = 'Pass'

else:

calstatus.status = 'Fail'

return calstatus.status

def calgrade():

if calstatus.status == 'Pass':

if percentage > 40:

calgrade.grade = 'PASS CLASS'

if percentage > 50:

calgrade.grade = 'SECOND CLASS'

if percentage > 60:

calgrade.grade = 'FIRST CLASS'

if percentage > 70:

calgrade.grade = 'DISTINCTION'

return calgrade.grade

else:

calgrade.grade = 'You have failed in one or more than one subject(s).\nTherefore your grade cannot be calculated.'

return calgrade.grade

calstatus()

calgrade()

print()

print()

print("Enrollment Id:", id)

print("Name:", name)

print()

print('Marks:')

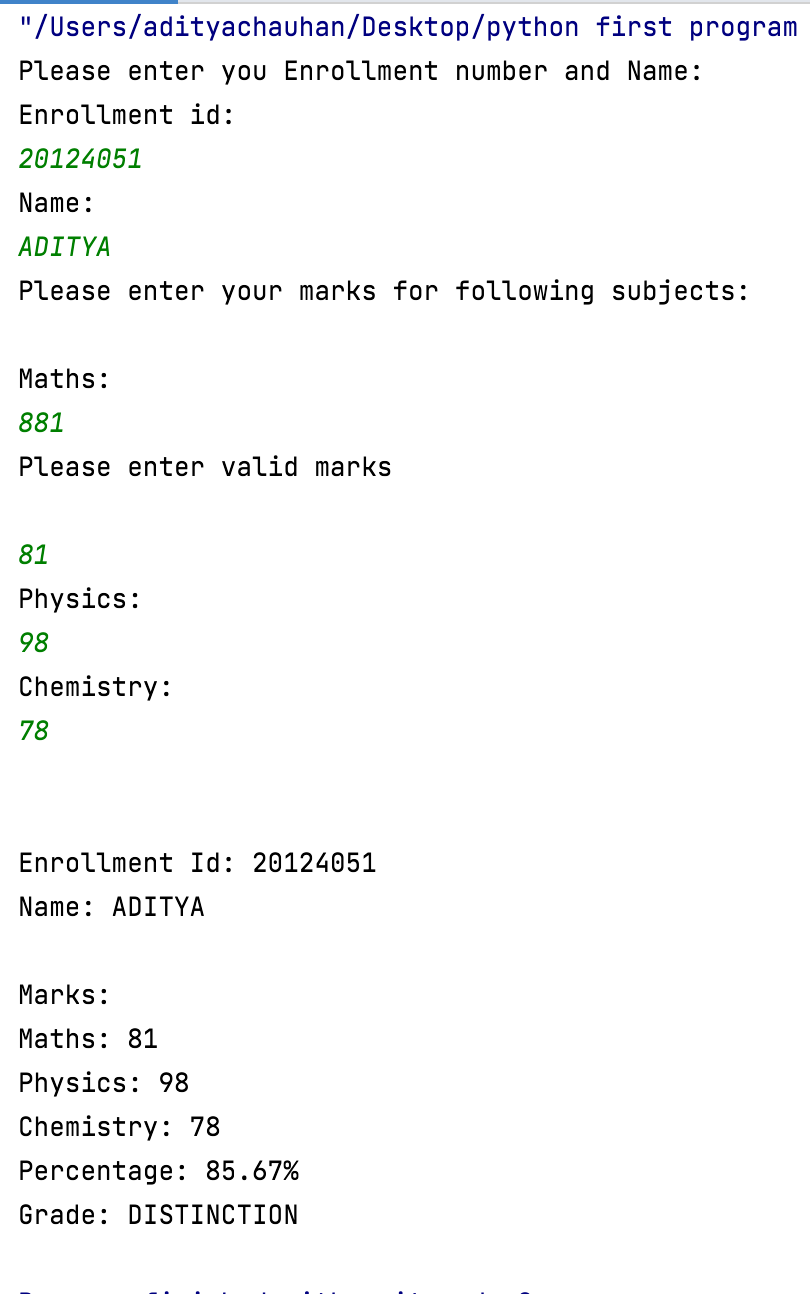
print('Maths:', maths)

print('Physics:', physics)

print('Chemistry:', chemistry)

print('Percentage: {0:.2f}%'.format(percentage))

print('Grade:', calgrade.grade)

