**Simple Python Programs**

# **Determine the type of an object in Python**

Python type() function

**type() function** is used to **determine the type of an object**, it accepts an object or value and returns it's type (i.e. a class of the object).

**Syntax:**

type(object)

**Example:**

Input:

b = 10.23

c = "Hello"

# Function call

print("type(b): ", type(b))

print("type(c): ", type(c))

Output:

type(b): <class 'float'>

type(c): <class 'str'>

### **Python code to determine the type of objects**

# Python code to determine the type of objects

# declaring objects and assigning values

a = 10

b = 10.23

c = "Hello"

d = (10, 20, 30, 40)

e = [10, 20, 30, 40]

# printing types of the objects

# using type() function

**print**("type(a): ", type(a))

**print**("type(b): ", type(b))

**print**("type(c): ", type(c))

**print**("type(d): ", type(d))

**print**("type(e): ", type(e))

# printing the type of the value

# using type() function

**print**("type(10): ", type(10))

**print**("type(10.23): ", type(10.23))

**print**("type(\"Hello\"): ", type("Hello"))

**print**("type((10, 20, 30, 40)): ", type((10, 20, 30, 40)))

**print**("type([10, 20, 30, 40]): ", type([10, 20, 30, 40]))

**Output**

type(a): <class 'int'>

type(b): <class 'float'>

type(c): <class 'str'>

type(d): <class 'tuple'>

type(e): <class 'list'>

type(10): <class 'int'>

type(10.23): <class 'float'>

type("Hello"): <class 'str'>

type((10, 20, 30, 40)): <class 'tuple'>

type([10, 20, 30, 40]): <class 'list'>

# **Python | Declare different types of variables, print their values, types and Ids**

There are two inbuilt functions are using in the program:

1. **type()** - its returns the **data type** of the variable/object.
2. **id()** - it returns the **unique identification number (id)** of created object/variable.

**Program:**

**print**("Numbers")

**print**("---------------------------------")

a=10

**print**(type(a),id(a),a)

a=23.7

**print**(type(a),id(a),a)

a=2+6j

**print**(type(a),id(a),a)

**print**("\nText")

**print**("---------------------------------")

a='h'

**print**(type(a),id(a),a)

a="h"

**print**(type(a),id(a),a)

a='hello'

**print**(type(a),id(a),a)

a="hello"

**print**(type(a),id(a),a)

**print**("\nBoolean")

**print**("---------------------------------")

a=True

**print**(type(a),id(a),a)

**print**("\nFunction")

**print**("---------------------------------")

**def** fun1():

**return** "I am Function"

a=fun1

**print**(type(a),id(a),a())

**print**("\nObjects")

**print**("---------------------------------")

**class** Demo:

**def** hi(self):

**return** "Hi"

a=Demo()

**print**(type(a),id(a),a.hi())

**print**("\nCollections")

**print**("---------------------------------")

a=[1,2,3]

**print**(type(a),id(a),a)

a=[]

**print**(type(a),id(a),a)

a=(1,2,3)

**print**(type(a),id(a),a)

a=()

**print**(type(a),id(a),a)

a=1,2,3

**print**(type(a),id(a),a)

a={1,2,3}

**print**(type(a),id(a),a)

a={}

**print**(type(a),id(a),a)

a={"id":1,"name":"pooja"}

**print**(type(a),id(a),a)

**Output**

Numbers

---------------------------------

<class 'int'> 10455328 10

<class 'float'> 139852163465696 23.7

<class 'complex'> 139852162869456 (2+6j)

Text

---------------------------------

<class 'str'> 139852162670976 h

<class 'str'> 139852162670976 h

<class 'str'> 139852162911512 hello

<class 'str'> 139852162911512 hello

Boolean

---------------------------------

<class 'bool'> 10348608 True

Function

---------------------------------

<class 'function'> 139852163226616 I am Function

Objects

---------------------------------

<class '\_\_main\_\_.Demo'> 139852162234016 Hi

Collections

---------------------------------

<class 'list'> 139852162259208 [1, 2, 3]

<class 'list'> 139852162261256 []

<class 'tuple'> 139852162244752 (1, 2, 3)

<class 'tuple'> 139852182028360 ()

<class 'tuple'> 139852162244968 (1, 2, 3)

<class 'set'> 139852163034472 {1, 2, 3}

<class 'dict'> 139852163024776 {}

<class 'dict'> 139852163024584 {'id': 1, 'name': 'pooja'}

# **Python | Typecasting Input to Integer, Float**

### **Typecasting input to integer**

**Syntax:**

int(input())

**Example:**

# input a number

num = int(input("Input a value: "))

# printing input value

**print** "num = ", num

**Output**

Input a value: 10

num = 10

### **Typecasting Input to float**

**Syntax:**

float(input())

**Example:**

# input a number

num = float(input("Input a value: "))

# printing input value

**print** "num = ", num

**Output**

Input a value: 10.23

num = 10.23

# **Python | Input two integers and find their addition**

**Program:**

# input two numbers: value of a and b

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

b = int(input("Enter B: "))

# find sum of a and b and assign to c

c = a+b

# print sum (c)

**print**("Sum: ",c)

**Output**

Enter A: 100

Enter B: 200

Sum: 300

**Explanation:**

Here, we are reading two values and assigning them in variable a and b - to input the value, we are using input() function, by passing the message to display to the user. Method input() returns a string value, and we are converting the input string value to the integer by using int() method.

After that, we are calculating the sum of a and b and assigning it to the variable c. And then, printing the value of c which is the sum of two input integers.

# **Python program to find sum of two numbers**

## ASCII value of character in Python

In python, to get an ASCII code of a character, we use ord() function. ord() accepts a character and returns the ASCII value of it.

**Example code:**

num1 = input("Enter first number: ")

num2 = input("Enter second number: ")

sum = int(num1) + int(num2)

**Example:**

Input:

num1 = 10

num2 = 20

Finding sum:

sum = num1 + num2

Output:

30

**Python code to find sum of two numbers**

# python program to find sum of

# two numbers

num1 = 10

num2 = 20

# finding sum

sum = num1 + num2

# printing sum

**print**("sum of ", num1, " and ", num2, " is = ", sum)

# taking input from user

num1 = input("Enter first number: ")

num2 = input("Enter second number: ")

# finding sum

sum = int(num1) + int(num2)

# printing sum

**print**("sum of ", num1, " and ", num2, " is = ", sum)

**Output**

sum of 10 and 20 is = 30

Enter first number: 100

Enter second number: 200

sum of 100 and 200 is = 300

# **Python program to print ASCII value of a character**

## ASCII value of character in Python

In python, to get an ASCII code of a character, we use ord() function. ord() accepts a character and returns the ASCII value of it.

**Syntax:**

ord(character);

**Example:**

Input:

char\_var = 'A'

Function call:

ord(char\_var)

Output:

65

**Python code to find ASCII value of a character**

# python program to print ASCII

# value of a given character

# Assigning character to a variable

char\_var = 'A'

# printing ASCII code

**print**("ASCII value of " + char\_var + " is = ", ord(char\_var))

char\_var = 'x'

# printing ASCII code

**print**("ASCII value of " + char\_var + " is = ", ord(char\_var))

char\_var = '9'

# printing ASCII code

**print**("ASCII value of " + char\_var + " is = ", ord(char\_var))

**Output**

ASCII value of A is = 65

ASCII value of x is = 120

ASCII value of 9 is = 57

# **Python program for simple interest**

Calculate simple interest

To calculate simple interest, we use the following formula,

**(P \* R \* T) / 100**

Where,

* **P** – Principle amount
* **R** – Rate of the interest, and
* **T** – Time in the years

**Example:**

Input:

p = 250000

r = 36

t = 1

# formula

si = (p\*r\*t)/100

print(si)

Output:

90000

Python program to find simple interest

# Python program to find simple interest

p = float(input("Enter the principle amount : "))

r = float(input("Enter the rate of interest : "))

t = float(input("Enter the time in the years: "))

# calculating simple interest

si = (p\*r\*t)/100

# printing the values

**print**("Principle amount: ", p)

**print**("Interest rate : ", r)

**print**("Time in years : ", t)

**print**("Simple Interest : ", si)

**Output**

First run:

Enter the principle amount : 10000

Enter the rate of interest : 3.5

Enter the time in the years: 1

Principle amount: 10000.0

Interest rate : 3.5

Time in years : 1.0

Simple Interest : 350.0

Second run:

Enter the principle amount : 250000

Enter the rate of interest : 36

Enter the time in the years: 1

Principle amount: 250000.0

Interest rate : 36.0

Time in years : 1.0

Simple Interest : 90000.0

Calculate compound interest

To calculate compound interest, we use the following formula,

**P(1 + R / 100)T**

Where,

* **P** – Principle amount
* **R** – Rate of the interest, and
* **T** – Time in the years

**Example:**

Input:

p = 250000

r = 36

t = 1

# formula

ci = p \* (pow((1 + r / 100), t))

print(ci)

Output:

339999.99999999994

Python program to find compound interest

# Python program to find compound interest

p = float(input("Enter the principle amount : "))

r = float(input("Enter the rate of interest : "))

t = float(input("Enter the time in the years: "))

# calculating compound interest

ci = p \* (pow((1 + r / 100), t))

