**Experiment No. 8-A**

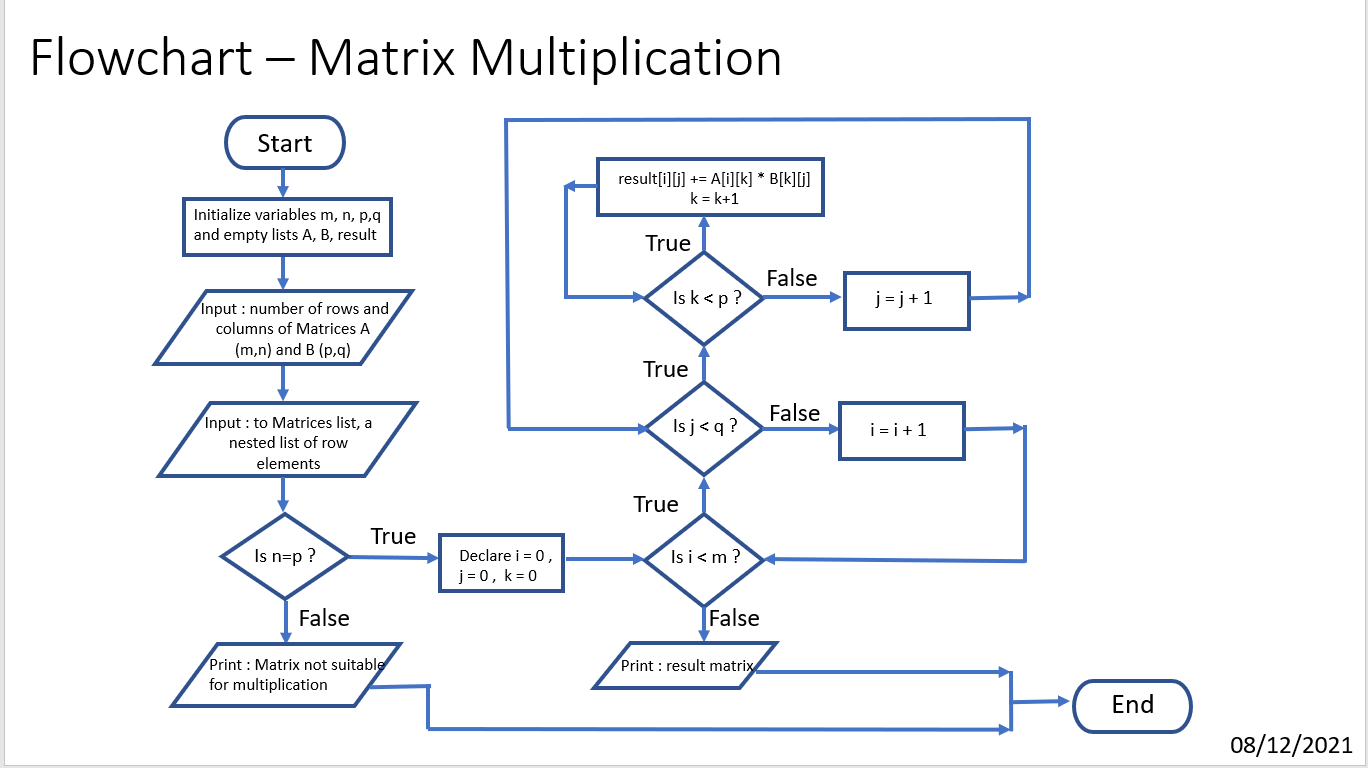
**Title:**

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

**Tool/Platform:**

Microsoft Word / PowerPoint and Python IDLE

**Flowchart:**



**Algorithm:**

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

**Source Code:**

#Matrix Multiplication

#Date: 08/12/2021

import sys

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

print()

print("For Matrix 1:")

m=int(input("Enter number of rows: "))

n=int(input("Enter number of column: "))

A = [ ]

print("Entries to be entered row - wise")

for i in range(m):

a=[ ]

for j in range(n):

a.append(int(input("Enter Element: ")))

A.append(a)

print()

print("For Matrix 2:")

p=int(input("Enter number of rows: "))

q=int(input("Enter number of columns: "))

B = [ ]

print("Entries to be entered row - wise")

for i in range(p):

b=[ ]

for j in range(q):

b.append(int(input("Enter Element: ")))

B.append(b)

print()

print("Printing Entered Matrices:")

for r in A:

print(r)

print()

for r in B:

print(r)

print()

if n != p:

print("As per fundamentals of Matrix Multiplication, number of columns of first matrix should be equal to number of rows of second matrix ... ")

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

rows,column=m,q

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

print("Performing Matrix Multiplication !")