**Output :-**

**Program: 2**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a program which inputs a number. Display that number in word format.**

**Eg.**

**459 – Four Five Nine**

**7091 – Seven Zero Nine One**

**26 - Two Six**

**Input :-**

class Word\_Format:

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

self.\_\_digit = input('Please enter a number: ')

def printValue(self, number):

if number == '0':

print("Zero ")

elif number == '1':

print("One ")

elif number == '2':

print("Two ")

elif number == '3':

print("Three")

elif number == '4':

print("Four ")

elif number == '5':

print("Five ")

elif number == '6':

print("Six ")

elif number == '7':

print("Seven")

elif number == '8':

print("Eight")

elif number == '9':

print("Nine ")

def printWord(self):

i = 0

num = self.\_\_digit

length = len(num)

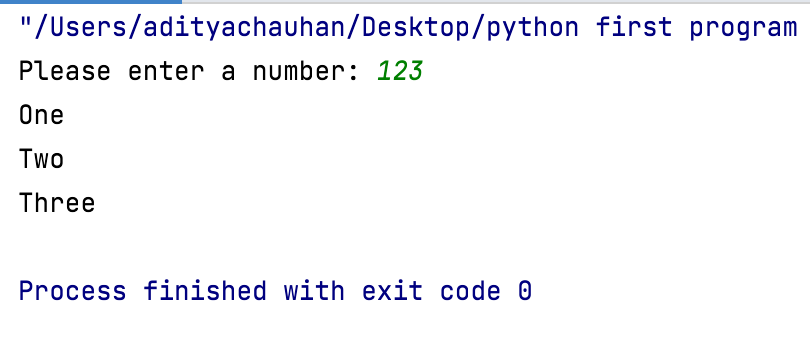
while i < length:

self.printValue(num[i])

i += 1

run = Word\_Format()

run.printWord()

**Output :-**

**Program: 3**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write an OOP to calculate exponent from inputted base and power value.**

**Eg. Enter a base value : 3**

**Enter a power value : 4**

**For base 3 and power 4, the answer is 81**

**Input :-**

class exponent:

def init(self):

self.\_\_num = 0

def get\_number(self, value):

self.\_\_num = value

def display(self):

print("The Answer for entered input is {0}".format(self.\_\_num))

def calculate(self, object):

solution = exponent()

solution.\_\_num = self.\_\_num \*\* object.\_\_num

return solution

base = exponent()

base.get\_number(5)

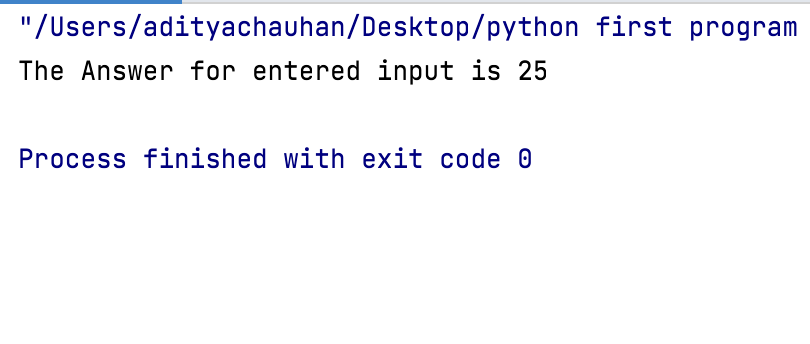
power = exponent()

power.get\_number(2)

solution = base.calculate(power)

solution.display()

**Output :-**

****

**Program:-4**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Program to print binary form a any number using 16 bit representation. (without library function) (You can use list for 16 bit representation)**

**Eg. Enter any number : 20**

**0000000000010100**

**Enter any number -5**

**1000000000000101**

**Input :-**

class bin\_representation:

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

self.\_\_string = ''

self.\_\_num = num

def convert(self):

i = 1 << 15

while (i > 0):

if ((self.\_\_num & i) != 0):

self.\_\_string += '1'

else:

self.\_\_string += '0'

i = i // 2

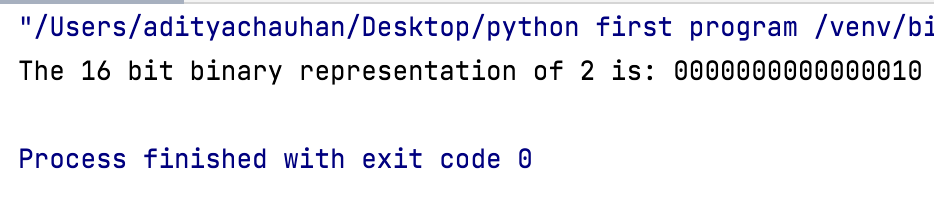
def display(self):

print('The 16 bit binary representation of {0} is: {1}'.format(self.\_\_num,self.\_\_string))

run = bin\_representation(2)

run.convert()

run.display()

**Output :-**

**Program:-5**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a OOP in python to input empid, name, basic salary, no. of experience in yrs. Calculate hra(35% of basic), da (58% of basic) and pf (9.5% of basic).**

**Also calculate bonus based on experience in years.**

**If experience in years is >= 30, bonus must be 59% of basic,**

**If experience in years is >=23, bonus must be 51% of basic,**

**If experience in years is >=15, bonus must be 45% of basic,**

**If experience in years is >=7, bonus must be 33% of basic,**

**If experience in years is <7, bonus must be 16% of basic**

**Calculate netsalary as basic+da+hra-pf+bonus.**

**Input :-**

class Employee:

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

self.\_\_empid = 0

self.\_\_name = ''

self.\_\_basic\_sal = 0

self.\_\_experience = 0

def get\_input(self):

while True:

try:

self.\_\_empid = int(input('Please enter Employee Id: '))

if (len(str(self.\_\_empid)) != 8):

print('Please enter a valid Employee Id: ')

self.\_\_empid = int(input('Please enter Employee Id: '))

else:

pass

except:

print("Alphabets or Special characters are not allowed in Employee Id.")

else:

break

self.\_\_name = input('Please enter your name: ')

while True:

try:

self.\_\_basic\_sal = int(input('Please enter your basic salary: '))

except:

print("Alphabets or Special characters are not allowed in Salary.")

else:

break

while True:

try:

self.\_\_experience = int(input('Please enter your experience: '))

except:

print("Alphabets or Special characters are not allowed in Salary.")

else:

break

def calculate(self):

self.\_\_hra = (self.\_\_basic\_sal \* 35) /100

self.\_\_da = (self.\_\_basic\_sal \* 58) /100

self.\_\_pf = (self.\_\_basic\_sal \* 9.5) /100

def cal\_bonus(self):

exp = self.\_\_experience

self.\_\_bonus = 0

if exp >= 30:

self.\_\_bonus = 59

elif exp >= 23:

self.\_\_bonus = 51

elif exp >= 15:

self.\_\_bonus = 45

elif exp >= 7:

self.\_\_bonus = 33

elif exp < 7:

self.\_\_bonus = 16

self.\_\_calculated\_bonus = (self.\_\_basic\_sal \* self.\_\_bonus) /100

def net\_salary(self):

self.\_\_netsalary = self.\_\_basic\_sal + self.\_\_da + self.\_\_hra - self.\_\_pf + self.\_\_calculated\_bonus

def display(self):

print("\n")

print("Employee Id: ",self.\_\_empid)

print("Name: ",self.\_\_name)

print("Basic salary: ",self.\_\_basic\_sal)

print("No. of Experience: ",self.\_\_experience)

print("HRA: ",self.\_\_hra)

print("DA: ",self.\_\_da)

print("PF: ",self.\_\_pf)

print("Bonus: ",self.\_\_calculated\_bonus)

print("Net Salary: ",self.\_\_netsalary)

run = Employee()

run.get\_input()

