**Python Basics**

**1 Installation, Expression, Data types, Variables, Execution**

**2 Operators and Decision Controls (IF-else)**

**3 Loops for, while, Range( Examples)**

**4 Functions or methods**

**5 Collections( List, tuple, Dictionary)**

Python Modules: One Month

String and Files

Debugging and Functions

Collectins

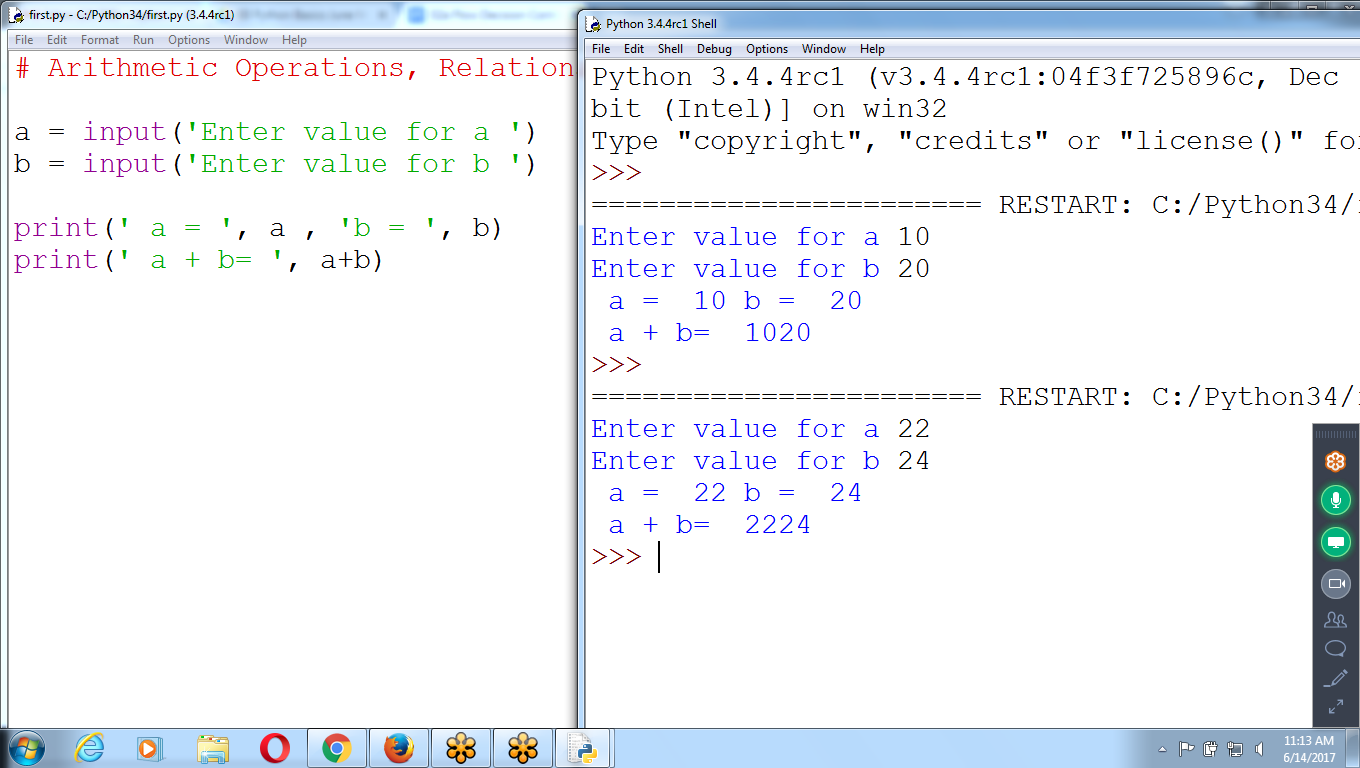
OOPS

Python with Aws/Devops / Data science

Python with Big data, Data Science

Python can use Testing(selenium), Networking, Gaming, GUI

# a and b are string type :: String COncatenation



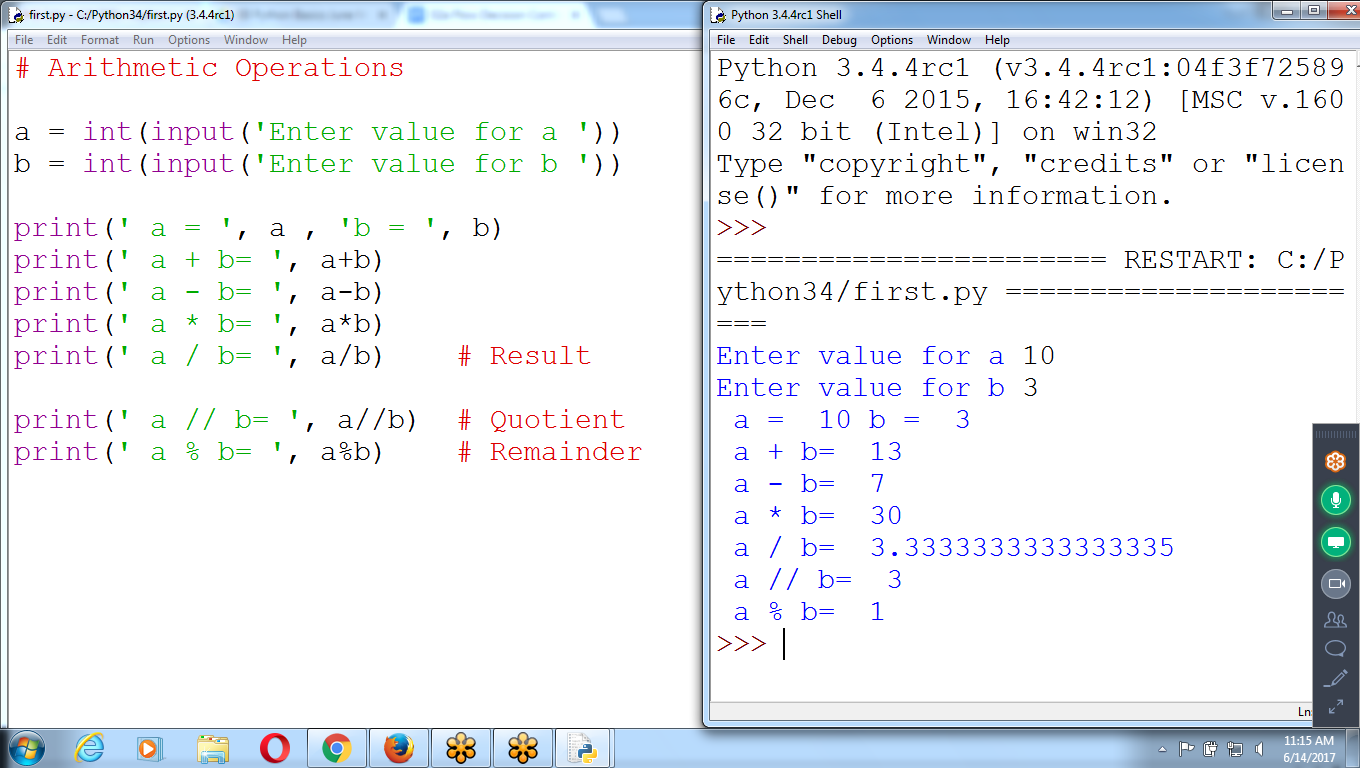
a = input('Enter value for a ')

b = input('Enter value for b ')

print(' a = ', a , 'b = ', b)

print(' a + b= ', a+b)

Arithmetic Operations



# Arithmetic Operations

a = int(input('Enter value for a '))

b = int(input('Enter value for b '))

print(' a = ', a , 'b = ', b)

print(' a + b= ', a+b)

print(' a - b= ', a-b)

print(' a \* b= ', a\*b)

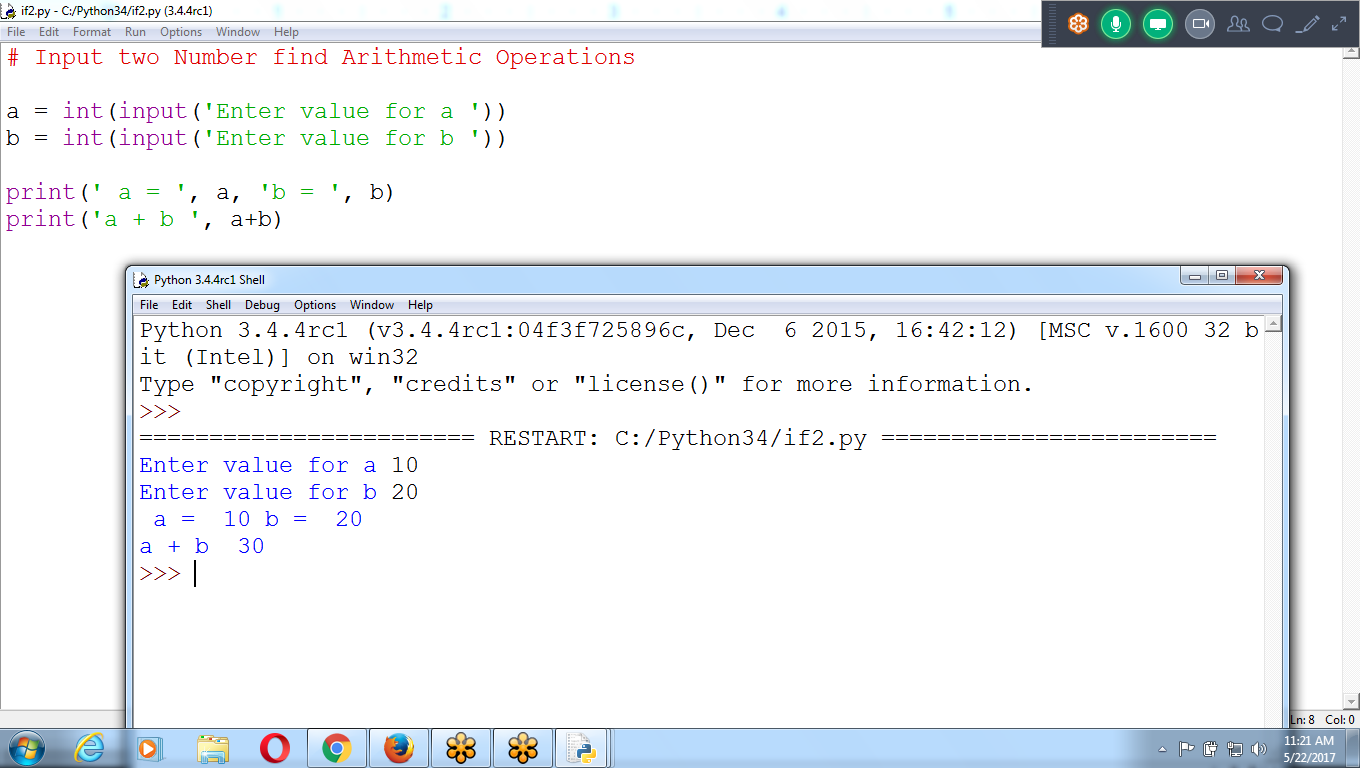
print(' a / b= ', a/b) # Result

print(' a // b= ', a//b) # Quotient

print(' a % b= ', a%b) # Remainder

A and b are INteger type

int() converts string to int



# **Boolean Values**

* integer, floating-point, and string data types have an **unlimited** number of possible values,
* *Boolean* data type has only two values: **True and False**

# **Comparison or Relational Operators**

*Comparison operators* compare two values and evaluate down to a **single Boolean value**

|  |  |
| --- | --- |
| **Operator** | **Meaning** |
| **==** | **Equal to (comparison)** |
| **!=** | **Not equal to** |
| **<** | **Less than** |
| **>** | **Greater than** |
| **<=** | **Less than or equal to** |
| **>=** | **Greater than or equal to** |

* **The == operator (Equal to) asks whether two values are the same as each other.**
* **The = operator (Assignment) puts the value on the right into the variable on the left.**

# **Boolean Operators**

The three Boolean operators (and, or, and not) are used to compare Boolean values.

The and and or operators always take two Boolean values (or expressions), so they’re considered *binary* operators.

The **and** operator evaluates an expression to True if *both* Boolean values are True; otherwise, it evaluates to False

|  |  |
| --- | --- |
| **Expression** | **Evaluates to...** |
| **True and True** | **True** |
| True and False | False |
| False and True | False |
| False and False | False |

The **or** operator evaluates an expression to True if *either* of the two Boolean values is True. If both are False, it evaluates to False.

|  |  |
| --- | --- |
| **Expression** | **Evaluates to...** |
| True or True | True |
| True or False | True |
| False or True | True |
| **False or False** | **False** |

## 

## 

## **The not Operator**

**The not operator simply evaluates to the opposite Boolean value**

|  |  |
| --- | --- |
| **Expression** | **Evaluates to...** |
| not True | False |
| not False | True |

# **Mixing Boolean and Comparison Operators**

the and, or, and not operators are called Boolean operators because they always operate on the Boolean values True and False

While expressions like 4 < 5 aren’t Boolean values, they are expressions that evaluate down to Boolean values.