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

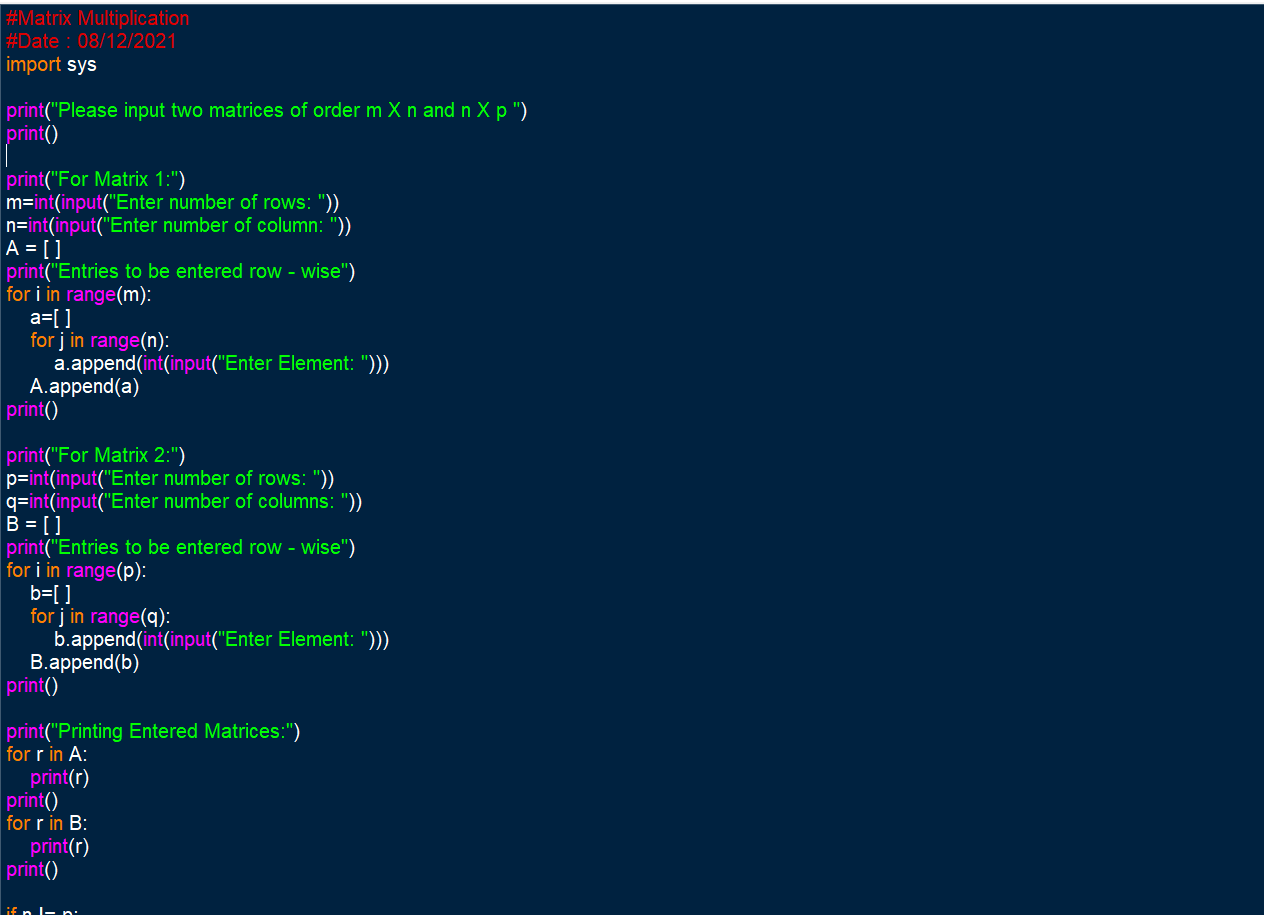
for j in range(q): #columns of B

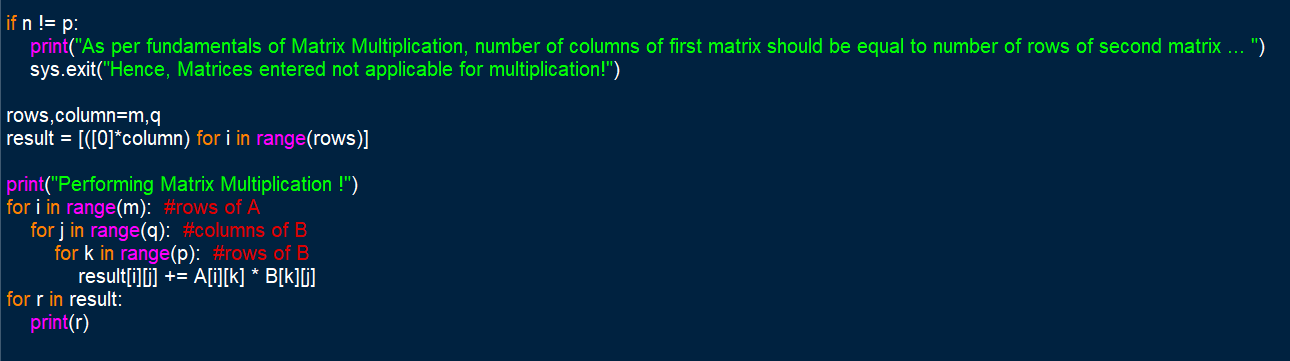