# printing the values

**print**("Principle amount : ", p)

**print**("Interest rate : ", r)

**print**("Time in years : ", t)

**print**("compound Interest : ", ci)

**Output**

First run:

Enter the principle amount : 10000

Enter the rate of interest : 3.5

Enter the time in the years: 1

Principle amount : 10000.0

Interest rate : 3.5

Time in years : 1.0

compound Interest : 10350.0

Second run:

Enter the principle amount : 250000

Enter the rate of interest : 36

Enter the time in the years: 1

Principle amount : 250000.0

Interest rate : 36.0

Time in years : 1.0

compound Interest : 339999.99999999994

# **Simple pattern printing programs in Python**

**Pattern 1:**

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

**Code:**

**for** row **in** range (0,5):

**for** column **in** range (0, row+1):

**print** ("\*", end="")

# ending row

**print**('\r')

**Pattern 2:**

Now if we want to print numbers or alphabets in this pattern then we need to replace the **\*** with the desired number you want to replace. Like if we want pattern like,

1

1 1

1 1 1

1 1 1 1

1 1 1 1 1

**Code:**

#row operation

**for** row **in** range(0,5):

# column operation

**for** column **in** range(0,row+1):

**print**("1 ",end="")

# ending line

**print**('\r')

**Pattern 3:**

If want increasing numbers in this pattern like,

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

Here we need to declare a starting number from which the patter will start. In the above case the number is starting from 1. So, here we have to create a variable and assigns its value to 1 then we need to print only the value of variable.

As its value is increasing every row by 1, but starting value is always 1.

So, for that we have to declare the value of the starting number before column operation (second for loop) and need to increase it by 1 after the column operation section after the printing value.

**Code:**

#row operation

**for** row **in** range (0, 5):

n = 1

# column operation

**for** column **in** range (0, row+1):

**print**(n, end=" ")

n = n+1

# ending line

**print**('\r')

**Pattern 4:**

1

2 3

4 5 6

7 8 9 10

11 12 13 14

To get the above pattern only we have to declare the variable before the row operation. Follow the code below,

**Code:**

n = 1

#row operation

**for** row **in** range (0, 5):

# column operation

**for** column **in** range (0, row+1):

**print**(n, end=" ")

n = n+1

# ending line

**print**('\r')

**Pattern 5:**

A

A B

A B C

A B C D

A B C D E

The above pattern can also be another type.

For that should have the knowledge of [ASCII values](https://www.includehelp.com/ascii-table.aspx) of 'A'.

Its [ASCII value](https://www.includehelp.com/ascii-table.aspx) is 65.

In column operation We have to convert the ASCII value to character using [chr() function](https://www.includehelp.com/python/chr-function-with-example.aspx).

**Code:**

#row operation

**for** row **in** range (0, 5):

n = 65

# column operation

**for** column **in** range (0, row+1):

c = chr(n)

**print**(c, end=" ")

n = n+1

# ending line

**print**('\r')

# **Python program to find power of a number using exponential operator**

**Example:**

Input:

a = 10

b = 3

# calculating power using exponential oprator (\*\*)

result = a\*\*b

print(result)

Output:

1000

**Finding power of integer values**

# python program to find the power of a number

a = 10

b = 3

# calculating power using exponential oprator (\*\*)

result = a\*\*b

**print** (a, " to the power of ", b, " is = ", result)

**Output**

10 to the power of 3 is = 1000

**Finding power of float values**

# python program to find the power of a number

a = 10.23

b = 3.2

# calculating power using exponential oprator (\*\*)

result = a\*\*b

**print** (a, " to the power of ", b, " is = ", result)

**Output**

10.23 to the power of 3.2 is = 1704.5197114724524

# **Python program to find the power of a number using loop**

### **Python code to find power of a number using loop**

num = int(input("Enter the number of which you have to find power: "))

pw = int(input("Enter the power: "))

kj = 1

**for** n **in** range(pw):

kj = kj\*num

**print**(kj)

**Output**

Enter the number of which you have to find power: 5

Enter the power: 4

625

# **Python program to extract and print digits in reverse order of a number**

### **Python code to extract and print digits of a number in reverse order**

num = int(input("Enter a number with multiple digit: "))

n=0

**while** num>0:

a = num%10

num = num - a

num = num/10

**print**(int(a),end="")

n = n + 1

**print**(n)

**Output**

Enter a number with multiple digit: 123456789

9876543219

Here we are first using a loop with condition num>0, and the last digit of the number is taken out by using simple % operator after that, the remainder term is subtracted from the num. Then number num is reduced to its 1/10th so that the last digit can be truncated.

The cycle repeats and prints the reverse of the number num.

# **Python program to find floor division**

**Python code to find floor division**

# python program to find floor division

a = 10

b = 3

# finding division

result1 = a/b

**print**("a/b = ", result1)

# finding floor division

result2 = a//b

**print**("a/b = ", result2)

**Output**

a/b = 3.3333333333333335

a/b = 3

# **Python | Some of the examples of simple if else**

**Example1: Enter a number and check whether it is 10 or not**

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

**if** a==10:

**print**("Equal to 10")

**else**:

**print**("Not Equal to 10")

**Output**

Enter A : 10

Equal to 10

**Example2: Find largest of two numbers**

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

b=int(input("Enter B: "))

**if** a>b:

g=a

**else**:

g=b

**print**("Greater = ",g)

**Output**

Enter A: 36

Enter B: 24

Greater = 36

**Example3: Find largest of two numbers using single statement**

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

b=int(input("Enter B: "))

c= a **if** a>b **else** b

**print**("Greater = ",c)

**Output**

Enter A: 24

Enter B: 36

Greater = 36

# **Python | Input age and check eligibility for voting**

**Program:**

# input age

age = int(input("Enter Age : "))

# condition to check voting eligibility

**if** age>=18:

status="Eligible"

**else**:

status="Not Eligible"

**print**("You are ",status," for Vote.")

**Output**

Enter Age : 19

You are Eligible for Vote.

# **Python | Find largest of three number using nested if else**

**Input three integer numbers and find the largest of them using nested if else in python.**

**Example:**

Input:

Enter first number: 10

Enter second number: 20

Enter third number: 5

Output:

Largest number: 20

**Program:**

# input three integer numbers

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

b=int(input("Enter B: "))

c=int(input("Enter C: "))

# conditions to find largest

**if** a>b:

**if** a>c:

g=a

**else**:

g=c

**else**:

**if** b>c:

g=b

**else**:

g=c

# print the largest number

**print**("Greater = ",g)

**Output**

Enter A: 10

Enter B: 20

Enter C: 5

Greater = 20

# **Python | Calculate discount based on the sale amount**

**Input same amount and calculate discount based on the amount and given discount rate in Python.**

The discount rates are:

Amount Discount

0-5000 5%

5000-15000 12%

15000-25000 20%

above 25000 30%

**Program:**

# input sale amount

amt = int(input("Enter Sale Amount: "))

# checking conditions and calculating discount

**if**(amt>0):

**if** amt<=5000:

disc = amt\*0.05

**elif** amt<=15000:

disc=amt\*0.12

**elif** amt<=25000:

disc=0.2 \* amt

**else**:

disc=0.3 \* amt

**print**("Discount : ",disc)

**print**("Net Pay : ",amt-disc)

**else**:

**print**("Invalid Amount")

**Output**

Enter Sale Amount: 30000

Discount : 9000.0

Net Pay : 21000.0

# **Python | Calculate discount based on the sale amount using Nested if else**

**Input same amount and calculate discount based on the amount and given discount rate in Python.**

**The discount rates are:**

Amount Discount

0-5000 5%

5000-15000 12%

15000-25000 20%

above 25000 30%

**Program:**

# input sale amount

amt = int(input("Enter Sale Amount: "))

# conditions to check amount and get discount

**if**(amt>0):

**if** amt<=5000:

disc = amt\*0.05

**else**:

**if** amt<=15000:

disc=amt\*0.12

**else**:

**if** amt<=25000:

disc=0.2 \* amt

**else**:

disc=0.3 \* amt

# printing discount and net payable amount

**print**("Discount : ",disc)

**print**("Net Pay : ",amt-disc)

**else**:

**print**("Invalid Amount")

**Output**

Enter Sale Amount: 22000

Discount : 4400.0

Net Pay : 17600.0

# **Python | Example of Ternary Operator**

**iven age of a person and we have to check whether person is eligible for voting or not using Ternary operator.**

**Syntax:**

[on\_true] if [expression] else [on\_false]

**Here,**

* [on\_true] is the statement that will be execute if the given condition [expression] is true.
* [expression] is the conditional expression to be checked.
* [on\_false] is the statement that will be executed if the given condition [expression] is false.

**Example:**

Input:

Enter Age :21

Output:

You are Eligible for Vote.

**Program:**

# input age

age = int(input("Enter Age :"))

# condition

status = "Eligible" **if** age>=18 **else** "Not Eligible"

# print message

**print**("You are",status,"for Vote.")

**Output**

Enter Age :21

You are Eligible for Vote.