>>> (4<5) and (5<6)

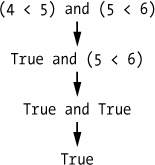
True

>>> (7<5) or (3 > -20)

>>> 2+2 == 4 and not 2+2 == 5 and 2\*2 == 2+2

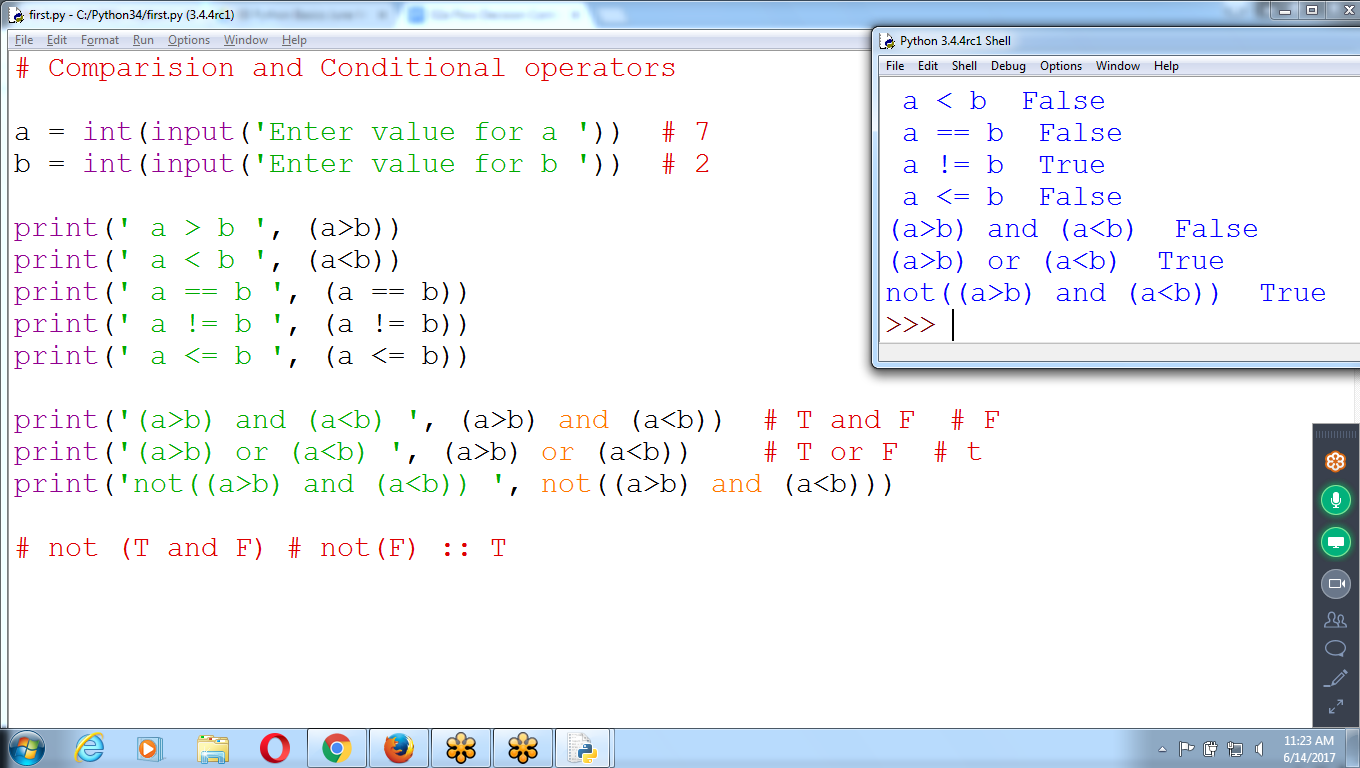
True

Python evaluates the **not operators first**, then the and operators, and then the or operators.



**# Input Two Integer Values and Print ARithmetic Operations,**

**Relational and Logical Operations Results**



# Comparison and Conditional operators

a = int(input('Enter value for a ')) # 7

b = int(input('Enter value for b ')) # 2

print(' a > b ', (a>b))

print(' a < b ', (a<b))

print(' a == b ', (a == b))

print(' a != b ', (a != b))

print(' a <= b ', (a <= b))

print('(a>b) and (a<b) ', (a>b) and (a<b)) # T and F # F

print('(a>b) or (a<b) ', (a>b) or (a<b)) # T or F # t

print('not((a>b) and (a<b)) ', not((a>b) and (a<b)))

# not (T and F) # not(F) :: T

# **Flow Control or Decision Controls**

* A **program** is just a **series of instructions**
* The program can decide **to skip (decision: if-else)** instructions, **repeat** (loops:for,while) them, or

**choose (Case)**which instructions to run.

* ***Flow control statements***  **decide** which Python instructions to execute under which conditions and correspond to the **symbols** in a flowchart

Flowcharts represent these branching points with diamonds, while the other steps are represented with rectangles. The starting and ending steps are represented with rounded rectangles.

# **Elements of Flow Control**

Flow control statements starts with a part called the *condition*, and all are followed by a block of code called the *clause*.

## **Conditions**

* Conditions always evaluate down to a Boolean value, True or False.
* A flow control statement decides what to do based on whether its condition is True or False

## **Blocks of Code**

Lines of Python code can be grouped together in *blocks*.

Indentation of the lines of code represents when a block begins and ends

1. Blocks begin when the indentation increases.
2. Blocks can contain other blocks.
3. Blocks end when the indentation decreases to zero or to a containing block’s indentation.

# **Flow Control Statements**

if statement’s clause (that is, the block following the if statement) will execute if the statement’s condition is True. The clause is skipped if the condition is False.

* The if keyword
* A condition (that is, an expression that evaluates to True or False)
* **A colon**
* Starting on the next line, an indented block of code (called the if clause)