for k in range(p): #rows of B

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

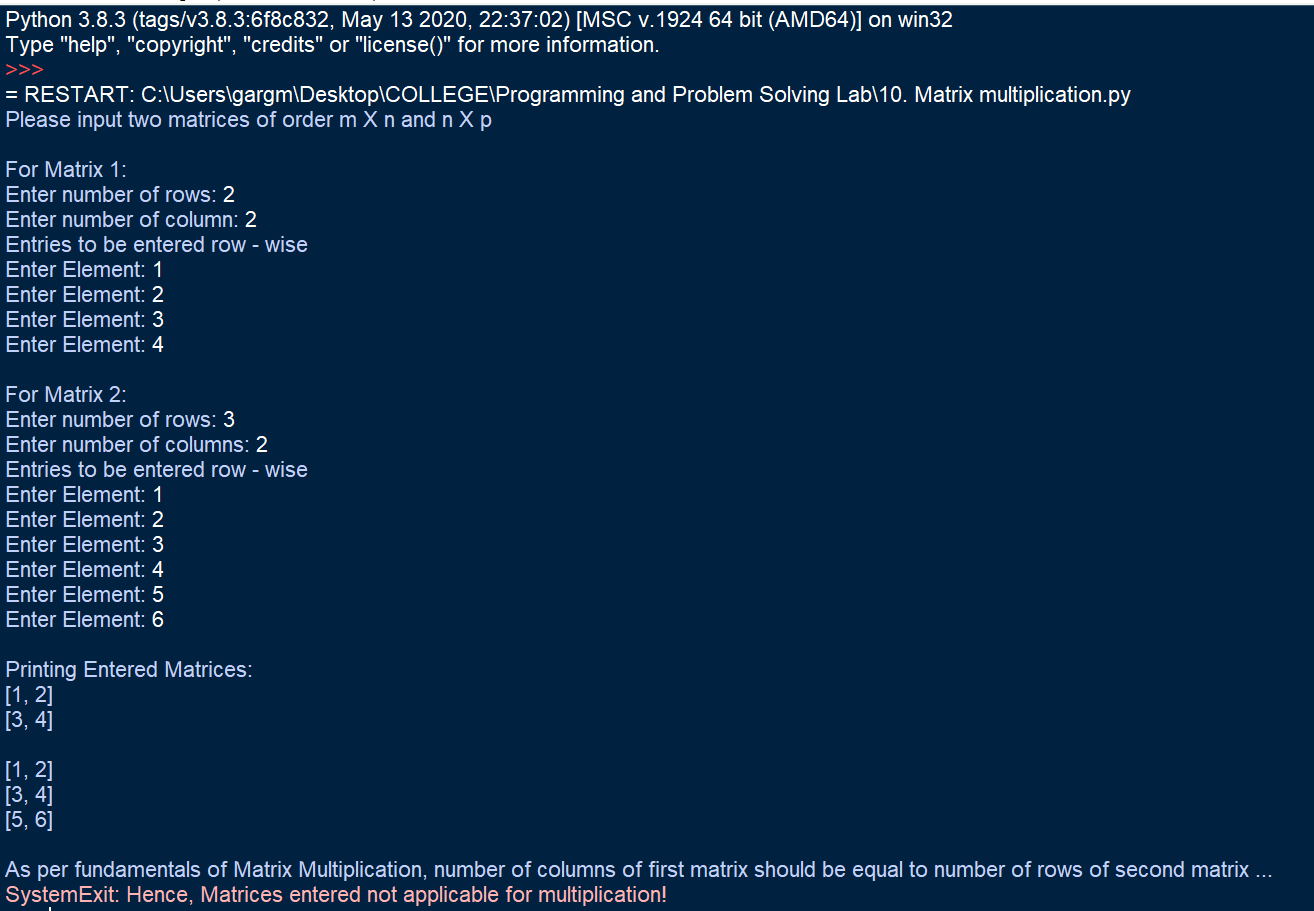
for r in result:

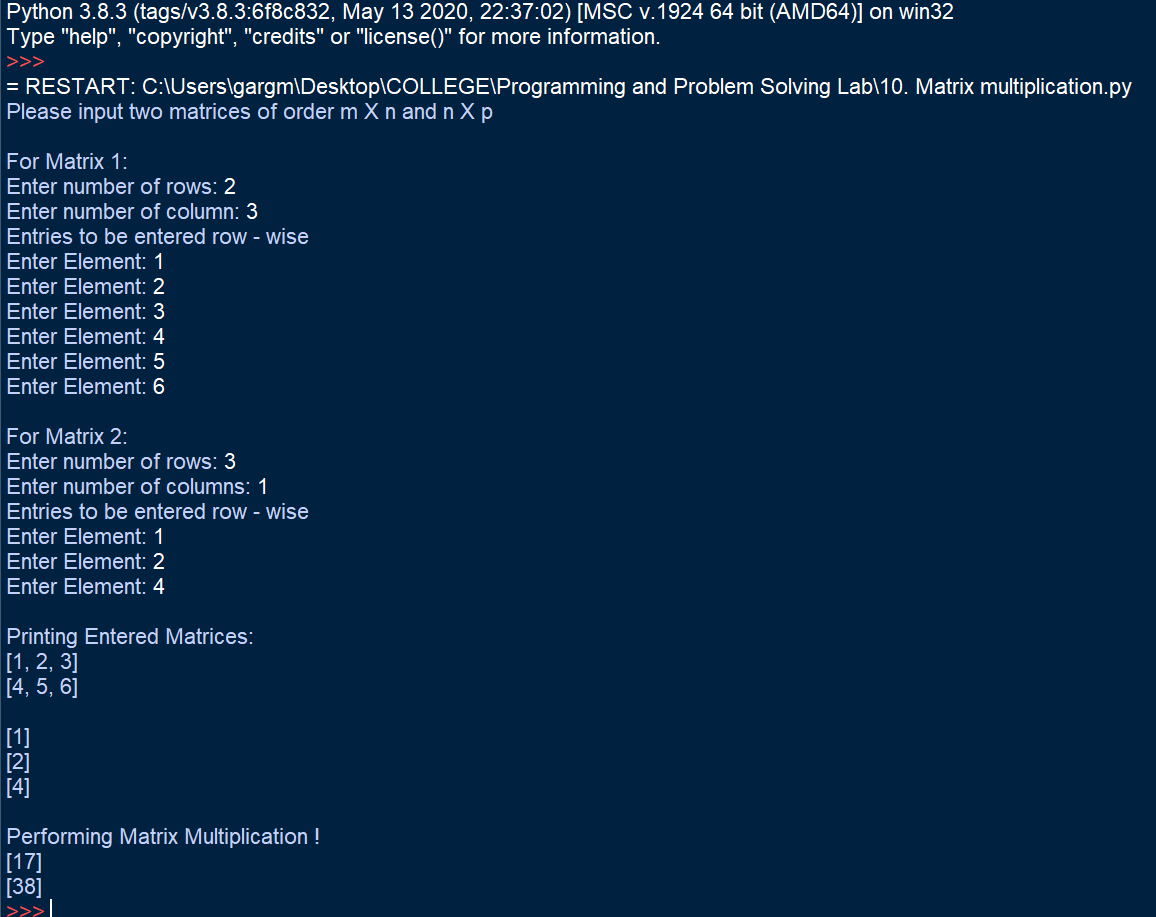
print(r)