# **Python | Design a simple calculator using if elif**

**Given two numbers and we have to design a calculator type application that will perform add, subtract, multiply and divide operations using Python.**

**Example:**

Message:

Calculator

1.Add

2.Substract

3.Multiply

4.Divide

Input:

Enter Choice(1-4): 3

Enter A:10

Enter B:20

Output:

Product = 200

**Program:**

# menus

**print**("Calculator")

**print**("1.Add")

**print**("2.Substract")

**print**("3.Multiply")

**print**("4.Divide")

# input choice

ch=int(input("Enter Choice(1-4): "))

**if** ch==1:

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

b=int(input("Enter B:"))

c=a+b

**print**("Sum = ",c)

**elif** ch==2:

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

b=int(input("Enter B:"))

c=a-b

**print**("Difference = ",c)

**elif** ch==3:

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

b=int(input("Enter B:"))

c=a\*b

**print**("Product = ",c)

**elif** ch==4:

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

b=int(input("Enter B:"))

c=a/b

**print**("Quotient = ",c)

**else**:

**print**("Invalid Choice")

**Output**

Calculator

1.Add

2.Substract

3.Multiply

4.Divide

Enter Choice(1-4): 3

Enter A:10

Enter B:20

Product = 200

# **Python | Demonstrate an example of for loop**

**Example:**

Here, we are running loop for given ranges with various arguments like argument 1,2,3 and reverse order of the loop.

**Program:**

**print**("Type 1")

**for** i **in** range(10): # start=0 , end=10,step=1

**print**(i,end=" ")

**print**("\nType 2")

**for** i **in** range(1,11): # start=1 , end=10,step=1

**print**(i,end=" ")

**print**("\nType 3")

**for** i **in** range(1,11,3): # start=1 , end=10,step=3

**print**(i,end=" ")

**print**("\nType 4")

**for** i **in** range(10,0,-1): # start=10 , end=0,step=-1

**print**(i,end=" ")

**Output**

Type 1

0 1 2 3 4 5 6 7 8 9

Type 2

1 2 3 4 5 6 7 8 9 10

Type 3

1 4 7 10

Type 4

10 9 8 7 6 5 4 3 2 1

# **Python | Demonstrate an example of for each loop**

**Example:**

**for each loop** is used with container type of data type like list, in this example, we have a list of fruits, we are printing its type and individual values (elements) using for each loop<a.< p="" style="box-sizing: border-box;"></a.<>

**Program:**

# declare and initialize a list

fruits = ["apple","mango","guava","grapes","pinapple"]

# pritning type of fruits

**print** (type(fruits))

# printing value

**for** fruit **in** fruits:

**print**(fruit)

**Output**

<class 'list'>

apple

mango

guava

grapes

pinapple

# **Python | Examples of loops (based on their control)**

Based on loop controls, here are examples of following types:

1. Condition Controlled Loop
2. Range Controlled loop
3. Collection Controlled loop

### **1) Condition Controlled Loop**

# Condition Controlled Loop

a=1

**while** a<=10:

**print**(a)

a=a+1

**Output**

1

2

3

4

5

6

7

8

9

10

### **2) Range Controlled loop**

#Range Controlled Loop

#range(end) start=0,step=+1

**for** i **in** range(10):

**print**(i,end=" ")

**print**("\n")

#range(start,end) step=+1

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

**print**(i,end=" ")

**print**("\n")

#range(start,end,step)

**for** i **in** range(1,11,3):

**print**(i,end=" ")

**print**()

**for** i **in** range(10,0,-1):

**print**(i,end=" ")

**Output**

0 1 2 3 4 5 6 7 8 9

1 2 3 4 5 6 7 8 9 10

1 4 7 10

10 9 8 7 6 5 4 3 2 1

### **3) Collection Controlled loop**

#Collection Controlled Loop

fruits=["apple","banana","guava","grapes","oranges"]

**for** item **in** fruits:

**print**(item)

**Output**

apple

banana

guava

grapes

oranges

# **Python | Some of the Examples of loops**

**. Print all the no. between 1 to n**

n=int(input("Enter N: "))

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

**print**(i)

**Output**

Enter N: 5

1

2

3

4

5

**2. Print table of number**

n=int(input("Enter N: "))

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

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

**Output**

Enter N: 2

2 x 1 = 2

2 x 2 = 4

2 x 3 = 6

2 x 4 = 8

2 x 5 = 10

2 x 6 = 12

2 x 7 = 14

2 x 8 = 16

2 x 9 = 18

2 x 10 = 20

**3. Print sum of n number**

n=int(input("Enter N: "))

s=0

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

s=s+i

**print**("Sum = ",s)

**Output**

Enter N: 10

Sum = 55

**4. Print factorial of n**

n=int(input("Enter N: "))

f=1

**for** i **in** range(n,0,-1):

f=f\*i

**print**("Factorial = ",f)

**Output**

Enter N: 4

Factorial = 24

**5. Check prime number**

n=int(input("Enter N: "))

c=0

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

**if** n%i==0:

c=c+1

**if** c==2:

**print**(n,"is Prime")

**else**:

**print**(n,"is Not Prime")

**Output**

Enter N: 131

131 is Prime

**6. Check palindrome number**

n=int(input("Enter Number: "))

m=n

rev=0

**while**(n>0):

dig=n%10

rev=rev\*10+dig

n=n//10

**if** rev==m:

**print**(m,"is Palindrome")

**else**:

**print**(m,"is not Palindrome")

**Output**

Enter Number: 12321

12321 is Palindrome

# **Python | Demonstrate an Example of break statement**

break is a keyword in python just like another programming language and it is used to break the execution of loop statement.

In the given example, loop is running from 1 to 10 and we are using the break statement if the value of i is 6. Thus when the value of i will be 6, program's execution will come out from the loop.

**Example 1:**

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

**if**(i==6):

**break**

**print**(i)

**Output**

1

2

3

4

5

**Example 2:** In this example, we are printing character by character of the value/string “Hello world” and terminating (using break), if the character is space.

**for** ch **in** "Hello world":

**if** ch == " ":

**break**

**print**(ch)

**Output**

H

e

l

l

o

# **Python | Demonstrate an Example of continue statement**

continue is a keyword in python just like another programming language and it is used to send the program’s section to loop by escaping the execution of next statement in the loop.

In the given example, loop is running from 1 to 10 and we are using the continue statement if value of ‘i’ is 6. This when the value of i will be 6, program’s execution will continue without prating the 6.

**Example 1:**

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

**if**(i==6):

**continue**

**print**(i)

**Output**

1

2

3

4

5

7

8

9

10

**Example 2:** In this example, we are printing character by character of the value/string “Hello world” and continuing the loop execution, if the character is space.

**for** ch **in** "Hello world":

**if** ch == " ":

**continue**

**print**(ch)

**Output**

H

e

l

l

o

w

o

r

l

d

# **Python | Demonstrate an Example of pass statement**

## ass statement in python

**"pass"** is a type of null operation or null statement, when it executes nothing happens. It is used when you want do not want to write any code/statement to execute but syntactically a statement is required.

**Let’s consider the given example...**

Here, we are using **"pass"** statement in the function **hello()** definition - there is no statement in the function and if we do not use **"pass"**, there will be an error.

**def** hello():

**pass**

hello()

**Example 2:**

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

**if**(i==6):

**continue**

**else**:

**pass**

**print**(i)

**Output**

1

2

3

4

5

7

8

9

10

# **Python | Program to print numbers from N to 1 (use range() with reverse order)**

**Given the value of N and we have to print numbers from N to 1 in Python.**

## range() Method

This method is used to iterate a range values.

Simply, we use range(start, stop)

Let’s understand by an example, if we want to iterate any loop till a to b, then range statement will be range(a, b+1).

**Iterate in reverse order**

To iterate range in reverse order, we use 3 parameters

1. Start – start value
2. Stop – end value
3. Step – Increment/Decrement to the value

**Examples:**

1) To print numbers from B to A

for i in range(B, A-1, -1)

print i

2) To print numbers from B to A by escaping one number between

for i in range(B, A-1, -2)

print i

## Program to print numbers from N to 1 in Python

# Python program to print numbers

# from n to 1

# input the value of n

n = int(input("Enter the value of n: "))

# check the input value

**if** (n<=1):

**print** "n should be greater than 1"

exit()

# print the value of n

**print** "value of n: ",n

# print the numbers from n to 1

# message

**print** "numbers from {0} to {1} are: ".format(n,1)

# loop to print numbers

**for** i **in** range(n,0,-1):

**print** i

**Output 1**

Enter the value of n: 10

value of n: 10

numbers from 10 to 1 are:

10

9

8

7

6

5

4

3

2

1

**Output 2** (when the value of n is 1)

Enter the value of n: 1

n should be greater than 1

# **Python | Print all numbers between 1 to 1000 which are divisible by 7 and must not be divisible by 5**

**Logic:**

* To implement this logic, we will use a [for and in](https://www.includehelp.com/python/looping-constructs.aspx) loop with range() method. The statement of range() method with the minimum to maximum range is range(begin, end+1).
* And, check the condition, that value should be divisible by 7 and should not be divisible by 5 (example code: ((cnt%7==0) and (cnt%5!=0)) ).
* If condition is true, print the numbers.