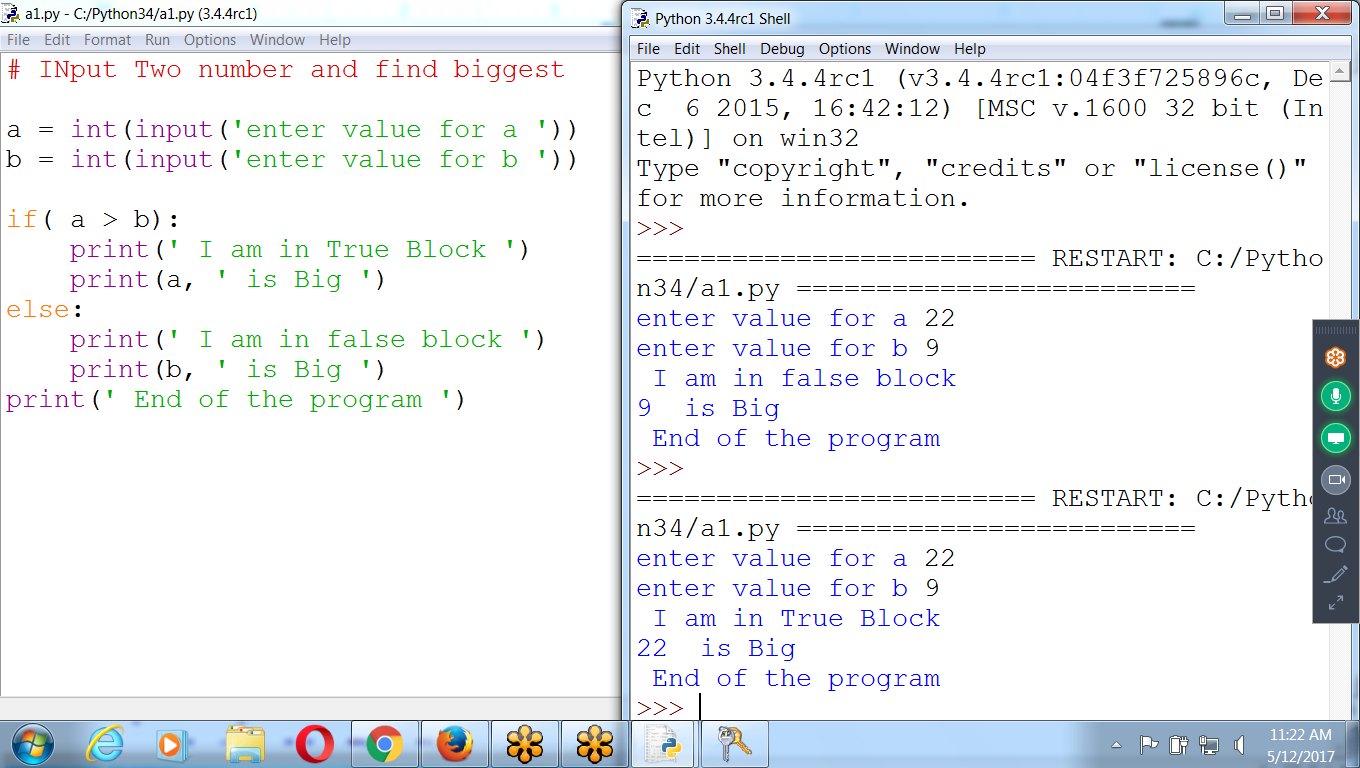
**FLOW Control Examples**

**Find Biggest among two values using if-else**

****

**ERROR :: INput values are string format**

**int() converts string to integer**

****

**# Input two numbers and Find Biggest number**

**a = int(input('Enter value for a '))**

**b = int(input('Enter value for b '))**

**if( a > b):**

**print(a, ' Is Big ')**

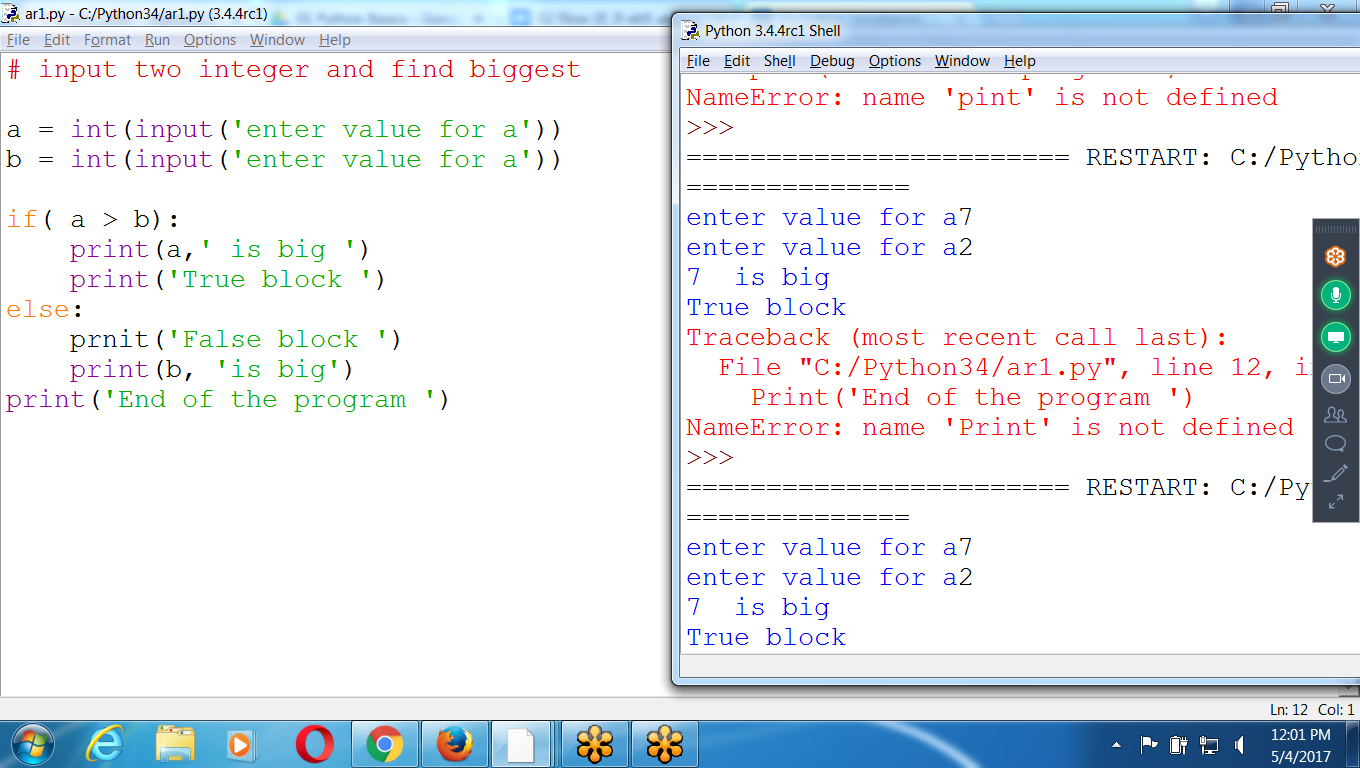
**print(' I am in true block ')**

**else:**

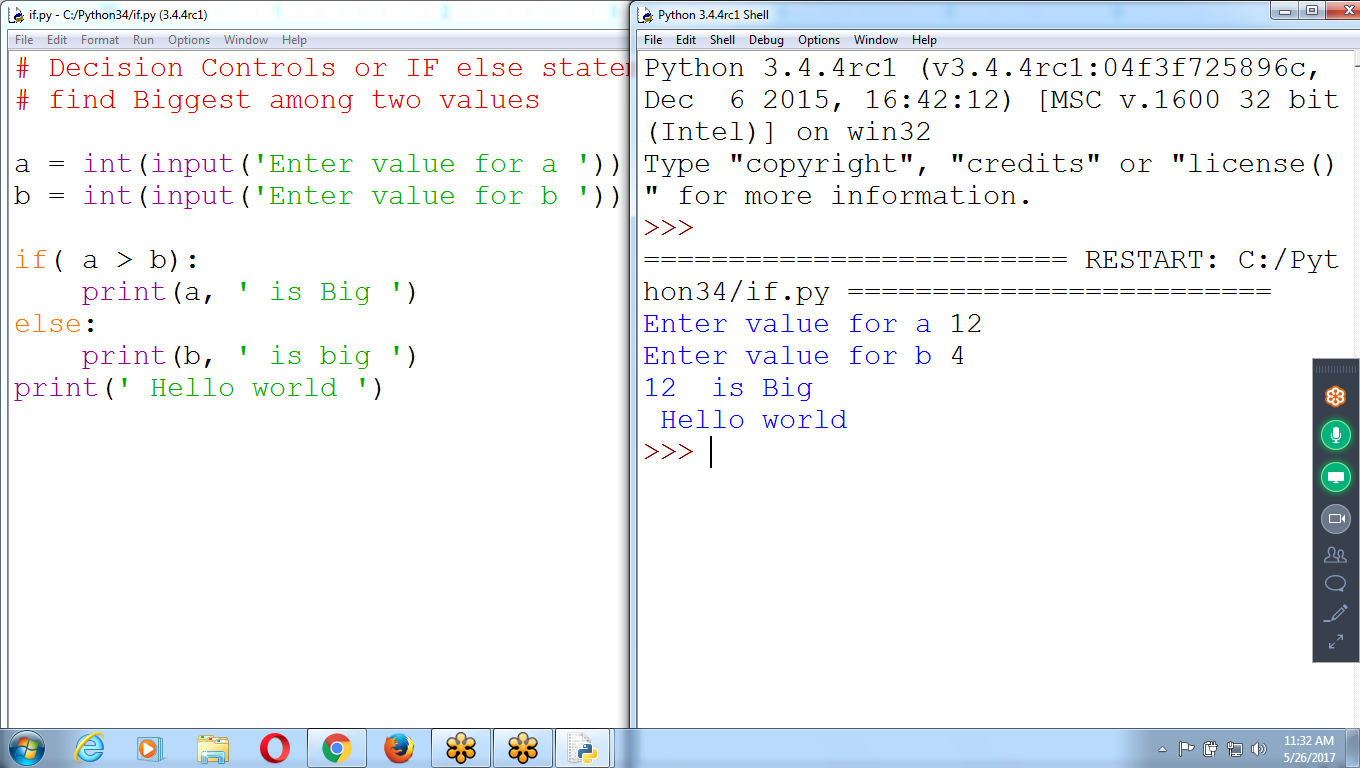
**print(b, ' Is Big')**

**print(' I am in false Block ')**

**print(' Bye…. ')**

****

**To Find Biggest number using if -else**

****

**Find Biggest using simple IF**

**Prog2 : Accept Name, age and**

**print Message using if-elif-else**

**# INput name and age and print message**

**name = input('Enter name ')**

**age = int(input('Enter age '))**

**if( age <=18):**

**print(name, ' Is Student ')**

**elif (age <=24):**

**print(name, ' PG STudent ')**

**elif (age <= 30):**

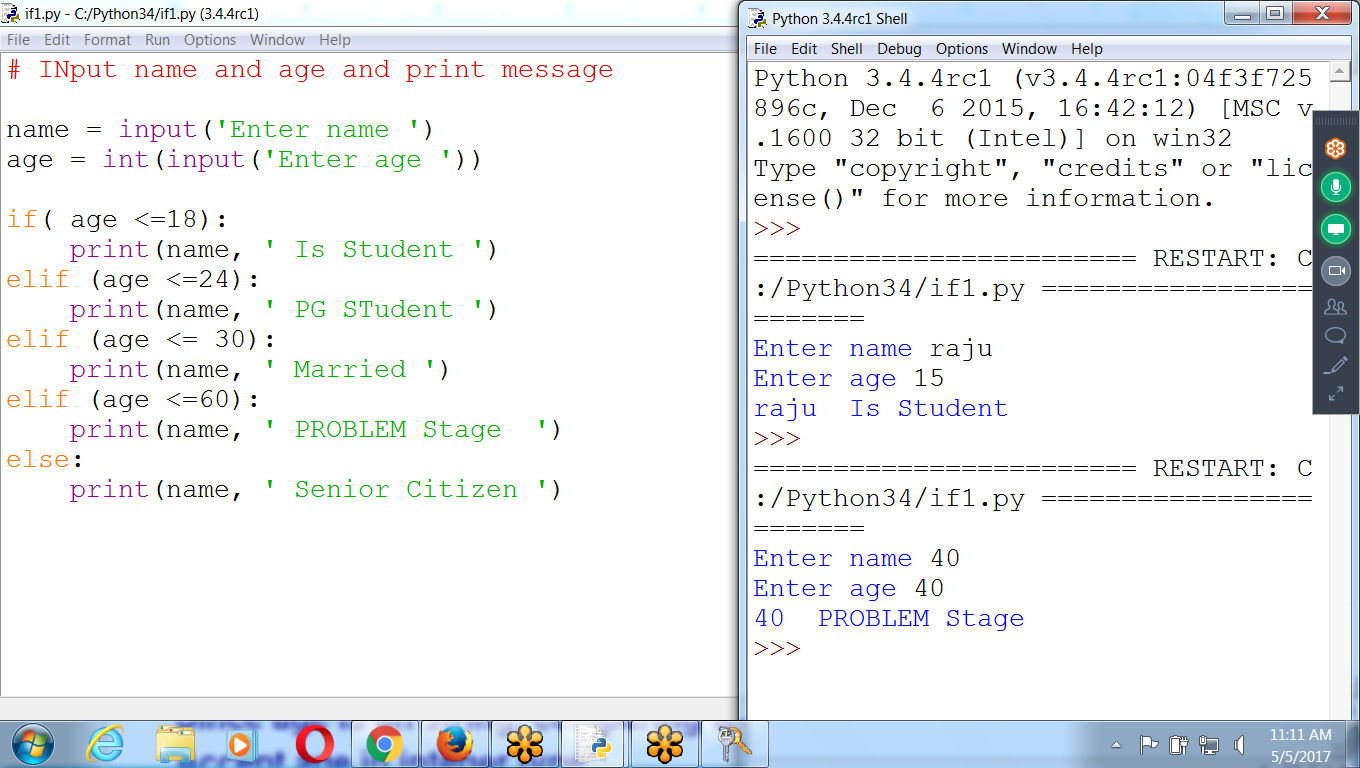
**print(name, ' Married ')**

**elif (age <=60):**

**print(name, ' PROBLEM Stage ')**

**else:**

**print(name, ' Senior Citizen ')**

****

**>>> 7/3 # Result**

**2.3333333333333335**

**>>> 7//3 # Quotient**

**2**

**>>> 7%3 # Remainder**

**1**

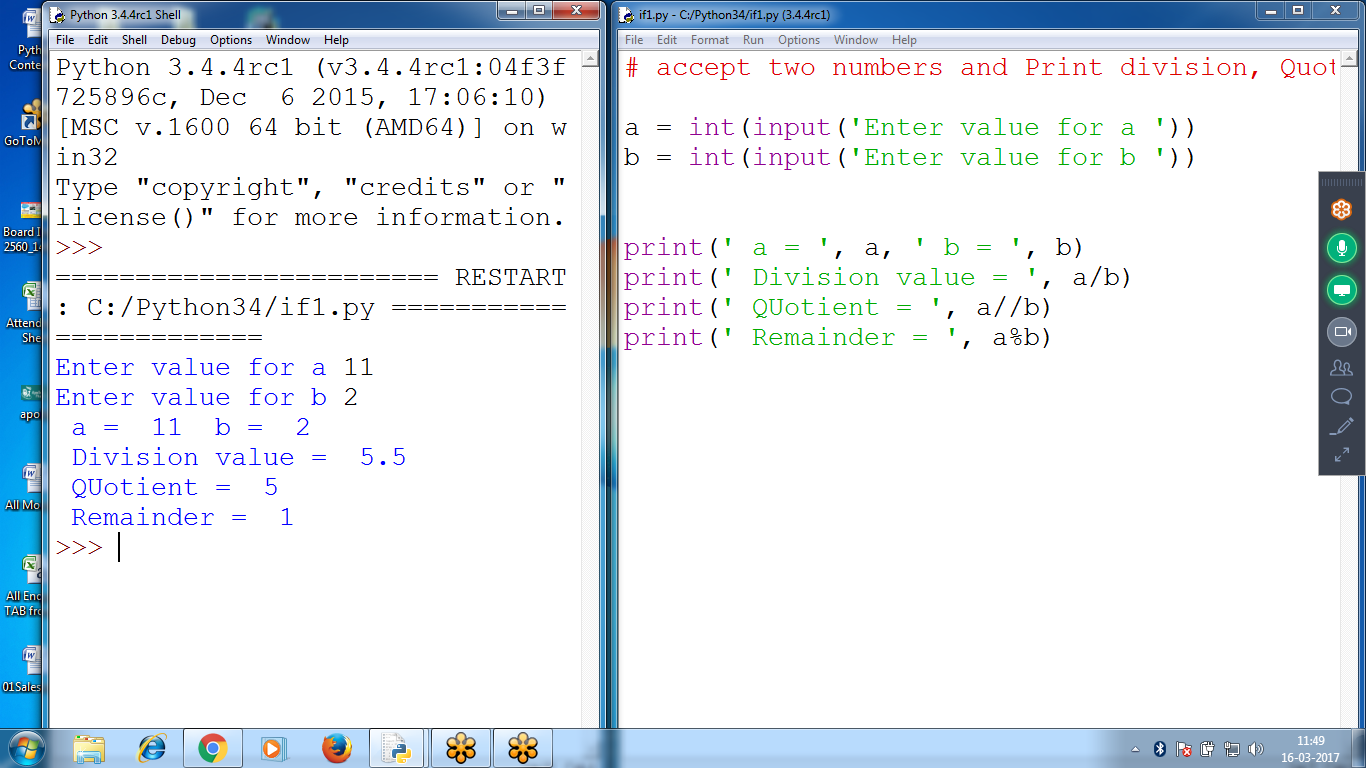
**>>>**

**Print given number is EVEN or ODD**

**Even : Divisible by 2 :: 2,4,6,8 ::: Remainder zero**

**Odd : 1,3,5,7,9,.... Remainder one**

**Get Division Result, QUotient, Remainder**

****

**# accept two numbers and Print division, Quotient and Remainder**

**a = int(input('Enter value for a '))**

**b = int(input('Enter value for b '))**

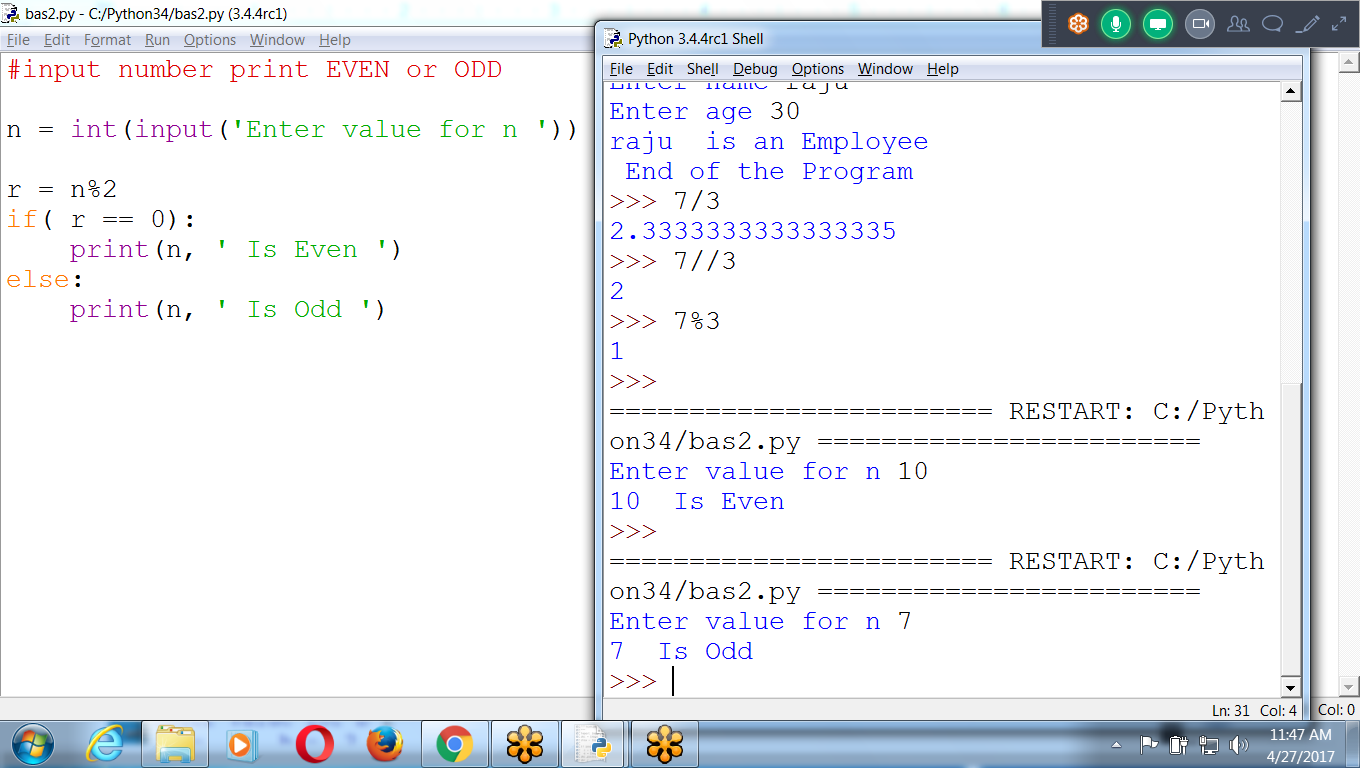