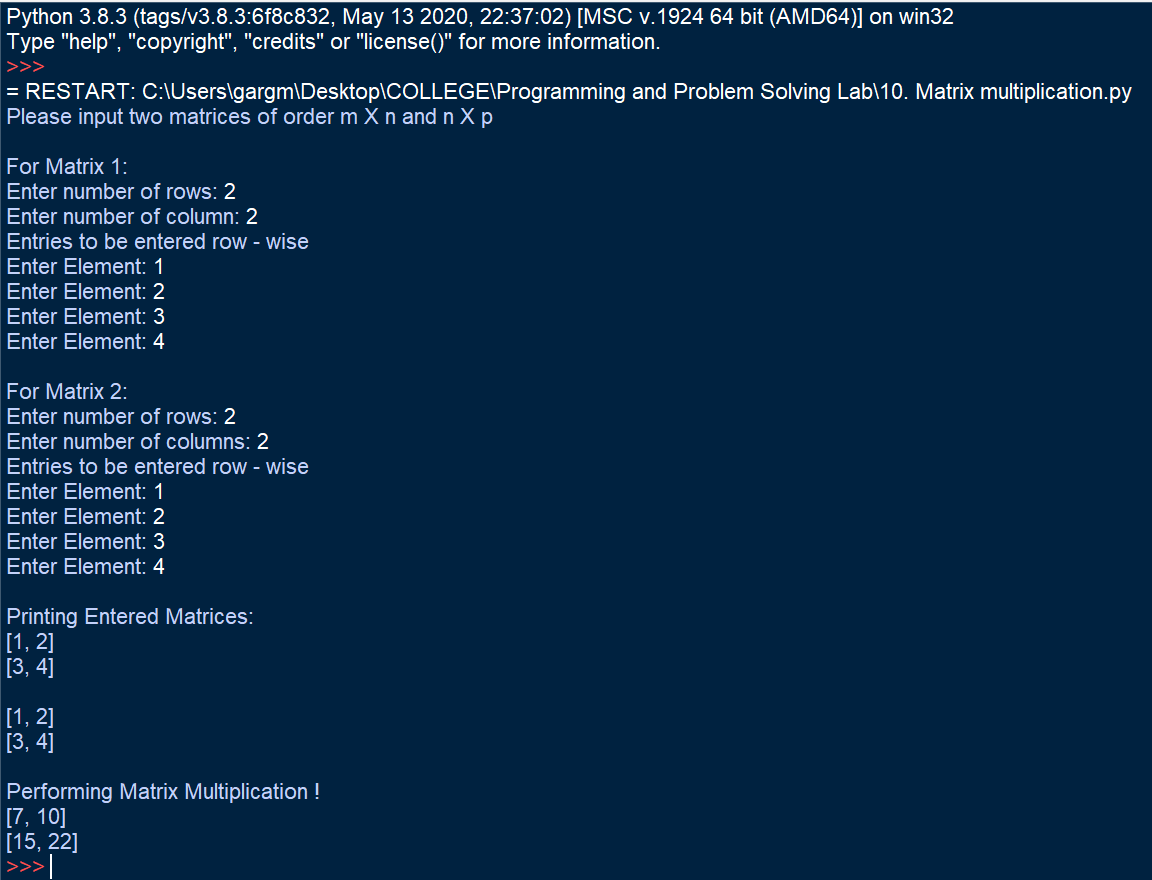




**Output Screenshots:**







**Learning Outcome:**

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

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

**Experiment No. 8-B**

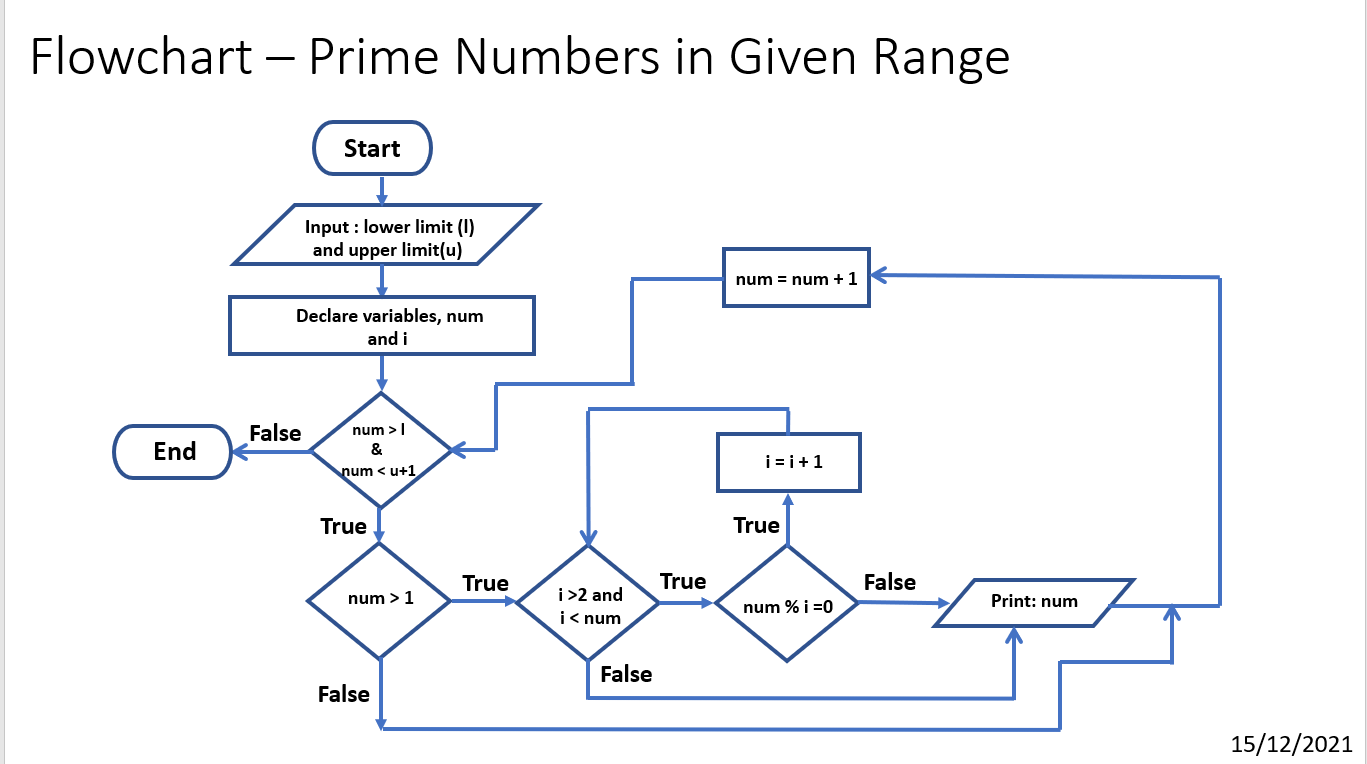
**Title:**

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

**Tool/Platform:**

Microsoft Word / PowerPoint and Python IDLE