**Program:**

# define range in variables

# so that we can change them anytime

begin = 1

end = 1000

# loop to check and print the numbers

# which are divisible by 7 and not

# divisible by 5

**for** cnt **in** range(begin, end+1):

**if**( cnt%7==0 **and** cnt%5!=0 ):

**print** cnt, # command after cnt will print space

**Output**

7 14 21 28 42 49 56 63 77 84 91 98 112 119 126

133 147 154 161 168 182 189 196 203 217 224 231

238 252 259 266 273 287 294 301 308 322 329 336

343 357 364 371 378 392 399 406 413 427 434 441

448 462 469 476 483 497 504 511 518 532 539 546

553 567 574 581 588 602 609 616 623 637 644 651

658 672 679 686 693 707 714 721 728 742 749 756

763 777 784 791 798 812 819 826 833 847 854 861

868 882 889 896 903 917 924 931 938 952 959 966

973 987 994

# **Python | Calculate square of a given number (3 different ways)**

**Given a number, and we have to calculate its square in Python.**

**Example:**

Input:

Enter an integer numbers: 8

Output:

Square of 8 is 64

**Calculating square** is a basic operation in mathematics; here we are calculating the square of a given number by using 3 methods.

1. By multiplying numbers two times: (number\*number)
2. By using Exponent Operator (\*\*): (number\*\*2)
3. By using math.pow() method: (math.pow(number,2)

### **1) By multiplying numbers two times: (number\*number)**

To **find the square of a number** - simple multiple the number two times.

**Program:**

# Python program to calculate square of a number

# Method 1 (using number\*number)

# input a number

number = int (raw\_input ("Enter an integer number: "))

# calculate square

square = number\*number

# print

**print** "Square of {0} is {1} ".format (number, square)

**Output**

Enter an integer number: 8

Square of 8 is 64

### **2) By using Exponent Operator (\*\*): (number\*\*2)**

The another way is to **find the square of a given number** is to use **Exponent Operator** (\*\*), it returns the exponential power. This operator is represented by \*\*

**Example:** Statement m\*\*n will be calculated as **"m to the power of n"**.

**Program:**

# Python program to calculate square of a number

# Method 2 (using number\*\*2)

# input a number

number = int (raw\_input ("Enter an integer number: "))

# calculate square

square = number\*\*2

# print

**print** "Square of {0} is {1} ".format (number, square)

**Output**

Enter an integer number: 8

Square of 8 is 64

### **3) By using math.pow() method: (math.pow(number,2)**

pow(m,n) is an inbuilt method of math library, it returns the value of **"m to the power n"**. To use this method, we need to import math library in the program.

The statement to import math library is import math.

**Program:**

# Python program to calculate square of a number

# Method 3 (using math.pow () method)

# importing math library

**import** math

# input a number

number = int (raw\_input ("Enter an integer number: "))

# calculate square

square = int(math.pow (number, 2))

# print

**print** "Square of {0} is {1} ".format (number, square)

**Output**

Enter an integer number: 8

Square of 8 is 64

# **Python | Find factorial of a given number (2 different ways)**

**Given a number and we have to find its factorial in Python.**

**Example:**

Input:

Num = 4

Output:

Factorial of 4 is: 24

### **1) Method 1: Using loop**

# Code to find factorial on num

# number

num = 4

# 'fact' - variable to store factorial

fact =1

# run loop from 1 to num

# multiply the numbers from 1 to num

# and, assign it to fact variable

**for** i **in** range (1,num+1) :

fact = fact\*i

# print the factorial

**print** "Factorial of {0} is: {1} ".format (num, fact)

**Output**

Factorial of 4 is: 24

### **2) Method 2: by creating a function using recursion method**

To find the factorial, fact() function is written in the program. This function will take number (num) as an argument and return the factorial of the number.

# function to calculate the factorial

**def** fact (n):

**if** n == 0:

**return** 1

**return** n \* fact (n - 1)

# Main code

num = 4

# Factorial

**print** "Factorial of {0} is: {1} ".format (num, fact(num))

**Output**

Factorial of 4 is: 24

# **Python | Write functions to find square and cube of a given number**

**Given a number, and we have to write user defined functions to find the square and cube of the number is Python.**

**Example:**

Input:

Enter an integer number: 6

Output:

Square of 6 is 36

Cube of 6 is 216

**Function to get square:**

**def** square (num):

**return** (num\*num)

**Function to get cube:**

**def** cube (num):

**return** (num\*num\*num)

**Program:**

# python program to find square and cube

# of a given number

# User defind method to find square

**def** square (num):

**return** (num\*num)

# User defind method to find cube

**def** cube (num) :

**return** (num\*num\*num)

# Main code

# input a number

number = int (raw\_input("Enter an integer number: "))

# square and cube

**print** "square of {0} is {1}".format(number, square(number))

**print** "Cube of {0} is {1}".format(number, cube (number))

**Output**

Enter an integer number: 6

square of 6 is 36

Cube of 6 is 216

# **Python | Declare any variable without assigning any value**

Since, Python is a dynamic programming language so there is no need to declare such type of variable, it automatically declares when first time value assign in it.

Still, this is a common question asked by many programmers that **can we declare any variable without any value?**

**The answer is: "Yes! We can declare such type of variable"**. To declare a variable without any variable, just assign None.

**Syntax:**

variable\_name = None

**Example:**

num = None

**Let’s understand through a program:**

# Python program to declare a

# variable without assigning any value

# declare variable

num = None

# print the value

**print** "value of num: ", num

# checking variable

**if** (num==None):

**print** "Nothing"

**else**:

**print** "Something"

# assign some value

num = 100

# print the value

**print** "value of num: ", num

# checking variable

**if** (num==None):

**print** "Nothing"

**else**:

**print** "Something"

**Output**

value of num: None

Nothing

value of num: 100

Something

# **BMI (Body Mass Index) calculator in Python**

**Given weight and height of a person and we have to find the BMI (Body Mass Index) using Python.**

**Example:**

**Input:**

Height = 1.75

Weigth = 64

**Output:**

BMI is: 20.89 and you are: Healthy

The steps that we will follow are:

* We will first get input values from user using input() and convert it to float using float().
* We will use the BMI formula, which is weight/(height\*\*2).
* Then print the result using conditional statements.
* Here we have used elif because once we satisfy a condition we don’t want to check the rest of the statements.

## Program to calculate BMI in Python

# getting input from the user and assigning it to user

height = float(input("Enter height in meters: "))

weight = float(input("Enter weight in kg: "))

# the formula for calculating bmi

bmi = weight/(height\*\*2)

# \*\* is the power of operator i.e height\*height in this case

**print**("Your BMI is: {0} and you are: ".format(bmi), end='')

#conditions

**if** ( bmi < 16):

**print**("severely underweight")

**elif** ( bmi >= 16 **and** bmi < 18.5):

**print**("underweight")

**elif** ( bmi >= 18.5 **and** bmi < 25):

**print**("Healthy")

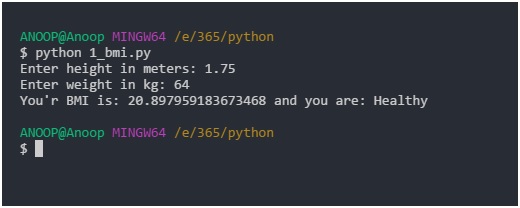
**elif** ( bmi >= 25 **and** bmi < 30):

**print**("overweight")

**elif** ( bmi >=30):

**print**("severely overweight")

**Output**



# **Python | Program to print Odd and Even numbers from the list of integers**

**Logic:** To do this, we will simply go through the list and check whether the number is divisible by 2 or not, if it is divisible by 2, then the number is EVEN otherwise it is ODD.

**Program:**

# Give number of elements present in list

n=int(input())

# list

l= list(map(int,input().strip().split(" ")))

# the number will be odd if on diving the number by 2

# its remainder is one otherwise number will be even

odd=[]

even=[]

**for** i **in** l:

**if**(i%2!=0):

odd.append(i)

**else**:

even.append(i)

**print**("list of odd number is:",odd)

**print**("list of even number is:",even)

**Output**

# Python program to print EVEN and ODD **Python | Program to print Palindrome numbers from the given list**

First few palindrome numbers are 0, 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 11 , 22 , 33 , 44 , 55 , 66 , 77 , 88 , 99 , 101 , 111 , 121 , ... and so on.

**Input format:** Given a number n, size of list then next line contains space separated n numbers.

**Logic:** We will simply convert the number into string and then using reversed(string) predefined function in python ,we will check whether the reversed string is same as the number or not.