**print(' a = ', a, ' b = ', b)**

**print(' Division value = ', a/b)**

**print(' QUotient = ', a//b)**

**print(' Remainder = ', a%b)**

**Print given number is EVEN or ODD**

****

**# input value for n and print even or ODD**

**# even remainder is zero**

**n = int(input('Enter value for n '))**

**r = n%2 # Remainder value**

**if( r == 0):**

**print(n, ' is EVEN ')**

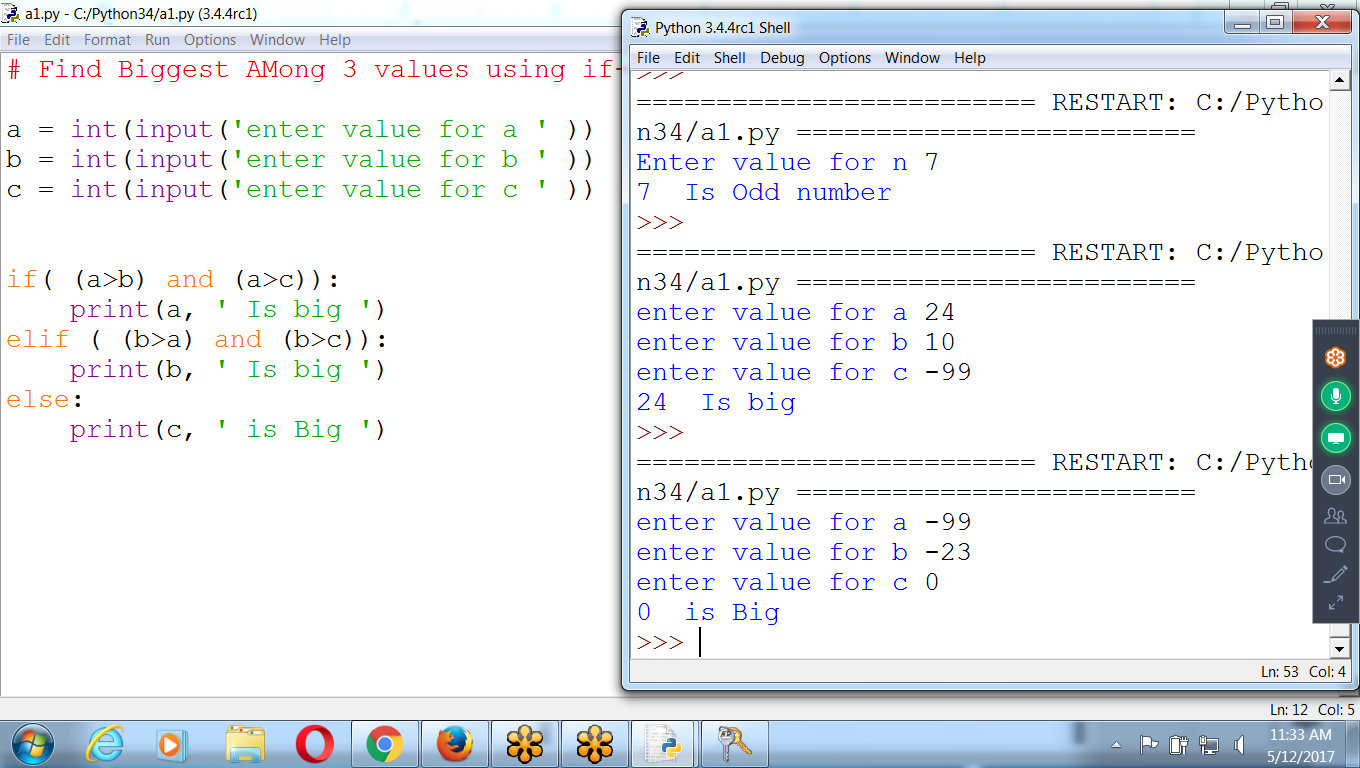
**else:**

**print(n, ' is Odd ')**

**print(' Remainder = ', r)**

**print(' End of the program ')**

**Final :: Find Biggest among Three values using if-elif**

****

**# Find Biggest among 3 values using if-elif**

**a = int(input('Enter value for a '))**

**b = int(input('Enter value for b '))**

**c = int(input('Enter value for c '))**

**if( (a>b) and (a>c)):**

**print(a, ' Is Big ')**

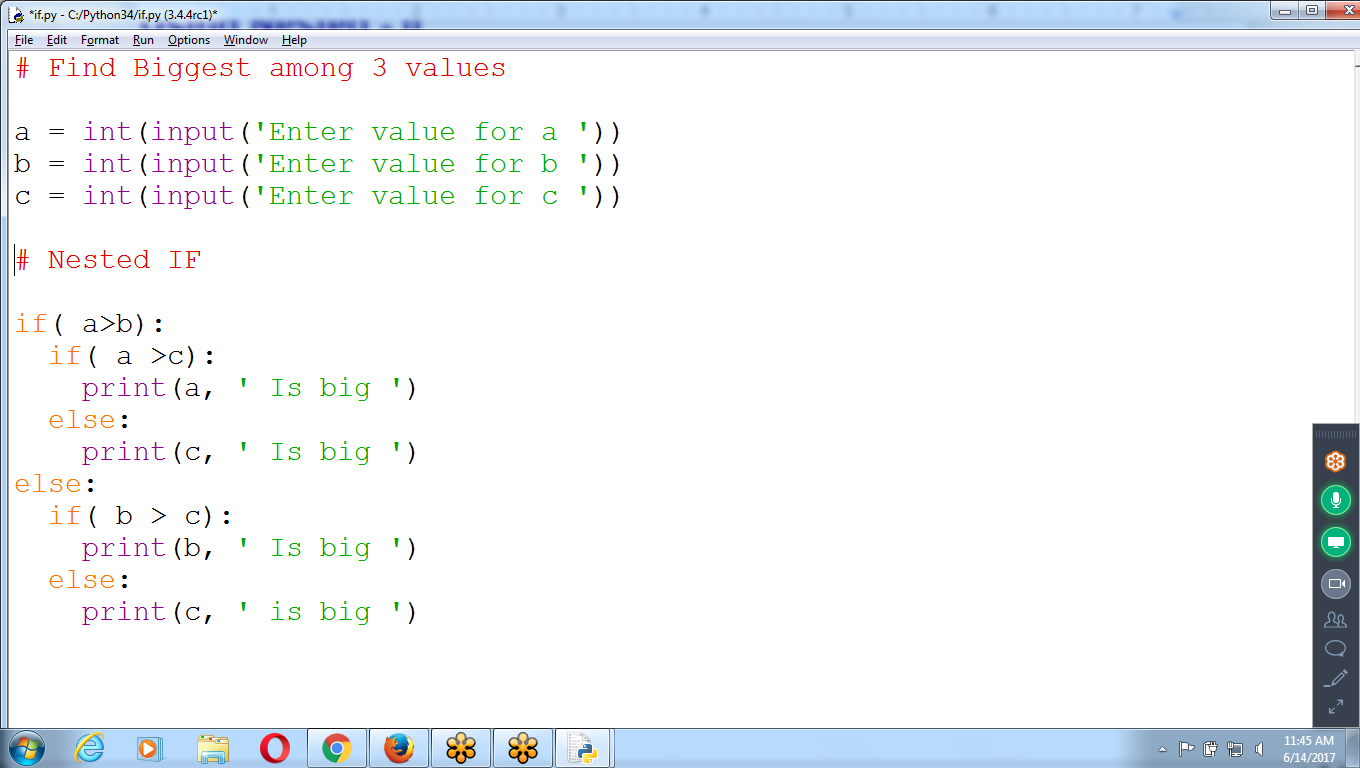
**elif (b>a) and (b>c):**

**print(b, ' Is Big ')**

**else:**

**print(c, ' Is Big ')**

**Using Nested - If**

****

# Find Biggest amoung 3 values using if-elif, nested if

a = int(input('Enter value for a '))

b = int(input('Enter value for a '))

c = int(input('Enter value for a '))

if( (a>b) and (a>c)):

**print(a, ' Is big ')**

elif( (b>a) and (b>c)):

print(b, ' Is big ')

else:

print(c, ' Is big ')

Assigning to a Variable

if( a > b):

if( a > c):

**big = a**

else:

big = c

else:

if(b > c):

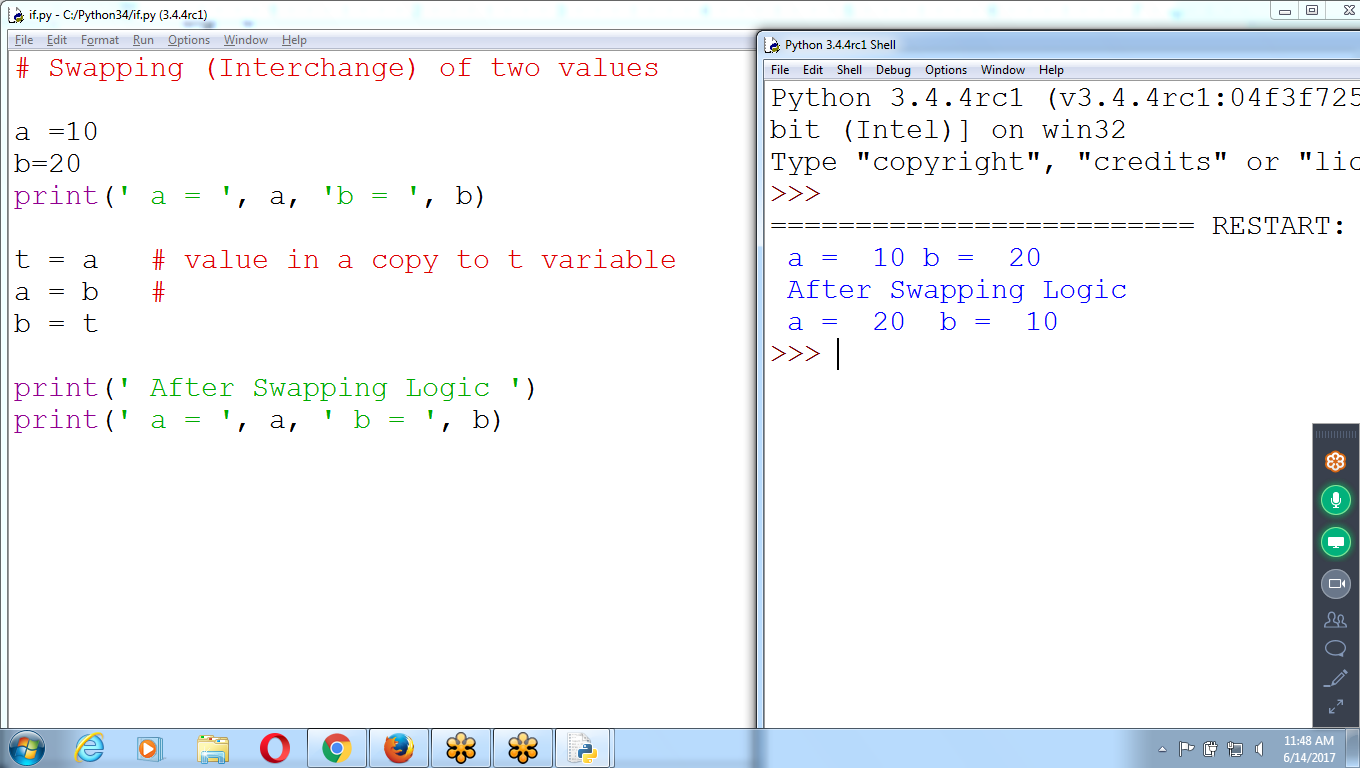
big = b

else:

big = c

print(' Biggest element ', big)

**Swapping (Interchanging) of two values**



# Swapping (Interchange) of two values

a =10

b=20

print(' a = ', a, 'b = ', b)