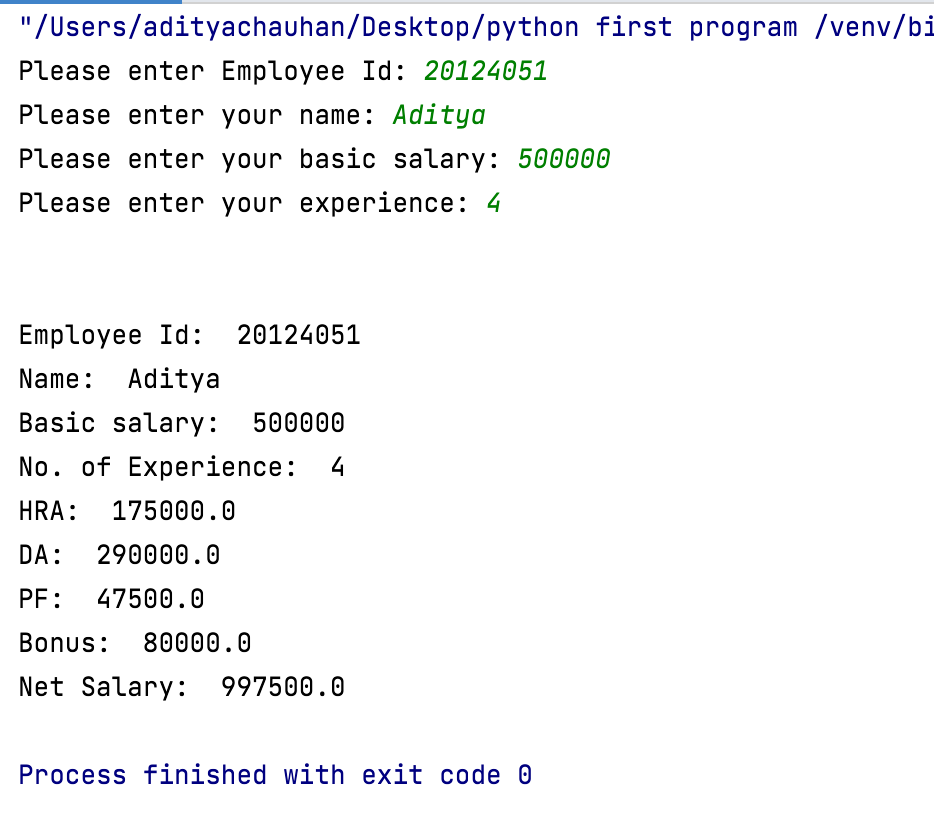
run.calculate()

run.cal\_bonus()

run.net\_salary()

run.display()

**Output :-**

****

**Program:-6**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a OOP program to input Customer id , Customer name, electricity unit charges used.**

**Calculate electricity bill according to the given condition:**

**For first 50 units Rs. 0.50/unit**

**For next 100 units Rs. 0.75/unit**

**For next 100 units Rs. 1.20/unit**

**For unit above 250 Rs. 1.50/unit**

**An additional surcharge of 20% is added to the bill**

**Input :-**

class ElectricityBill:

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

self.\_\_cusid = 0

self.\_\_name = ''

self.\_\_unit\_used = 0

self.\_\_tunit = 0

def get\_input(self):

self.\_\_cusid = int(input("Enter Customer Id : "))

self.\_\_name = input("Enter Customer Name : ")

self.\_\_unit\_used = int(input("Enter Customer Used Units : "))

def calculate(self):

self.\_extra = (self.\_\_unit\_used \* 0.2 )

def condition(self):

unit = self.\_\_unit\_used

if(unit <= 50):

self.\_\_tunit = unit\*0.5

elif(unit <= 100):

self.\_\_tunit = (50\*0.5) + ((unit-50)\*0.75)

elif (unit <= 150):

self.\_\_tunit = (50\*0.5) + (50\*0.75) + ((unit-100) \* 1.2)

elif (unit <= 250):

self.\_\_tunit = (50\*0.5) +(50\*0.75)+(100\*1.2)+((unit-150) \* 1.5)

elif (unit >= 250):

self.\_\_tunit = (50\*0.5) + (50\*0.75)+(100\*1.2)+((unit - 150) \* 2.3)

def net\_units(self):

self.\_net\_units = (self.\_\_tunit + self.\_extra)

def display(self):