**Flowchart:**



**Algorithm:**

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

**Source Code:**

#Prime Numbers in given range

# Date: 15/12/2021

l = int(input("Enter Lower Limit of the Range: "))

u = int(input("Enter Upper Limit of the Range: "))

print()

c=0

print(f'Printing Prime Numbers Between Entered Range {l} to {u}')

for num in range(l, u+1):

if num>1:

for i in range(2, num):

if (num%i==0):

break

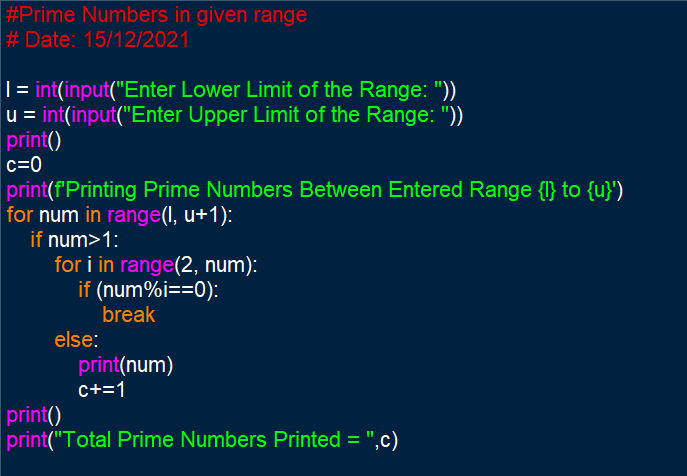
else:

print(num)