**Program:**

# Give size of list

n=int(input())

# Give list of numbers having size n

l=list(map(int,input().strip().split(' ')))

**print**("Palindrome numbers are:")

# check through the list to check

# number is palindrome or not

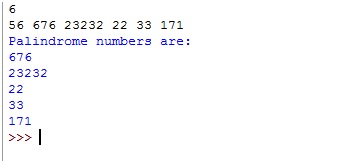
**for** i **in** l:

num=str(i)

**if**("".join(reversed(num))==num):

**print**(i)

**Output**



# **Python | Compute the net amount of a bank account based on the transactions**

**Given a few transactions (deposit, withdrawal), and we have to compute the Net Amount of that bank account based on these transactions in Python.**

**Example:**

Input:

Enter transactions: D 10000

Want to continue (Y for yes): Y

Enter transaction: W 5000

Want to continue (Y for yes): Y

Enter transaction: D 2000

Want to continue (Y for yes): Y

Enter transaction: W 100

Want to continue (Y for yes): N

Output:

Net amount: 6900

**Logic:**

* For this program, we will input transactions (till user’s choice is "Y") with type ("D" for deposit and "W" for withdrawal) and the amount – for that we will computer it in an infinite loop while True:
* Transactions will be like "D 2000" that means Deposit 2000, so there are two values needs to be extracted 1) "D" is the type of transaction and 2) 2000 is the amount to be deposited.
* A transaction input will be string type- convert/split the values to the list which is delimited by the space. For this, we use string.split() method.
* Now, the values will be in the list, the first value is the type – which is in string format and the second value is the amount – which is in also string format, we need to convert the amount in integer format. So we will convert its type (example code: amount = int(list[1])).
* After that, based on the transaction type check the conditions and adding/subtracting the amount from the net amount.
* After that, ask for the user for the next transaction and check the weather, if the user’s choice is not "Y" or "y", break the loop and print the net amount.

**Program:**

# computes net bank amount based on the input

# "D" for deposit, "W" for withdrawal

# define a variable for main amount

net\_amount = 0

**while** True:

# input the transaction

str = raw\_input ("Enter transaction: ")

# get the value type and amount to the list

# seprated by space

transaction = str.split(" ")

# get the value of transaction type and amount

# in the separated variables

type = transaction [0]

amount = int (transaction [1])

**if** type=="D" **or** type=="d":

net\_amount += amount

**elif** type=="W" **or** type=="w":

net\_amount -= amount

**else**:

**pass**

#input choice

str = raw\_input ("want to continue (Y for yes) : ")

**if** **not** (str[0] =="Y" **or** str[0] =="y") :

# break the loop

**break**

# print the net amount

**print** "Net amount: ", net\_amount

**Output**

Enter transaction: D 10000

want to continue (Y for yes) : Y

Enter transaction: W 5000

want to continue (Y for yes) : Y

Enter transaction: D 2000

want to continue (Y for yes) : Y

Enter transaction: W 100

want to continue (Y for yes) : N

Net amount: 6900

# **Python program to check whether a string contains a number or not**

**Given a string and we have to check whether it contains only digits or not in Python.**

To **check that a string contains only digits** (or a string has a number) – we can use **isdigit() function**, it returns true, if all characters of the string are digits.

**Syntax:**

string.isdigit()

**Example:**

Input:

str1 = "8789"

str2 = "Hello123"

str3 = "123Hello"

str4 = "123 456" #contains space

# function call

str1.isdigit()

str2.isdigit()

str3.isdigit()

str4.isdigit()

Output:

True

False

False

False

**Python code to check whether a strings contains a number or not**

# python program to check whether a string

# contains only digits or not

# variables declaration & initializations

str1 = "8789"

str2 = "Hello123"

str3 = "123Hello"

str4 = "123 456" #contains space

# checking

**print**("str1.isdigit(): ", str1.isdigit())

**print**("str2.isdigit(): ", str2.isdigit())

**print**("str3.isdigit(): ", str3.isdigit())

**print**("str4.isdigit(): ", str4.isdigit())

# checking & printing messages

**if** str1.isdigit():

**print**("str1 contains a number")

**else**:

**print**("str1 does not contain a number")

**if** str2.isdigit():

**print**("str2 contains a number")

**else**:

**print**("str2 does not contain a number")

**if** str3.isdigit():

**print**("str3 contains a number")

**else**:

**print**("str3 does not contain a number")

**if** str4.isdigit():

**print**("str4 contains a number")

**else**:

**print**("str4 does not contain a number")

**Output**

str1.isdigit(): True

str2.isdigit(): False

str3.isdigit(): False

str4.isdigit(): False

str1 contains a number

str2 does not contain a number

str3 does not contain a number

str4 does not contain a number

# **Python program for Zip, Zap and Zoom game**

Write a python program that displays a message as follows for a given number:

* If it is a multiple of three, display **"Zip"**.
* If it is a multiple of five, display **"Zap"**.
* If it is a multiple of both three and five, display **"Zoom"**.
* If it does not satisfy any of the above given conditions, display **"Invalid"**.

**Examples:**

Input:

Num = 9

Output : Zip

Input:

Num = 10

Output : Zap

Input:

Num = 15

Output : Zoom

Input:

Num = 19

Output: Invalid

**Code**

# Define a function for printing particular messages

**def** display(Num):

**if** Num % 3 == 0 **and** Num % 5 == 0 :

**print**("Zoom")

**elif** Num % 3 == 0 :

**print**("Zip")

**elif** Num % 5 == 0 :

**print**("Zap")

**else** :

**print**("Invalid")

# Main code

**if** \_\_name\_\_ == "\_\_main\_\_" :

Num = 9

# Function call

display(Num)

Num = 10

display(Num)

Num = 15

display(Num)

Num = 19

display(Num)

**Output**

Zip

Zap

Zoom

Invalid

# **Python program to convert temperature from Celsius to Fahrenheit and vice-versa**

**Formula used:**

**Celsius to Fahrenheit: °C= (5/9)\*(°F-32)**

**Fahrenheit to Celsius: °F= (9/5)\*(°C) + 32**

**Code:**

# Define a function to convert

# celsius temperature to Fahrenheit

**def** Celsius\_to\_Fahrenheit(c) :

f = (9/5)\*c + 32

**return** f

# Define a function to convert

# Fahrenheit temperature to Celsius

**def** Fahrenheit\_to\_Celsius(f) :

c = (5/9)\*(f - 32)

**return** c

# Driver Code

**if** \_\_name\_\_ == "\_\_main\_\_" :

c = 36

**print**(c, "degree celsius is equal to:",Celsius\_to\_Fahrenheit(c),"Fahrenheit")

f = 98

**print**(f,"Fahrenheit is equal to:",Fahrenheit\_to\_Celsius(f),"degree celsius")

**Output**

36 degree celsius is equal to: 96.8 Fahrenheit

98 Fahrenheit is equal to: 36.66666666666667 degree celsius

# **Python program for swapping the value of two integers**

So here is the code:

x = float(input('ENTER THE VALUE OF X: '))

y = float(input('ENTER THE VALUE OF Y: '))

temp = x

x = y

y = temp

**print**('X :', x,' Y :', y)

**Output:**

ENTER THE VALUE OF X: 10

ENTER THE VALUE OF Y: 20

X : 20.0 Y : 10.0

# **Python program for swapping the value of two integers without third variable**

One simple step, two different variables and it's all done.

So here is the code:

x = int(input("Enter the value of x :"))

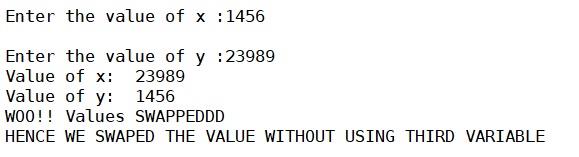
y = int(input("Enter the value of y :"))

(x,y) = (y,x)

**print**('Value of x: ', x, '\nValue of y: ', y, '\nWOO!! Values SWAPPEDDD')

**print**('HENCE WE SWAPED THE VALUE WITHOUT USING THIRD VARIABLE')

**Output:**



# **Python program to find winner of the day**

**Problem statement:**

There are two basketball teams (Team1 and Team2) in a school and they play some matches every day depending on their time and interest. Some days they play 3 matches, some days 2, some days 1, etc.

Write a python function, **find\_winner\_of\_the\_day()**, which accepts the name of the winner of each match and returns the name of the overall **winner of the day**. In case of the equal number of wins, return **"Tie"**.

**Example:**

Input : Team1 Team2 Team1

Output : Team1

Input : Team1 Team2 Team2 Team1 Team2

Output : Team2

**Code:**

# Python3 program to find winner of the day

# function which accepts the name of winner

# of each match of the day and return

# winner of the day

# This function accepts variable number of arguments in a tuple

**def** find\_winner\_of\_the\_day(\*match\_tuple):

team1\_count = 0

team2\_count = 0

# Iterating through all team name

# present in a match tuple variable

**for** team\_name **in** match\_tuple :

**if** team\_name == "Team1" :

team1\_count += 1

**else** :

team2\_count += 1

**if** team1\_count == team2\_count :

**return** "Tie"

**elif** team1\_count > team2\_count :

**return** "Team1"

**else** :

**return** "Team2"

# Driver Code

**if** \_\_name\_\_ == "\_\_main\_\_" :

**print**(find\_winner\_of\_the\_day("Team1","Team2","Team1"))

**print**(find\_winner\_of\_the\_day("Team1","Team2","Team1","Team2"))

**print**(find\_winner\_of\_the\_day("Team1","Team2","Team2","Team1","Team2"))

**Output**

Team1

Tie

Team2

# **Python program to convert Centimeter to Inches**

**ey:** 1 inch = 2.54 cms

**Example:**

Input:

Centimeter: 245

Output:

Inches: 96.45669291338582

**Python code to convert Centimeter to Inches**

# Python program to convert Centimeter to Inches

# taking input

num = float(input("Enter the distance measured in centimeter : "))

# converting from cms to inches

""" 1 inch = 2.54 centimeters"""

inc = num/2.54

# printing the result

**print**("Distance in inch : ", inc)

**Output**

First run:

Enter the distance measured in centimeter : 245

Distance in inch : 96.45669291338582

Second run:

Enter the distance measured in centimeter : 54

Distance in inch : 21.25984251968504

Third run:

Enter the distance measured in centimeter : 2.54

Distance in inch : 1.0

**Key:** 1 meter = 1.094 yards

**Example:**

Input:

Meters: 245

Output:

Yards: 268.03000000000003

**Python code to convert Meters to yards**

# Python program to convert Centimeter to Inches

# taking input

num = float(input("Enter the distance measured in centimeter : "))

# converting from cms to inches

""" 1 inch = 2.54 centimeters"""

inc = num/2.54

# printing the result

**print**("Distance in inch : ", inc)

**Output**

First run:

Enter the distance measured in meters : 245

Distance in yards : 268.03000000000003

Second run:

Enter the distance measured in meters : 54

Distance in yards : 59.07600000000001

Third run:

Enter the distance measured in meters : 100

Distance in yards : 109.4

**Key:** 1 meter = 1.094 yards

**Example:**

Input:

Yards: 54

Output:

Meters: 49.36014625228519

**Python code to convert Yards to meters**

# Python program to convert yards into meters

# input

num = float(input("Enter the distance measured in yards : "))

# converting from yards into meters

""" 1 meter = 1.094 yards"""

met = num/1.094

# printing the result

**print**("Distance in meters : ", met)

**Output**

First run:

Enter the distance measured in yards : 54

Distance in meters : 49.36014625228519

Second run:

Enter the distance measured in yards : 1.094

Distance in meters : 1.0

Third run:

Enter the distance measured in yards : 245

Distance in meters : 223.9488117001828

# **Python program to find perfect number**

Given an integer number and we have to check whether it is perfect number or not?

This Python program is used to find its all positive divisors excluding that number.

**Explanation:** For Example, 28 is a perfect number since divisors of 28 are 1, 2, 4,7,14 then sum of its divisor is 1 + 2 + 4 + 7 + 14 = 28.

**Note:** A perfect number is a positive integer that is equal to the sum of its proper positive divisors.

### **Python code to find perfect number**

**if** \_\_name\_\_ == "\_\_main\_\_" :

# initialisation

i = 2;sum = 1;

# take input from user and typecast into integer

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

# iterating till n//2 value

**while**(i <= n//2 ) :

# if proper divisor then add it.

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

sum += i

i += 1

# check sum equal to n or not

**if** sum == n :

**print**(n,"is a perfect number")

**else** :

**print**(n,"is not a perfect number")

**Output**

First run:

Enter a number: 28

28 is a perfect number

Second run:

Enter a number: 14

14 is not a perfect number

# **Python | Program to declare and print a list**

**Declare a list and print the elements/objects of the list in Python.**

**Program:**

# declaring list with integer elements

list1 = [10, 20, 30, 40, 50]

# printing list1

**print** "List element are: ", list1

# printing elements of list1 by index

**print** "Element @ 0 index:", list1[0]

**print** "Element @ 1 index:", list1[1]

**print** "Element @ 2 index:", list1[2]

**print** "Element @ 3 index:", list1[3]

**print** "Element @ 4 index:", list1[4]

# declaring list with string elements

list2 = ["New Delhi", "Mumbai", "Chennai", "calcutta"]

# printing list2

**print** "List elements are: ", list2

#printing elements of list2 by index

**print** "Element @ 0 index:",list2 [0]

**print** "Element @ 1 index:",list2 [1]

**print** "Element @ 2 index:",list2 [2]

**print** "Element @ 3 index:",list2 [3]

**print** " " # prints new line

# declaring list with mixed elements

list3 = ["Amit Shukla", 21, "New Delhi", 9876543210]

#printing list3

**print** "List elements are: ", list3

# printing elements of list3 by index

**print** "Element @ 0 index (Name) :", list3[0]

**print** "Element @ 1 index (Age ) :", list3[1]

**print** "Element @ 2 index (City) :", list3[2]

**print** "Element @ 3 index (Mob.) :", list3[3]

**print** "" # prints new line

**Output**

List element are: [10, 20, 30, 40, 50]

Element @ 0 index: 10

Element @ 1 index: 20

Element @ 2 index: 30

Element @ 3 index: 40

Element @ 4 index: 50

List elements are: ['New Delhi', 'Mumbai', 'Chennai', 'calcutta']

Element @ 0 index: New Delhi

Element @ 1 index: Mumbai

Element @ 2 index: Chennai

Element @ 3 index: calcutta

List elements are: ['Amit Shukla', 21, 'New Delhi', 9876543210]

Element @ 0 index (Name) : Amit Shukla

Element @ 1 index (Age ) : 21

Element @ 2 index (City) : New Delhi

Element @ 3 index (Mob.) : 9876543210

# **Python | Program for Adding, removing elements in the list**

## Python List append() Method

It is used to add/append an object (which will be passed in method as parameter) to the list.

**Syntax:**

list.append(element)

**Here,**

* list - is the name of the list.
* append() - is the method name, that is used to add element/object to the list.
* element - is an element (which is considered as on object or element) to be added in the list.

## Python List pop() Method

It is used to remove/pop an object from the list.

**Syntax:**

list.pop()

**Here,**

* list is the name of the list.
* pop() is the method name that is used to remove last element from the list.

**Program:**

# Declaring a list with integer and string elements

list = [10, 20, 30, "New Delhi", "Mumbai"]

# printing list

**print** "List elements are: ", list

# adding elements

list.append (40)

list.append (50)

list.append ("Chennai")

# printing list after adding elements

**print** "List elements: ", list

# removing elements

list.pop () ;

# printing list

**print** "List elements: ", list

# removing elements

list.pop () ;

# printing list

**print** "List elements: ", list

**Output**

List elements are: [10, 20, 30, 'New Delhi', 'Mumbai']

List elements: [10, 20, 30, 'New Delhi', 'Mumbai', 40, 50, 'Chennai']

List elements: [10, 20, 30, 'New Delhi', 'Mumbai', 40, 50]

List elements: [10, 20, 30, 'New Delhi', 'Mumbai', 40]

# **Python | Program to print a list using ‘FOR and IN’ loop**

**Given a list and we have to print its all elements using FOR and IN loop in Python.**

**FOR and IN** constructs as loop is very useful in the Python, it can be used to access/traverse each element of a list.

**Syntax of *for ... in* loop**

for *variable* in *list\_name*:

Statements

**Example/Programs: Declaring and printing a list**

# Declaring a list

list = [10, 20, 30, 40, 50]

# printing without using FOR and IN

**print** "List elements are: ", list

**print** " " # prints new line

# printing using FOR and IN

**print** "List elements are: "

**for** L **in** list:

**print** L

**print** " " # prints new line

# calculating Sum of all elements

sum = 0

**for** L **in** list:

sum += L

**print** "Sum is: ", sum

**Output**

List elements are: [10, 20, 30, 40, 50]

List elements are:

10

20

30

40

50

Sum is: 150

# **Python | Program to add an element at specified index in a list**

**Given a list and we have to add an element at specified index in Python.**

[list.appened() Method](https://www.includehelp.com/python/adding-removing-elements-in-the-list.aspx) is used to append/add an element at the end of the list. But, if we want to add an element at specified index, we use insert() method. It takes 2 arguments, *index* and *element*.

**Syntax:**

list.insert(index, element)

**Here,**

* list is the name of the list, in which we have to insert element at given index.
* index is the position, where we want to insert an element.
* element is an element/item to be inserted in the list.

**Example:**

list.insert(2, 100)

It will insert 100 at 2nd position in the list name ‘list’.

**Program:**

# Declaring a list

list = [10, 20, 30]

# printing elements

**print** (list)

# O/P will be: [10, 20, 30]

# inserting "ABC" at 1st index

list.insert (1, "ABC")

# printing

**print** (list)

# O/P will be: [10, 'ABC', 20, 30]

# inserting "PQR" at 3rd index

list.insert (3, "PQR")

# printing

**print** (list)

# O/P will be: [10, 'ABC', 20, 'PQR', 30]

# inserting 'XYZ' at 5th index

list.insert (5, "XYZ")

**print** (list)

# O/P will be: [10, 'ABC', 20, 'PQR', 30, 'XYZ']

# inserting 99 at second last index

list.insert (len (list) -1, 99)

# printing

**print** (list)

# O/P will be: [10, 'ABC', 20, 'PQR', 30, 99, 'XYZ']

**Output**

[10, 20, 30]

[10, 'ABC', 20, 30]

[10, 'ABC', 20, 'PQR', 30]

[10, 'ABC', 20, 'PQR', 30, 'XYZ']

[10, 'ABC', 20, 'PQR', 30, 99, 'XYZ']

# **Python | Program to remove first occurrence of a given element in the list**

**Python program to remove first occurrence of a given element in the list**: Given a list of the elements and we have to remove first occurrence of a given element.

## list.remove()

This method removes first occurrence given element from the list.

**Syntax:**

list.remove(element)

**Here,**

* list is the name of the list, from where we have to remove the element.
* element an element/object to be removed from the list.