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

print("Customer Id: ", self.\_\_cusid)

print("Customer Name: ", self.\_\_name)

print("Units Used By the Costomer : ", self.\_\_unit\_used)

print("Net Amount you want to pay : ", self.\_net\_units)

run=ElectricityBill()

run.get\_input()

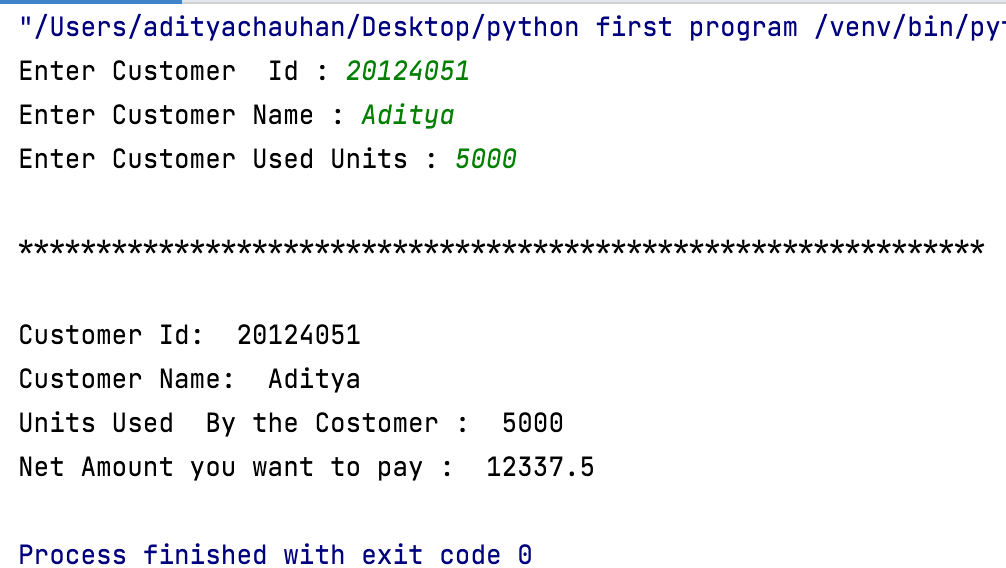
run.calculate()

run.condition()

run.net\_units()

run.display()

**Output :-**

****

**Program:-7**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write an OOP program to accept two numbers and one mathematical operator. Calculate and display appropriate answer.**

**Eg output**

**Enter first number : 45**

**Enter mathematical operator : +**

**Enter second number : 60**

**45 + 60 = 105**

**Input :-**

class Calculator:

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

self.\_\_num\_1 = int(input('Please enter number 1: '))

self.\_\_operator = input('Please enter mathematical operator: ')

self.\_\_num\_2 = int(input('Please enter number 2: '))

def calculate(self):

self.\_\_calculation = 0

self.\_\_opName = ''

if self.\_\_operator == '+':

self.\_\_calculation = self.\_\_num\_1 + self.\_\_num\_2

self.\_\_opName = 'Addition'

elif self.\_\_operator == '-':

self.\_\_calculation = self.\_\_num\_1 - self.\_\_num\_2

self.\_\_opName = 'Subtraction'

elif self.\_\_operator == '\*':

self.\_\_calculation = self.\_\_num\_1 \* self.\_\_num\_2

self.\_\_opName = 'Multiplication'

elif self.\_\_operator == '/':

self.\_\_calculation = self.\_\_num\_1 / self.\_\_num\_2

self.\_\_opName = 'Division'

def display(self):