**t = a # value in ‘a’ copies to ‘t’ variable**

**a = b**

**b = t**

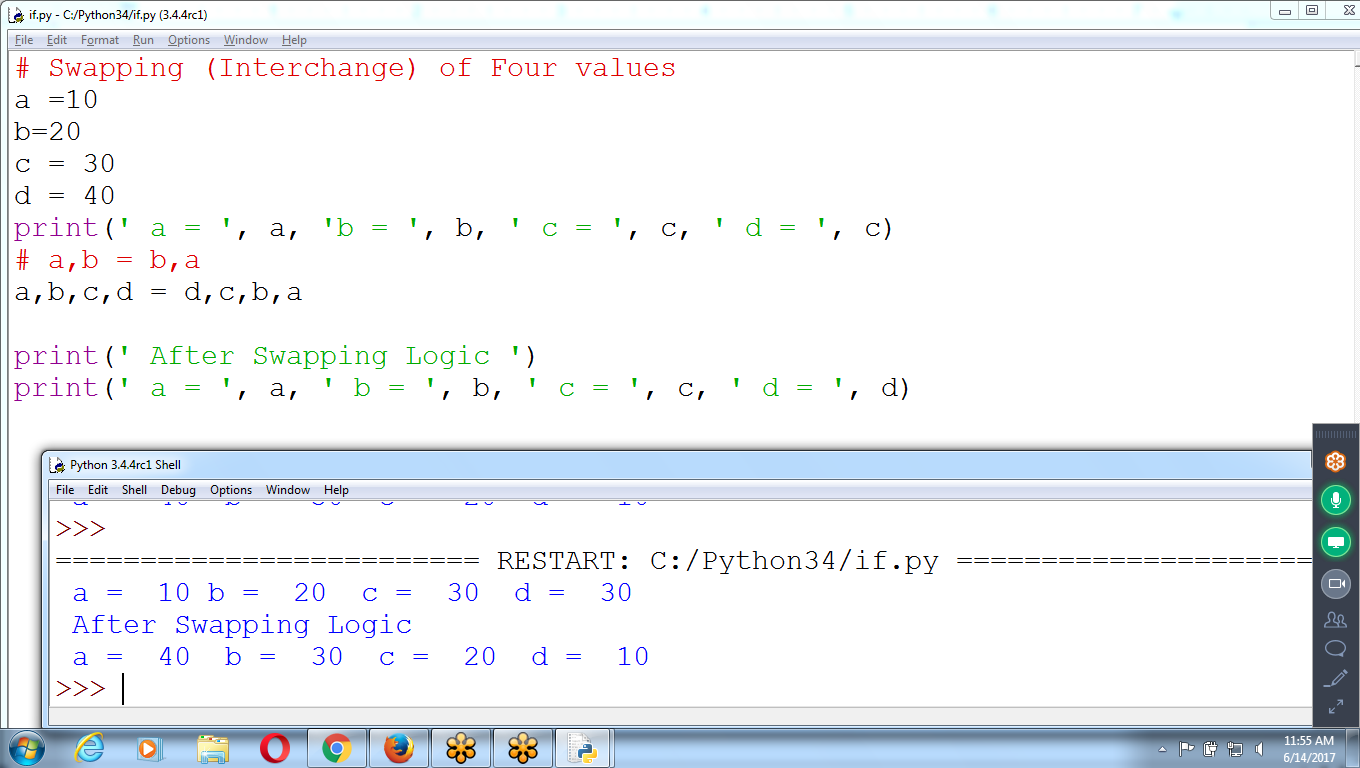
print(' After Swapping Logic ')

print(' a = ', a, ' b = ', b)

**MULTI Assignment Variable**

**INterchange 4 variables, Without Temporary Variable**

**a,b,c,d = d,c,b,a**

****

**# input Two Values and Swap (Interchange) Them**

**a = int(input('Enter value for a '))**

**b = int(input('Enter value for b '))**

**print(' Before Swapping ')**

**print(' a = ', a, ' b = ', b)**

**a,b = b,a**

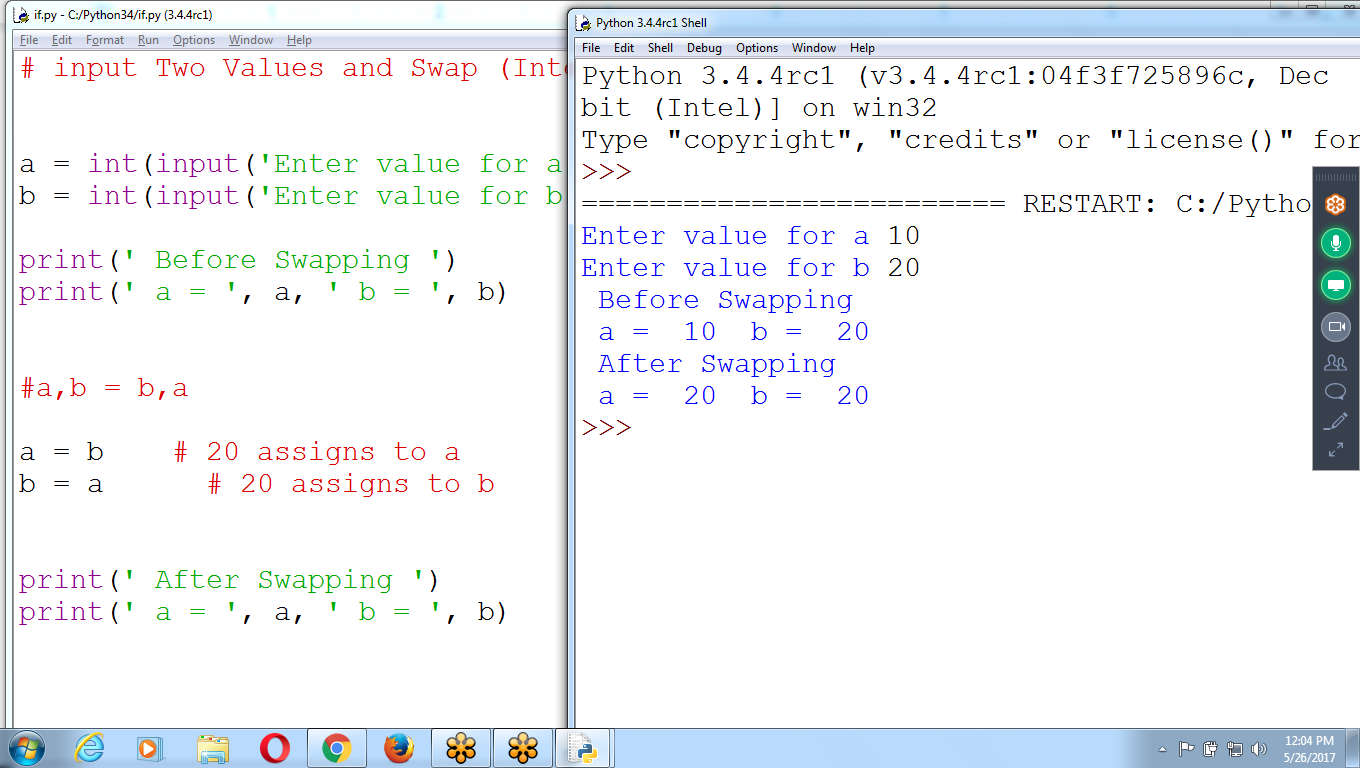
**print(' After Swapping ')**

**print(' a = ', a, ' b = ', b)**

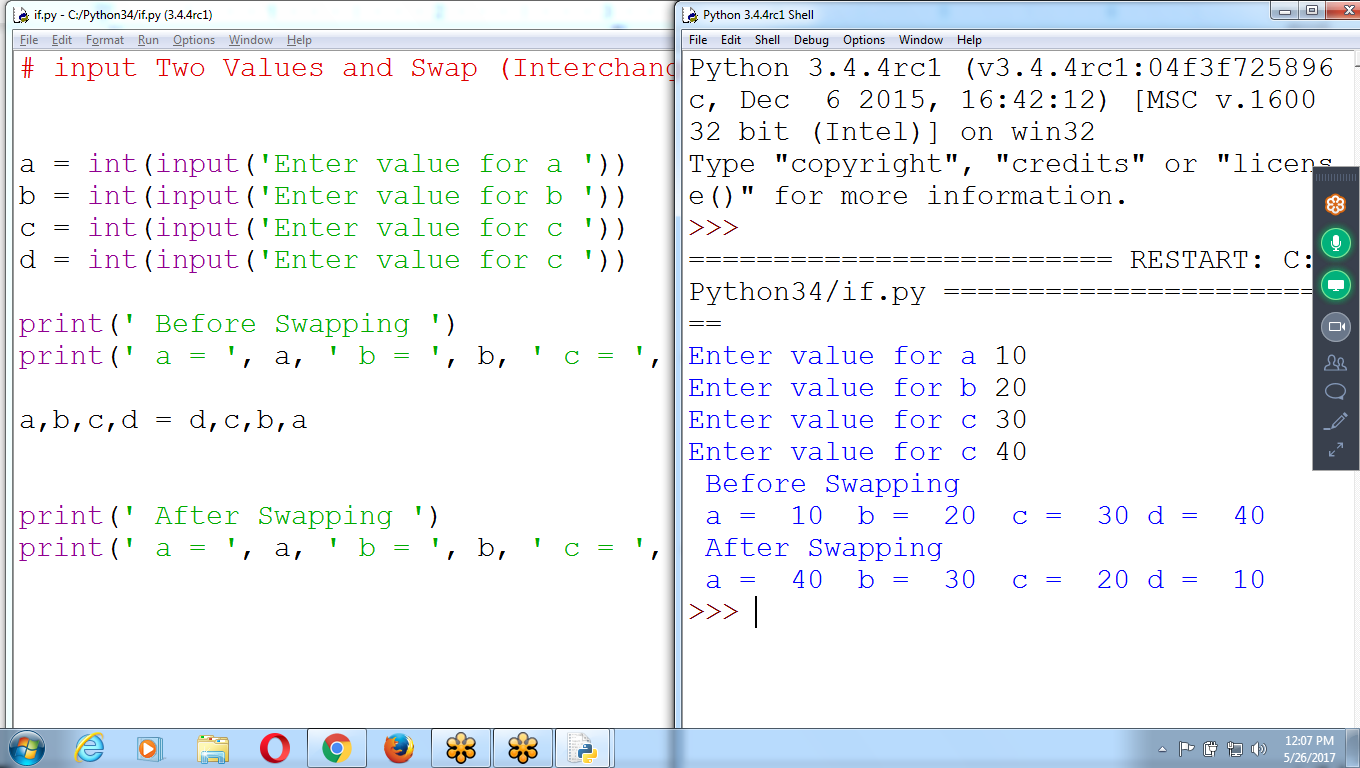
**Individual Statements**

**a=b**

**b=a**

****

Interchange of 4 Variables



# input Two Values and Swap (Interchange) Them

a = int(input('Enter value for a '))

b = int(input('Enter value for b '))

c = int(input('Enter value for c '))

d = int(input('Enter value for c '))

print(' Before Swapping ')

print(' a = ', a, ' b = ', b, ' c = ', c, 'd = ', d)

**a,b,c,d = d,c,b,a**

print(' After Swapping ')

print(' a = ', a, ' b = ', b, ' c = ', c, 'd = ', d)

Python Basics : 5 Days

**Python : 45 Days ( 3k) :: 6 Chapters**

Django : 45 Days ( 5k)

1 to 2:: Web developer :: 3 to 5 lacs

Data Science : 45 Days ( 150 Hours) ::

Timing : M to Fri :: 6 to 9 pm

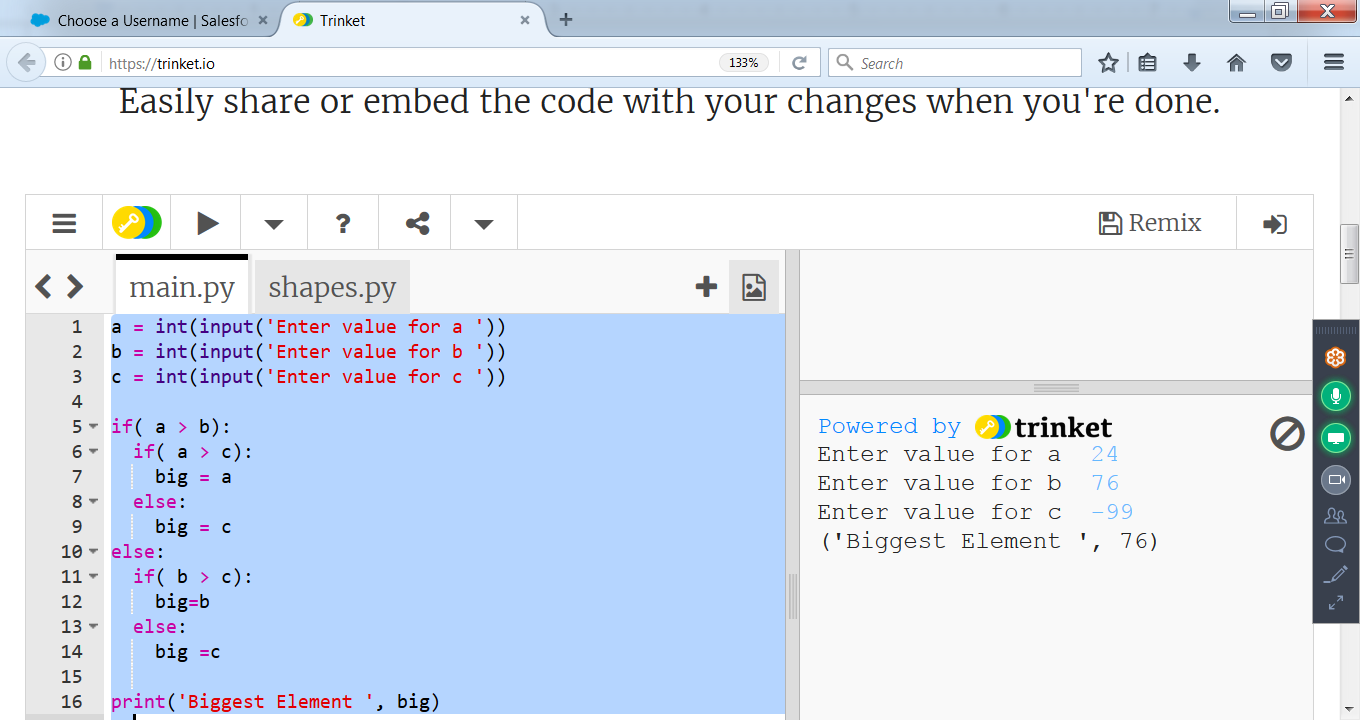
Fee: 15k

Python, R Program, NLP, SA, Machine Learning, Statistics

Advance DS : Deep Learning, Artificial Learning , Adv DL, ADV NLP

Timing: 8 Weekends

**Trinket.io :: Web based execution :: No need python installation**



a = int(input('Enter value for a '))

b = int(input('Enter value for b '))

c = int(input('Enter value for c '))

if( a > b):

if( a > c):

big = a

else:

big = c

else:

if( b > c):

big=b

else:

big =c

print('Biggest Element ', big)

**Python Basics**

**1 Installation, Expression, Data types, Variables, Execution (THus)**

**2 Operators and Flow Controls (If, if-else) (Fri)**

**3 Loops for, while, Range( Examples) (mon)**

**4 Functions and Collections( List, tuple, Dictionary) (Tue)**

## **For and while Loops and the range() Function**

* The while loop keeps looping while its condition is True
* To execute a block of code only a certain number of times?