**Example:**

Input :

list [10, 20, 30, 40, 30]

function call to remove 30 from the list:

list.remove(30)

Output:

List elements after removing 30

list [10, 20, 40, 30]

**Program:**

# Declaring a list

list = [10, 20, 30, 40, 30]

# print list

**print** "List element:"

**for** l **in** range(len(list)):

**print** (list[l])

# removing 30 from the list

list.remove(30);

# print list after removing 30

**print** "List element after removing 30:"

**for** l **in** range(len(list)):

**print** (list[l])

**Output**

List element:

10

20

30

40

30

List element after removing 30:

10

20

40

30

# **Python | Program to remove all elements in a range from the List**

**Given a list and we have to remove elements in a range from the list in Python.**

## del list(start\_index, end\_index)

del() method is used to remove all elements of list in range starting from *start\_index* to *end\_index*.

**Syntax:**

del list(start\_index, end\_index)

**Program:**

# Declaring a list

list = [10, 20, 30, 40, 50]

# print list

**print** "List element:"

**for** l **in** range(len(list)):

**print** (list[l])

# delete element from index 1 to 30del list[1.3]

**del** list[1:3]

# print list after deleting

# element from index 1 to 3

**print** "List element after del[1:3]:"

**for** l **in** range(len(list)):

**print** (list[l])

**Output**

List element:

10

20

30

40

50

List element after del[1:3]:

10

40

50

# **Python | Program to sort the elements of given list in Ascending and Descending Order**

**Given a list of the elements and we have to sort the list in Ascending and the Descending order in Python.**

Python list.sort() Method

sort() is a inbuilt method in Python, it is used to sort the elements/objects of the list in Ascending and Descending Order.

**Sorting elements in Ascending Order (list.sort())**

**Syntax:**

list.sort()

**Program to sort list elements in Ascending Order**

# List of integers

num = [10, 30, 40, 20, 50]

# sorting and printing

num.sort()

**print** (num)

# List of float numbers

fnum = [10.23, 10.12, 20.45, 11.00, 0.1]

# sorting and printing

fnum.sort()

**print** (fnum)

# List of strings

str = ["Banana", "Cat", "Apple", "Dog", "Fish"]

# sorting and printing

str.sort()

**print** (str)

**Output**

[10, 20, 30, 40, 50]

[0.1, 10.12, 10.23, 11.0, 20.45]

['Apple', 'Banana', 'Cat', 'Dog', 'Fish']

**Sorting in Descending Order (list.sort(reverse=True))**

To sort a list in descending order, we pass reverse=True as an argument with sort() method.

**Syntax:**

list.sort(reverse=True)

**Program to sort list elements in Descending Order**

# List of integers

num = [10, 30, 40, 20, 50]

# sorting and printing

num.sort(reverse=True)

**print** (num)

# List of float numbers

fnum = [10.23, 10.12, 20.45, 11.00, 0.1]

# sorting and printing

fnum.sort(reverse=True)

**print** (fnum)

# List of strings

str = ["Banana", "Cat", "Apple", "Dog", "Fish"]

# sorting and printing

str.sort(reverse=True)

**print** (str)

**Output**

[50, 40, 30, 20, 10]

[20.45, 11.0, 10.23, 10.12, 0.1]

['Fish', 'Dog', 'Cat', 'Banana', 'Apple']

# **Python | Program to find the differences of two lists**

**Given two lists of integers, we have to find the differences i.e. the elements which are not exists in second lists.**

**Example:**

Input:

List1 = [10, 20, 30, 40, 50]

List2 = [10, 20, 30, 60, 70]

Output:

Different elements:

[40, 50]

**Logic:**

To **find the differences of the lists**, we are using set() Method, in this way, we have to explicitly convert lists into sets and then subtract the set converted lists, the result will be the elements which are not exist in the second.

**Program to find difference of two lists in Python**

# list1 - first list of the integers

# lists2 - second list of the integers

list1 = [10, 20, 30, 40, 50]

list2 = [10, 20, 30, 60, 70]

# printing lists

**print** "list1:", list1

**print** "list2:", list2

# finding and printing differences of the lists

**print** "Difference elements:"

**print** (list (set(list1) - set (list2)))

**Output**

list1: [10, 20, 30, 40, 50]

list2: [10, 20, 30, 60, 70]

Difference elements:

[40, 50]

**Program 2: with mixed type of elements, printing 1) the elements which are not exist in list2 and 2) the elements which are not exists in list1.**

# list1 - first list with mixed type elements

# lists2 - second list with mixed type elements

list1 = ["Amit", "Shukla", 21, "New Delhi"]

list2 = ["Aman", "Shukla", 21, "Mumbai"]

# printing lists

**print** "list1:", list1

**print** "list2:", list2

# finding and printing differences of the lists

**print** "Elements not exists in list2:"

**print** (list (set(list1) - set (list2)))

**print** "Elements not exists in list1:"

**print** (list (set(list2) - set (list1)))

**Output**

list1: ['Amit', 'Shukla', 21, 'New Delhi']

list2: ['Aman', 'Shukla', 21, 'Mumbai']

Elements not exists in list2:

['Amit', 'New Delhi']

Elements not exists in list1:

['Aman', 'Mumbai']

# **Python | Program to Print the index of first matched element of a list**

**Given a list and we have to find the index of first matched of a list in Python.**

**Example:**

Input:

List = [10, 20, 10, 20, 30, 40, 50]

element = 10

Output:

Index of first matched 10 is: 0

Input:

List = [10, 20, 10, 20, 30, 40, 50]

element = 30

Output:

Index of first matched 20 is: 4

## list.index() Method

It’s an inbuilt method in Python, it returns the index of first matched element of a list.

**Syntax:**

list.index(element)

Here, *list* is the name of the list and *element* is the element/item whose first matched index to be returned.

**Program to find the index of first matched element in Python**

# declare a list of Integers

list = [10, 20, 10, 20, 30, 40, 50]

# printing index of 10

**print** (list.index (10))

#printing index of 20

**print** (list.index (20))

# printing index of 30

**print** (list.index (30))

# printing index of 40

**print** (list.index (40))

# printing index of 50

**print** (list.index (50))

**Output**

0

1

4

5

6

# **Python | Program to find the position of minimum and maximum elements of a list**

**Given a list and we have to find the index/position of minimum and maximum elements of a list in Python.**

**Prerequisite:**

* [Python | min() Method](https://www.includehelp.com/python/find-minimum-value-from-given-parameters-using-min.aspx)
* [Python | max() Method](https://www.includehelp.com/python/find-maximum-value-from-given-parameters-using-max.aspx)

**Example:**

Input:

list = [10, 1, 2, 20, 3, 20]

Output:

Positive of minimum element: 1

Positive of maximum element: 3

**Logic:**

To **find the positions/indexes of minimum and maximum elements of a list**, we need to find the maximum and minimum elements of the list – to find the maximum element of the list, we will use max(list) and to find the minimum element of the list, we will use min(list).

And, to get their indexes, we will use list.index(max(list)) and list.index(min(list)).

**Program to find the position of min and max elements of a list in Python**

# declare a list of Integers

list = [10, 1, 2, 20, 3, 20]

# min element's position/index

min = list.index (min(list))

# max element's position/index

max = list.index (max(list))

# printing the position/index of min and max elements

**print** "position of minimum element: ", min

**print** "position of maximum element: ", max

**Output**

position of minimum element: 1

position of maximum element: 3

**Explanation:**

* The minimum number of the list is 1 and it is at 1st position in the list. To get it’s index, we use list.index(min(list)) statement, min(list) returns 1 (as minimum element) and list.index(1) returns the index of 1 from the list. Hence, the position of minimum element is: 1
* The maximum number of the list if 20 and it is two times in the list, first occurrence of 20 is at 3rdposition and the second occurrence of 20 is at 5th position. Statement max(list) returns the maximum element of the list, which is 20 and the statement list.index(20) returns the [index/position of first matched element](https://www.includehelp.com/python/print-the-index-of-first-matched-element-of-a-list.aspx). Hence, the position of maximum element is: 3

# **Python | Program to input, append and print the list elements**

**ead the value of N (limit of the list), input N elements and print the elements in Python.**

**Example:**

Input:

Enter limit of the list: 5

Enter an integer: 10

Enter an integer: 20

Enter an integer: 30

Enter an integer: 40

Enter an integer: 50

Output:

Input list elements are:

10

20

30

40

50

**Program:**

# declare a list

list = []

# read limit (value of n)

# for maximum number of elements

n = int (input ("Enter limit of the list: "))

# input n integer element

# and append to the list

**for** i **in** range (n) :

item = int (input ("Enter an integer: "))

list.append (item)

# print all elements

**print** "Input list elements are: "

**for** i **in** range (n) :

**print** list [i]

**Output**

Enter limit of the list: 5

Enter an integer: 10

Enter an integer: 20

Enter an integer: 30

Enter an integer: 40

Enter an integer: 50

Input list elements are:

10

20

30

40

50

# **Python | Program to remove duplicate elements from the list**

**Example:**

Input:

list1: [10, 20, 10, 20, 30, 40, 30, 50]

Output:

List after removing duplicate elements

list2: [10, 20, 30, 40, 50]

**Logic:**

To implement the program is too easy, we have to append elements one by one to another list by checking whether element is available in the new list or not.

Let suppose, 20 is available three times in the list list1 and when we append 20 (first occurrence) to the list list2, it will be appended, but when we append 20 (second occurrence) to the list list2, condition will be false and item will not be appended. And finally, we will get list without duplicate elements.