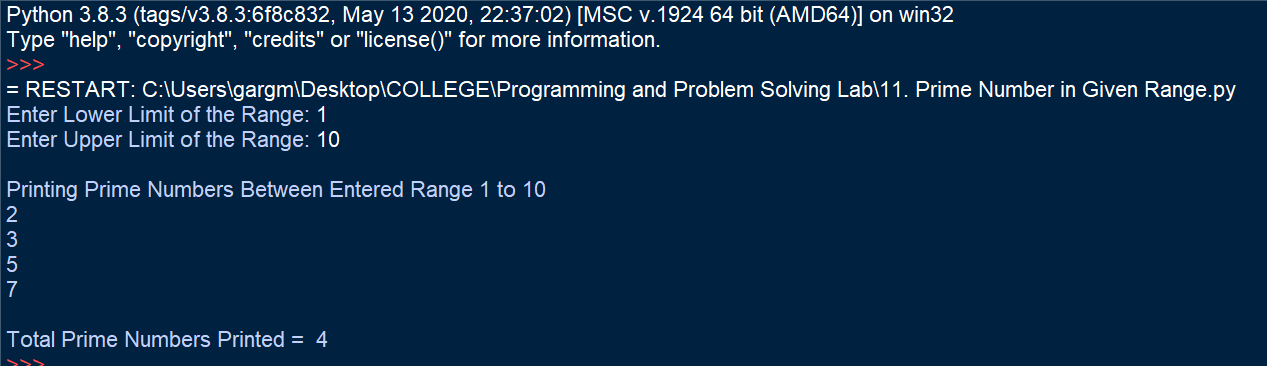
c+=1

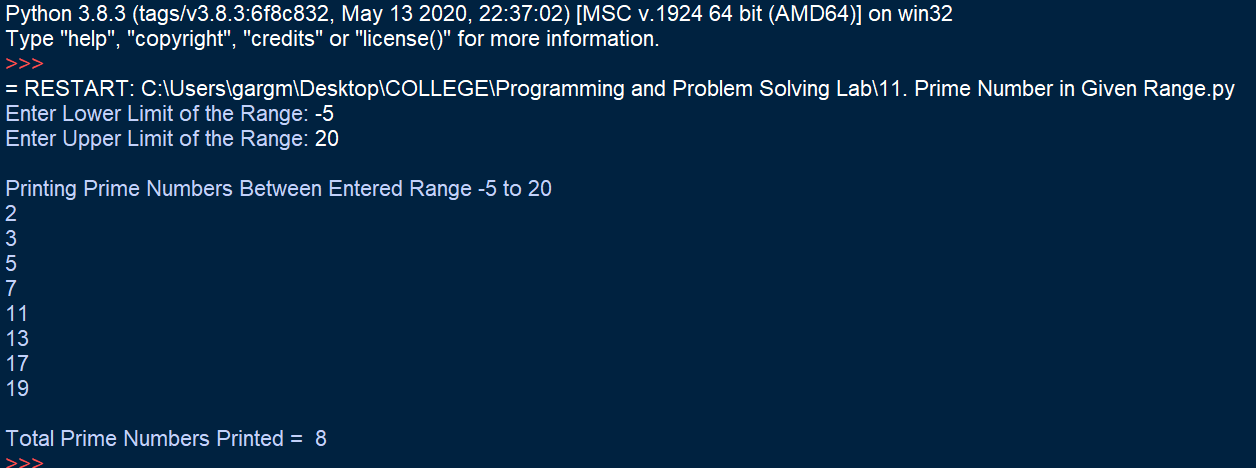
print()

print("Total Prime Numbers Printed = ",c)



**Output Screenshots:**





**Learning Outcome:**

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

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