Use for loop statement and the range() function.

for statement looks something like for i in range(5): and always includes the following:

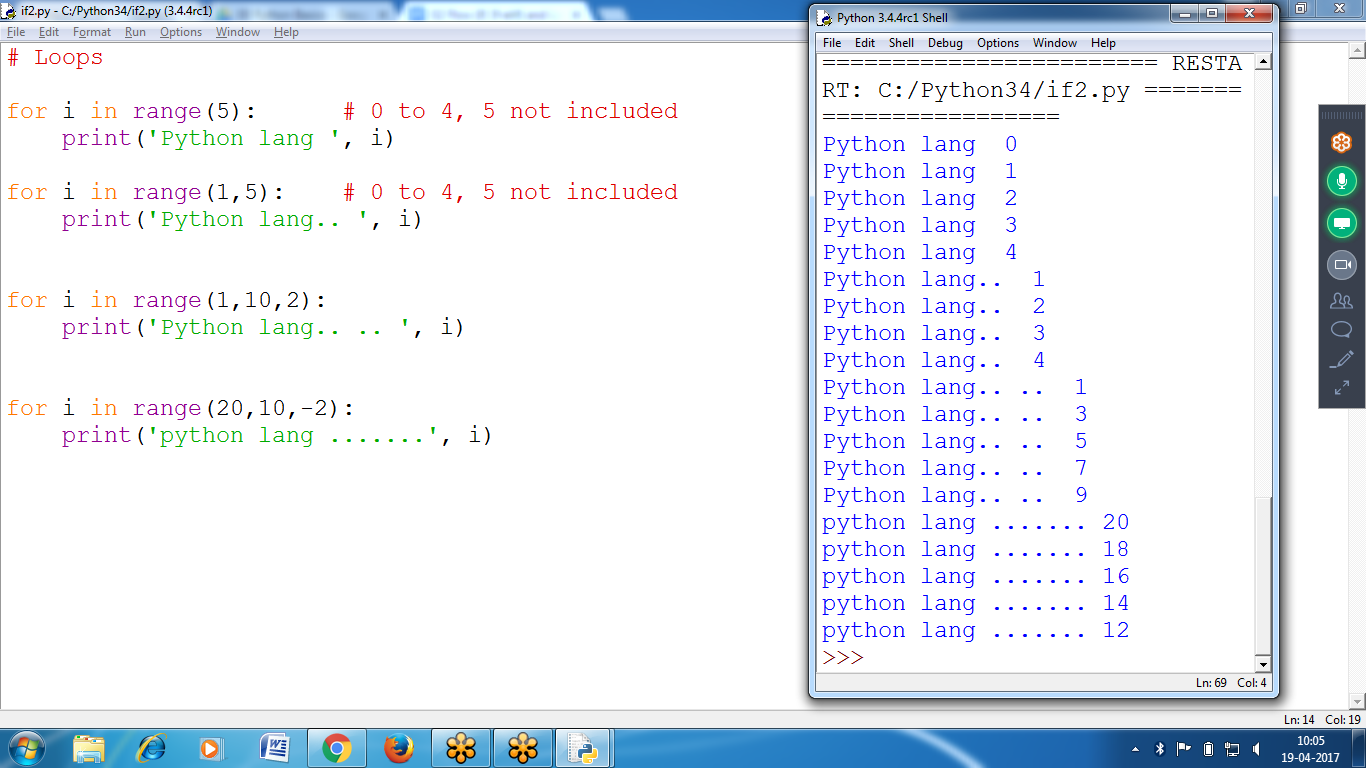
* The for keyword
* A variable name
* The in keyword
* A call to the range() method with up to three integers passed to it
* A colon
* Starting on the next line, an indented block of code

### **The Starting, Stopping, and Stepping Arguments to range()**

The first argument will be where the for loop’s variable start, and the second argument will be up to, but not including.

The range() function can also be called with three arguments.

The first two arguments will be the start and stop values, and the **third** will be the *step argument*.

****

**# Loops**

**for i in range(5): # 0 to 4, 5 not included**

**print('Python lang ', i)**

**for i in range(1,5): # 0 to 4, 5 not included**

**print('Python lang.. ', i)**

**for i in range(1,10,2):**

**print('Python lang.. .. ', i)**

**for i in range(20,10,-2):**

**print('python lang .......', i)**

## 

## 

## **Display values from 1 to 5 using for and while**

## 

## **# For and While Loops**

## 

## **for i in range(5):**

## **print(i, ' Python ')**

## **i=1**

## **while(i<5):**

## **print(i, 'Language ')**

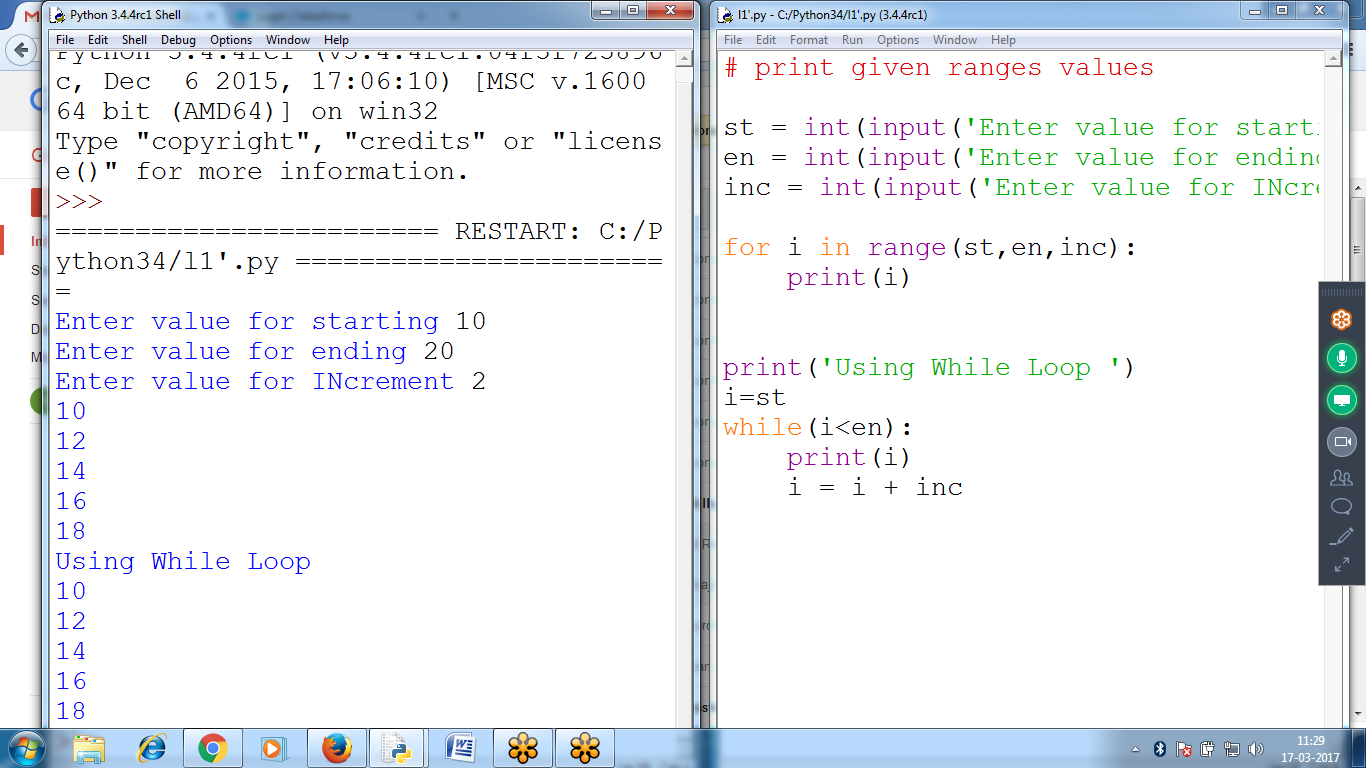
## **i = i+1**

## 

## 

## 

## **Display values within given range using starting, ending and increment values**



# print given ranges values

st = int(input('Enter value for starting '))

en = int(input('Enter value for ending '))

inc = int(input('Enter value for INcrement '))

for i in range(st,en,inc):

print(i)

print('Using While Loop ')

i=st

while(i<en):

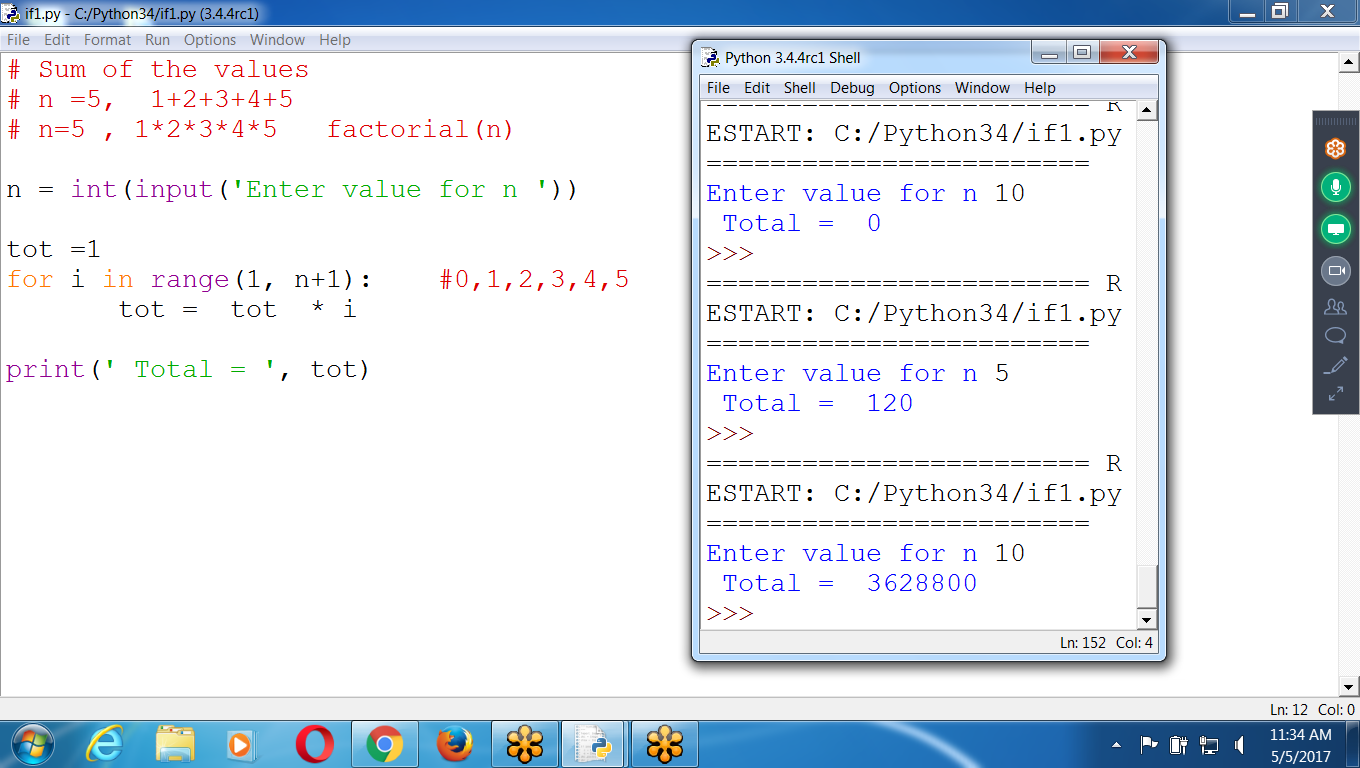
print(i)

i = i + inc

**To add all the numbers from 1 to 5**

**n =5 ::: 1 + 2 + 3 + 4 + 5**

# **Find Factorial of given number**



# Sum of the values

# n =5, 1+2+3+4+5

# n=5 , 1\*2\*3\*4\*5 factorial(n)

n = int(input('Enter value for n '))