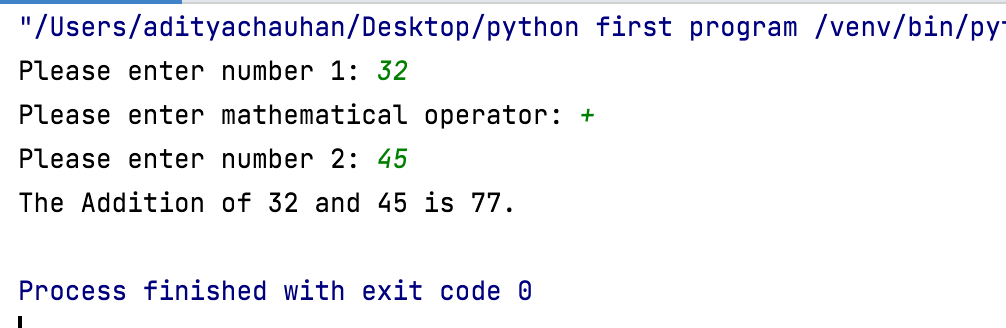
print('The {0} of {1} and {2} is {3}.'.format(self.\_\_opName,self.\_\_num\_1,self.\_\_num\_2,self.\_\_calculation))

run = Calculator()

run.calculate()

run.display()

**Output :-**

****

**Program:-8**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a program to check whether number is prime or not.**

**Enter a number : 13**

**13 is prime**

**Enter a number : 45**

**45 is not a prime number**

**Input :-**

class PrimeNumber:

def \_\_int\_\_(self):

self.Starting\_Number=0

self.Ending\_Number=0

self.num=0

def display\_Condition(self):

self.Starting\_Number=int(input("Enter Starting Number: "))

self.Ending\_Number=int(input("Enter Ending Number: "))

lower\_number = int(self.Starting\_Number)

upper\_number = int(self.Ending\_Number)

print("\nPrime Numbers between the given range:")

for self.num in range(lower\_number, upper\_number+1):

if self.num>1:

for i in range(2, self.num):

if(self.num%i)==0:

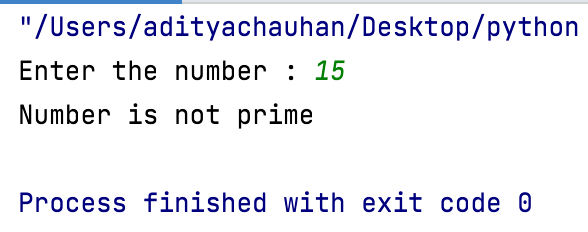
break

else:

print(self.num)

r=PrimeNumber()

r.display\_Condition()

**Output :-**

**Program:-9**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a program to display set of prime numbers between the given input range from user.**

**Enter start number : 10**

**Enter end number : 30**

**11,13,17,19,23,29**

**Input :-**

class PrimeNumber:

def \_\_int\_\_(self):

self.Starting\_Number=0

self.Ending\_Number=0

self.num=0

def display\_Condition(self):

self.Starting\_Number=int(input("Enter Starting Number: "))

self.Ending\_Number=int(input("Enter Ending Number: "))

lower\_number = int(self.Starting\_Number)

upper\_number = int(self.Ending\_Number)

print("\nPrime Numbers between the given range:")

for self.num in range(lower\_number, upper\_number+1):

if self.num>1:

for i in range(2, self.num):

if(self.num%i)==0:

break

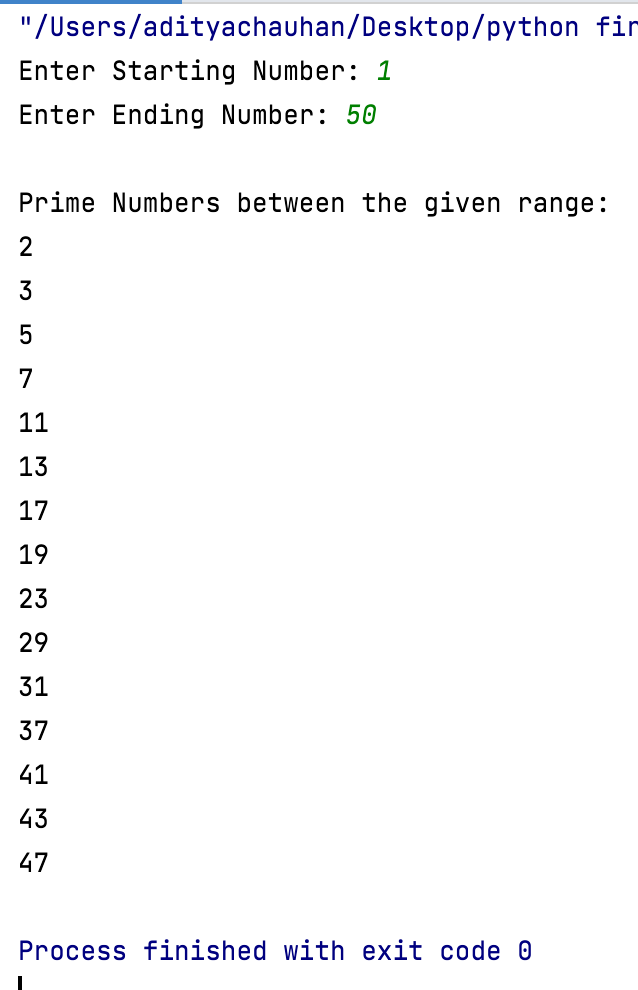
else:

print(self.num)

r=PrimeNumber()

r.display\_Condition()

**Output :-**

****

**Program:-10**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**A program to check whether inputted string is palindrome or not.**

**Eg Enter a name : liril**

**Liril is a palindrom**

**Input :-**

class Palindrome:

def \_\_int\_\_(self):

self.String = ''

def display\_Condition(self):

self.String = input("Enter a string:")

if (self.String == self.String[::-1]):

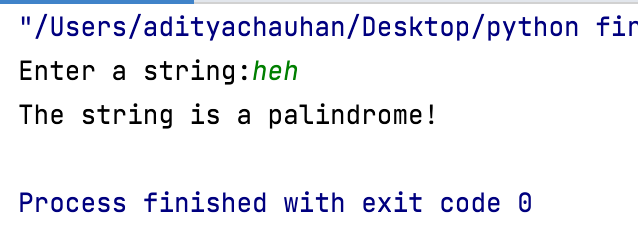
print("The string is a palindrome!")

else:

print("The string isn't a palindrome!")

r = Palindrome()

r.display\_Condition()

**Output :-**

**Program:-11**

**@author: 20124051 Aditya Chauhan**

**@description: Program No. -**

**Write a OO program to find Euclidean Distance.**

**Input :-**

class Point:

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

self.\_\_x1 = 0

self.\_\_y1 = 0

self.\_\_x2 = 0

self.\_\_y2 = 0

self.\_\_delta\_x = 0

self.\_\_delta\_y = 0

self.\_\_euDist = 0

def get\_input(self):

self.\_\_x1 = int(input('Please enter integer value for x1: '))

self.\_\_y1 = int(input('Please enter integer value for y1: '))

self.\_\_x2 = int(input('Please enter integer value for x2: '))

self.\_\_y2 = int(input('Please enter integer value for y2: '))

def dist\_to\_point(self):

self.\_\_delta\_x = self.\_\_x2 - self.\_\_x1

self.\_\_delta\_y = self.\_\_y2 - self.\_\_y1

self.\_\_euDist = (self.\_\_delta\_x \*\* 2 + self.\_\_delta\_y \*\* 2) \*\* 0.5

return self.\_\_euDist

def display(self):

print("The Euclidean distance is: ",self.\_\_euDist)

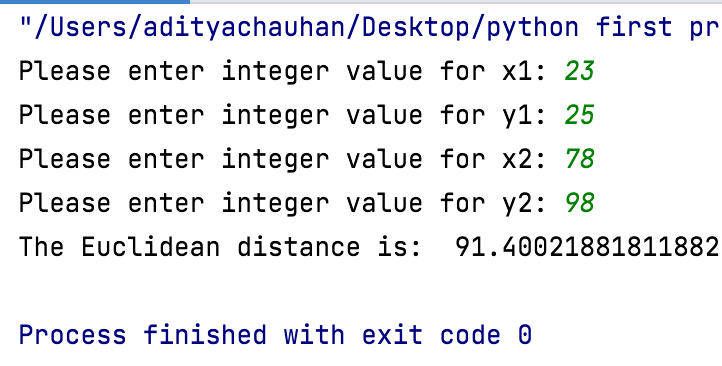
run = Point()

run.get\_input()

run.dist\_to\_point()

run.display()

**Output :-**

**