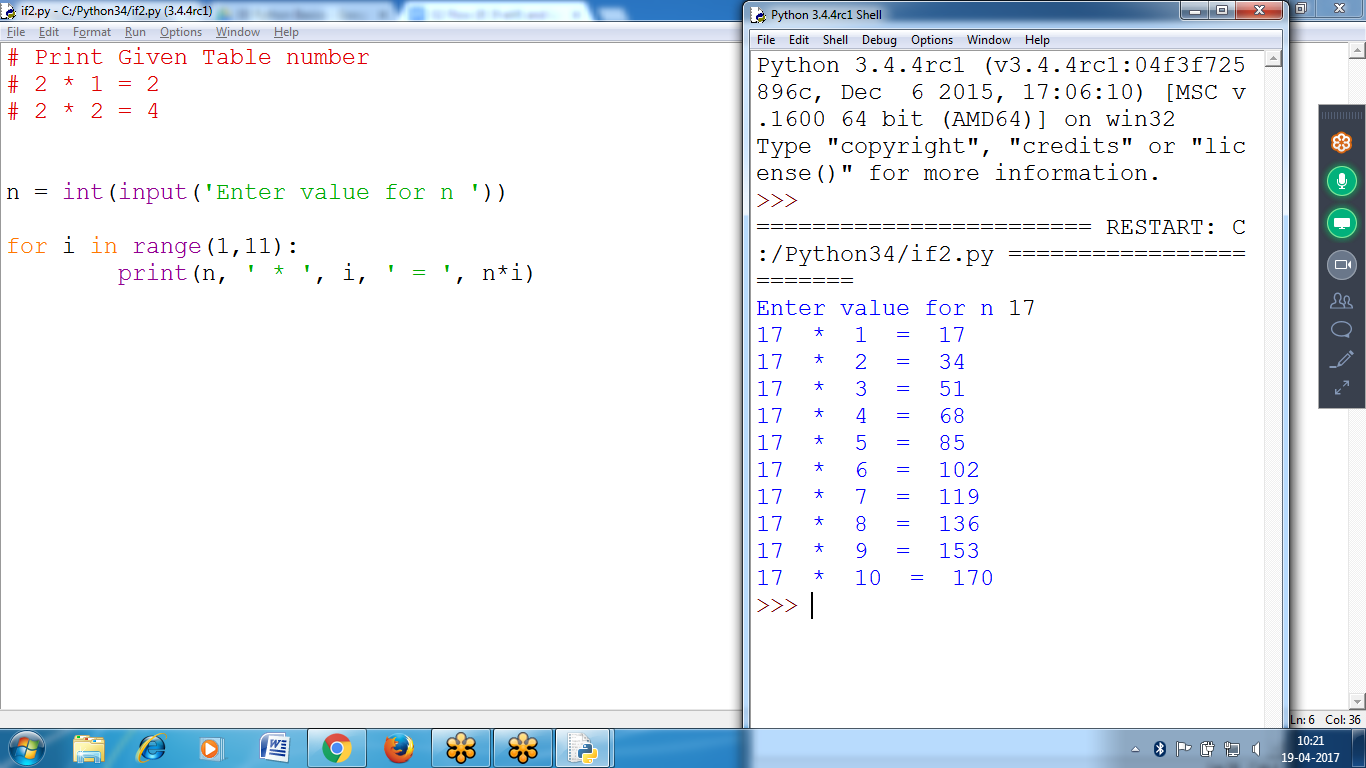
tot =1

for i in range(1, n+1): #0,1,2,3,4,5

**tot = tot \* i**

print(' Total = ', tot)

To Print Given Table Number



# Print Given Table number

# 2 \* 1 = 2

# 2 \* 2 = 4

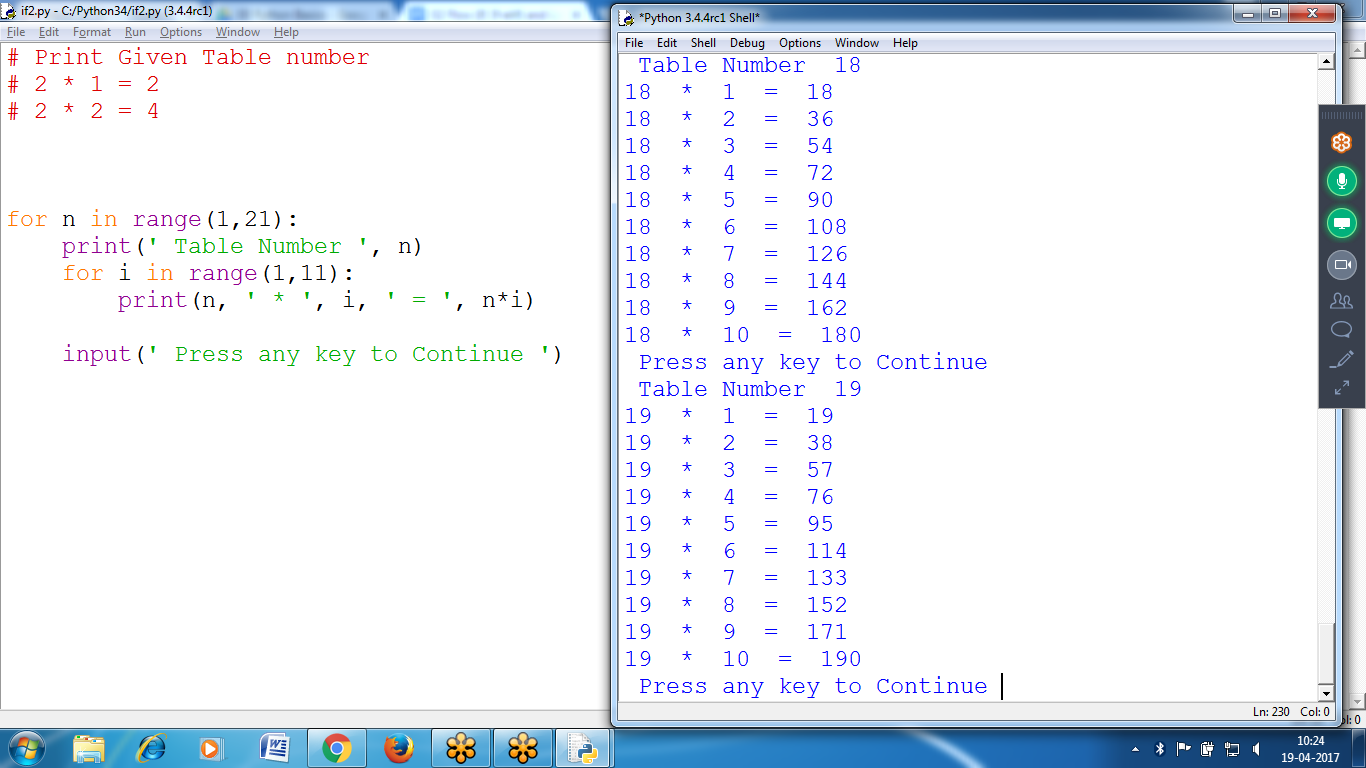
n = int(input('Enter value for n '))

for i in range(1,11):

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

Given Range of Tables using Nested Loop

Nested :: Loop within Loop



for n in range(1,21):

**print(' Table Number ', n)**

**for i in range(1,11):**

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

input(' Press any key to Continue ')

## **Break Statements**

If the execution reaches a **break statemen**t, it immediately exits the while loop’s clause.

* If you ever run a program that has a bug causing it to get stuck in an infinite loop, **press CTRL-C**. This will send a KeyboardInterrupt error to your program and cause it to **stop immediately**

**# Input name continuously until user press enter key or bye**

while True:

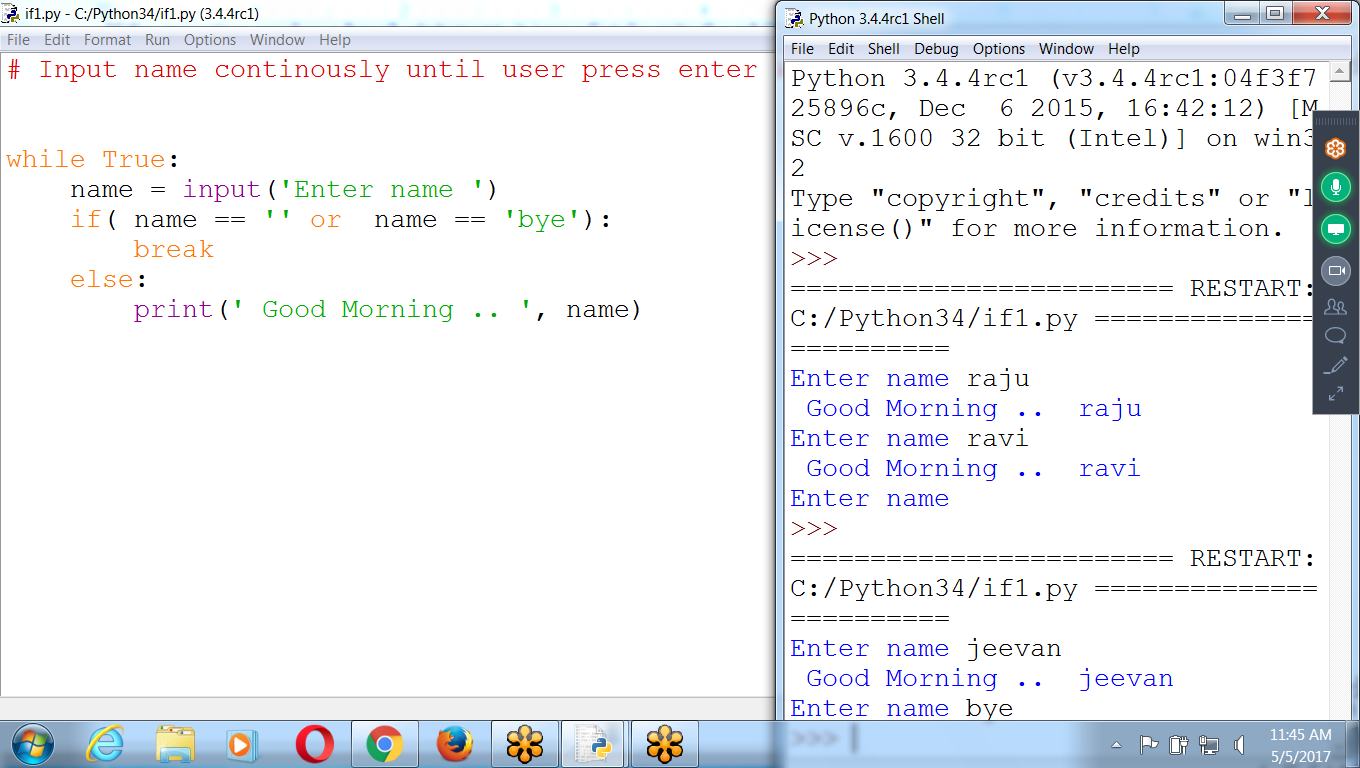
name = input('Enter name ')

if( name == '' or name == 'bye'):

break

else:

print(' Good Morning .. ', name)



**# Program to add number until user enter 0**

**# input number until user press 0**

**tot =0**

**while True:**

**n = int(input('Enter value for n '))**

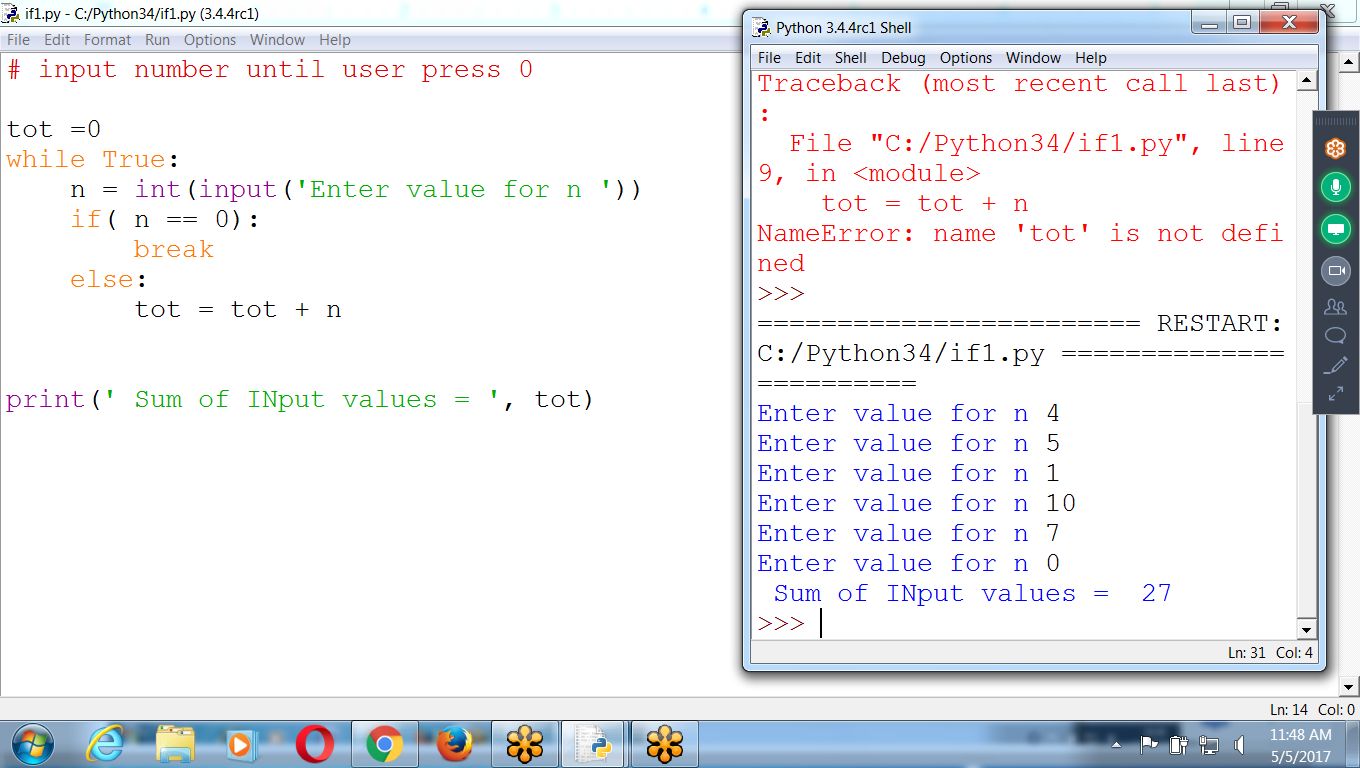
**if( n == 0):**

**break**

**else:**

**tot = tot + n**

**print(' Sum of INput values = ', tot)**

****

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

To Find Given Number is Prime or not

Enter number and find given number is prime or not

# To find given number is prime or not

# 1,2,3,5,7,11,13,17

# two divisors

# 17 : 1,2,3,4,,,,,, 17 = 1,17

# 12 : 1,2,3,4,,,, 12 :: 1,2,3,4,6,12

n = int(input('Enter value for n '))

count=0

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

if( n%i == 0):

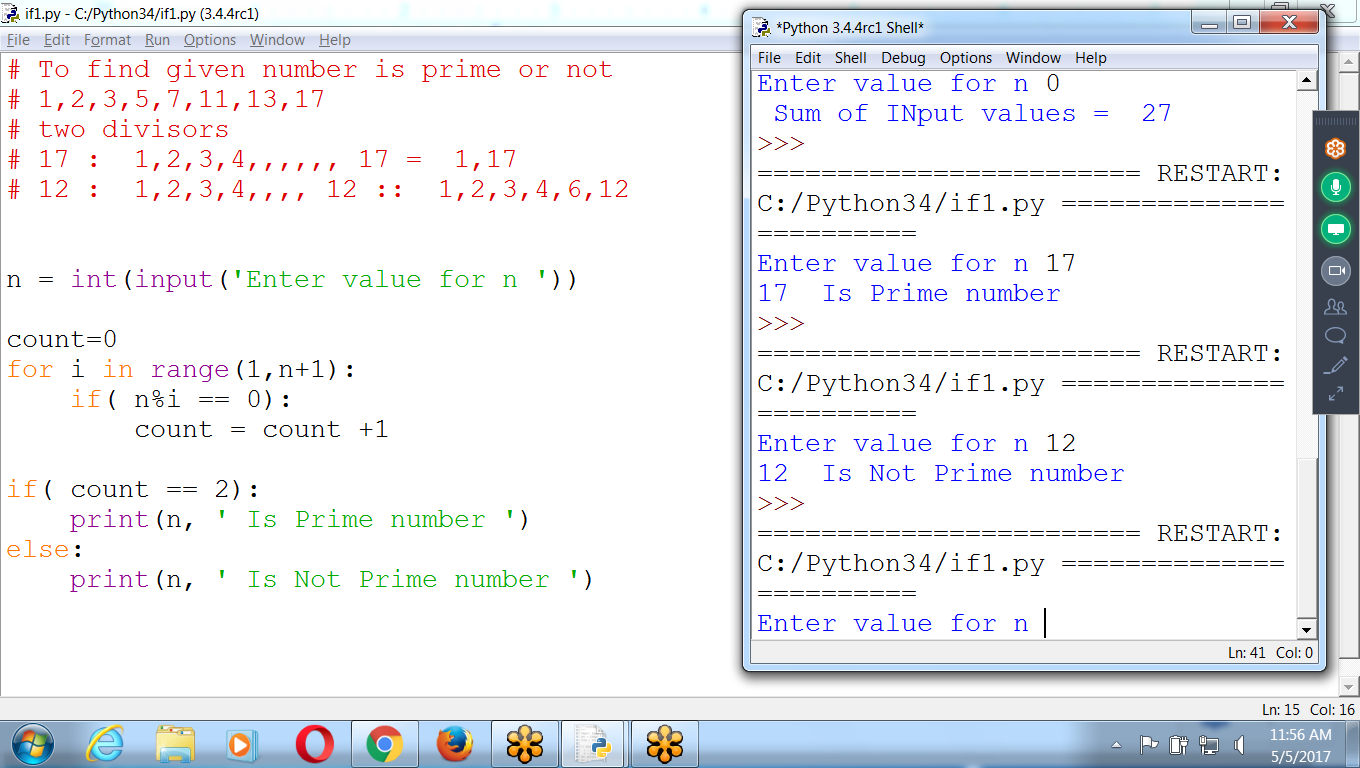
count = count +1

if( count == 2):

print(n, ' Is Prime number ')

else:

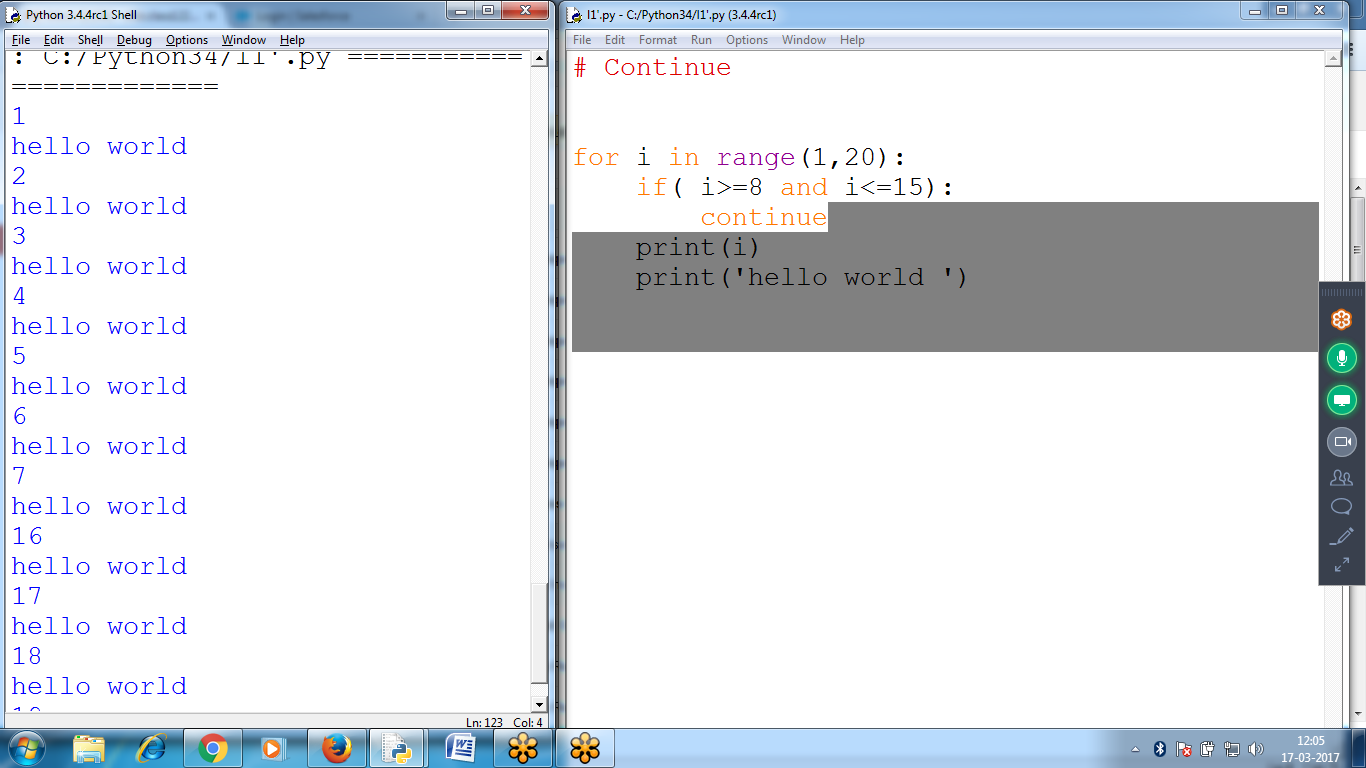
print(n, ' Is Not Prime number ')



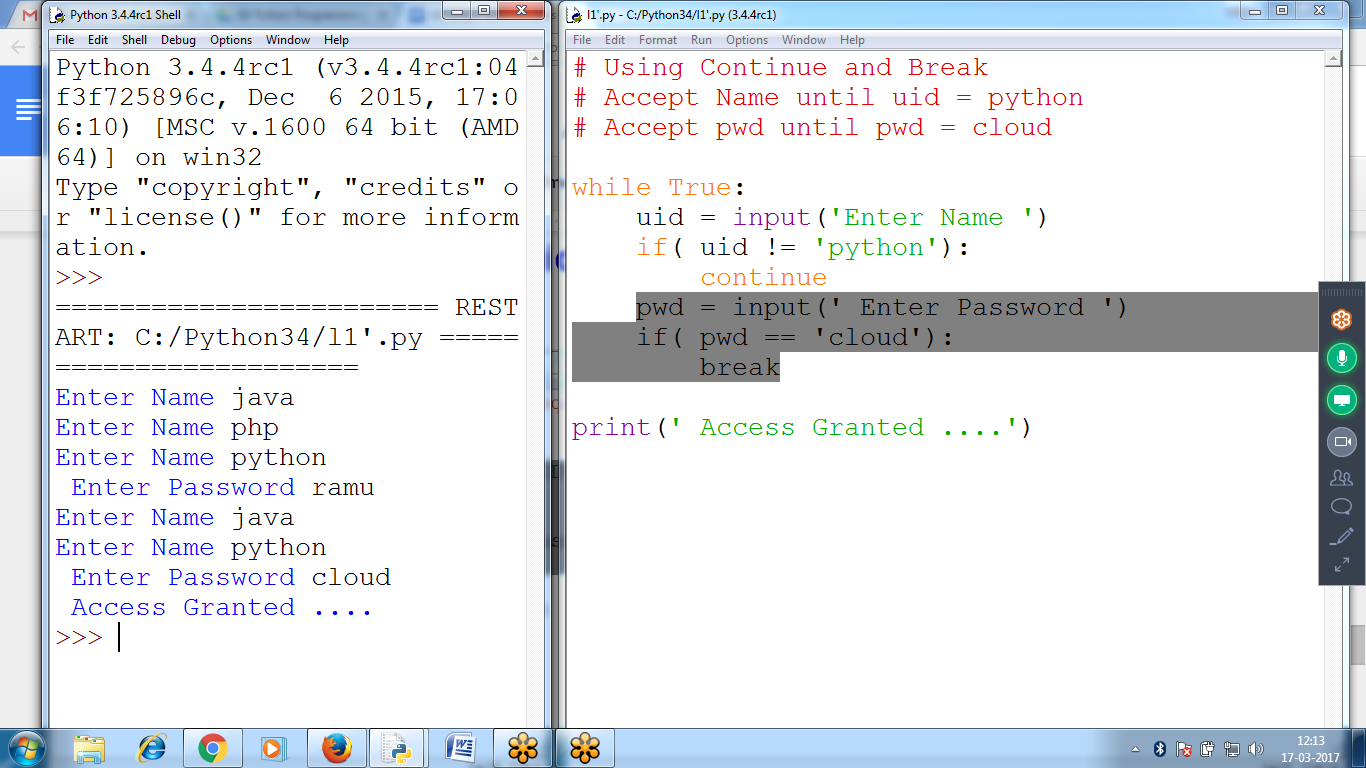
**Continue**

* When the program execution reaches a continue statement,

the program execution immediately jumps back to the start of the loop and reevaluates the loop’s condition.



**# Program to ACCEPT uid AND pwd UNTIL criteria satisfies**

****

**# Using Continue and Break**

**# Accept Name until uid = python**

**# Accept pwd until pwd = cloud**

**while True:**

**uid = input('Enter Name ')**

**if( uid != 'python'):**

**continue**

**pwd = input(' Enter Password ')**

**if( pwd == 'cloud'):**

**break**

**print(' Access Granted ....')**

# 