**Program:**

# declare list

list1 = [10, 20, 10, 20, 30, 40, 30, 50]

# creating another list with unique elements

# declare another list

list2 = []

# appending elements

**for** n **in** list1:

**if** n **not** **in** list2:

list2.append(n)

# printing the lists

**print** "Original list"

**print** "list1: ", list1

**print** "List after removing duplicate elements"

**print** "list2: ", list2

**Output**

Original list

list1: [10, 20, 10, 20, 30, 40, 30, 50]

List after removing duplicate elements

list2: [10, 20, 30, 40, 50]

**Program (Defining User defines function):**

# Function to remove duplicates

**def** removeDuplicates (list1):

# declare another list

list2 = []

# appending elements

**for** n **in** list1:

**if** n **not** **in** list2:

list2.append (n)

**return** list2

# Main code

# declare a list

list1 = [10, 20, 10, 20, 30, 40, 30, 50]

# print the list

**print** "Original list: ", list1

**print** "List after duplicate remove: ", removeDuplicates (list1)

**Output**

Original list: [10, 20, 10, 20, 30, 40, 30, 50]

List after duplicate remove: [10, 20, 30, 40, 50]

# **Python | Program to print all numbers which are divisible by M and N in the List**

**Given a list of the integers and *M*, *N* and we have to print the numbers which are divisible by *M*, *N* in Python.**

**Example:**

Input:

List = [10, 15, 20, 25, 30]

M = 3, N=5

Output:

15

30

To find and print the list of the numbers which are divisible by M and N, we have to traverse all elements using a loop, and check whether the number is divisible by M and N, if number is divisible by M and N both print the number.

**Program:**

# declare a list of integers

list = [10, 15, 20, 25, 30]

# declare and assign M and N

M = 3

N = 5

# print the list

**print** "List is: ", list

# Traverse each element and check

# whether it is divisible by M, N

# or not, if condition is true print

# the element

**print** "Numbers divisible by {0} and {1}".format (M, N)

**for** num **in** list:

**if**( num%M==0 **and** num%N==0 ) :

**print** num

**Output**

List is: [10, 15, 20, 25, 30]

Numbers divisible by 3 and 5

15

30

# **Python | Create three lists of numbers, their squares and cubes**

**Take a range i.e. start and end, and we have to create three lists, list1 should contains numbers, list2 should contain squares of the numbers and list3 should contain cubes of the numbers in Python.**

**Example:**

Input:

Start = 1

End = 10

Output:

numbers: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

squares: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

cubes : [1, 8, 27, 64, 125, 216, 343, 512, 729, 1000]

**Logic:**

* Declare three lists.
* Define range, here we are defining start with 1 and end with 10.
* Run a loop with the range as range(start, end+1) and loop counter as count.
* Append the loop counter count to the list named numbers, append square to the list named squares and append the cube to the list named cubes.
* Finally, print the lists.

**Program:**

# declare lists

numbers = []

squares = []

cubes = []

# start and end numbers

start = 1

end = 10

# run a loop from start to end+1

**for** count **in** range (start, end+1) :

numbers.append (count)

squares.append (count\*\*2)

cubes.append (count\*\*3)

# print the lists

**print** "numbers: ",numbers

**print** "squares: ",squares

**print** "cubes : ",cubes

**Output**

numbers: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

squares: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

cubes : [1, 8, 27, 64, 125, 216, 343, 512, 729, 1000]

### **By defining own functions**

# define function to add numbers in list

**def** listNumbers(a,b):

#define dynamic list

list = []

**for** count **in** range(a,b+1):

list.append(count)

#return list

**return** list

# define function to add squares in list

**def** listSquares(a,b):

#define dynamic list

list = []

**for** count **in** range(a,b+1):

list.append(count\*\*2)

#return list

**return** list

# define function to add cubes in list

**def** listCubes(a,b):

#define dynamic list

list = []

**for** count **in** range(a,b+1):

list.append(count\*\*3)

#return list

**return** list

# Main code

# declare lists

numbers = []

squares = []

cubes = []

# start and end numbers

start = 1

end = 10

# get values in lists

numbers = listNumbers(start, end)

squares = listSquares(start, end)

cubes = listCubes(start, end)

# print the lists

**print** "numbers: ",numbers

**print** "squares: ",squares

**print** "cubes : ",cubes

**Output**

numbers: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

squares: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

cubes : [1, 8, 27, 64, 125, 216, 343, 512, 729, 1000]

# **Python | Count vowels in a string**

**Given a string, and we have to count the total number of vowels in the string using python program.**

**Example:**

Input:

Str = "Hello world"

Output:

Total vowels are: 3

**Program:**

# count vowels in a string

# declare, assign string

str = "Hello world"

# declare count

count = 0

# iterate and check each character

**for** i **in** str:

# check the conditions for vowels

**if**( i=='A' **or** i=='a' **or** i=='E' **or** i=='e'

**or** i=='I' **or** i=='i' **or** i=='O' **or** i=='o'

**or** i=='U' **or** i=='u'):

count +=1;

# print count

**print** "Total vowels are: ", count

**Output**

Total vowels are: 3

**Implement the program by creating functions to check vowel and to count vowels:**

Here, we are creating two functions:

**1) isVowel()**

This function will take character as an argument, and returns True, if character is vowel.

**2) countVowels()**

This function will take string as an argument, and return total number of vowels of the string.

**Program:**

# count vowels in a string

# function to check character

# is vowel or not

**def** isVowel(ch):

# check the conditions for vowels

**if**(ch=='A' **or** ch=='a' **or** ch=='E' **or** ch=='e'

**or** ch=='I' **or** ch=='i' **or** ch=='O' **or** ch=='o'

**or** ch=='U' **or** ch=='u'):

**return** True

**else**:

**return** False

# function to return total number of vowels

**def** countVowel(s) :

# declare count

count =0

# iterate and check characters

**for** i **in** str:

**if**(isVowel(i) == True):

count += 1

**return** count

# Main code

# declare, assign string

str = "Hello world"

# print count

**print** "Total vowels are: ", countVowel(str)

**Output**

Total vowels are: 3

Python Program to check if a Number is Positive, Negative or Zero

**We can use a Python program to distinguish that if a number is positive, negative or zero.**

**Positive Numbers:** A number is known as a positive number if it has a greater value than zero. **i.e.** 1, 2, 3, 4 etc.

**Negative Numbers:** A number is known as a negative number if it has a lesser value than zero. i.e. -1, -2, -3, -4 etc.

Python Condition And Loops1

**See this example:**

1. num = float(input("Enter a number: "))
3. **if** num > 0:
4. **print**("{0} is a positive number".format(num))
5. **elif** num == 0:
6. **print**("{0} is zero".format(num))
7. **else**:
8. **print**("{0} is negative number".format(num))

# **Python Program to Check if a Number is Odd or Even**

**Odd and Even numbers:**

If you divide a number by 2 and it gives a remainder of 0 then it is known as even number, otherwise an odd number.

**Even number examples:** 2, 4, 6, 8, 10, etc.

**Odd number examples:**1, 3, 5, 7, 9 etc.

**See this example:**

1. num = int(input("Enter a number: "))
2. **if** (num % 2) == 0:
3. **print**("{0} is Even number".format(num))
4. **else**:
5. **print**("{0} is Odd number".format(num))

# **Python Program to Check Leap Year**

**Leap Year:**

A year is called a leap year if it contains an additional day which makes the number of the days in that year is 366. This additional day is added in February which makes it 29 days long.

A leap year occurred once every 4 years.

**How to determine if a year is a leap year?**

You should follow the following steps to determine whether a year is a leap year or not.

1. If a year is evenly divisible by 4 means having no remainder then go to next step. If it is not divisible by 4. It is not a leap year. For example: 1997 is not a leap year.
2. If a year is divisible by 4, but not by 100. For example: 2012, it is a leap year. If a year is divisible by both 4 and 100, go to next step.
3. If a year is divisible by 100, but not by 400. For example: 1900, then it is not a leap year. If a year is divisible by both, then it is a leap year. So 2000 is a leap year.

**See this example:**

1. year = int(input("Enter a year: "))
2. **if** (year % 4) == 0:
3. **if** (year % 100) == 0:
4. **if** (year % 400) == 0:
5. **print**("{0} is a leap year".format(year))
6. **else**:
7. **print**("{0} is not a leap year".format(year))
8. **else**:
9. **print**("{0} is a leap year".format(year))
10. **else**:
11. **print**("{0} is not a leap year".format(year))

# **Python Program to Check Prime Number**

**Prime numbers:**

A prime number is a natural number greater than 1 and having no positive divisor other than 1 and itself.

For example: 3, 7, 11 etc are prime numbers.

**Composite number:**

Other natural numbers that are not prime numbers are called composite numbers.

For example: 4, 6, 9 etc. are composite numbers.

**See this example:**

1. num = int(input("Enter a number: "))
3. **if** num > 1:
4. **for** i **in** range(2,num):
5. **if** (num % i) == 0:
6. **print**(num,"is not a prime number")
7. **print**(i,"times",num//i,"is",num)
8. **break**
9. **else**:
10. **print**(num,"is a prime number")
12. **else**:
13. **print**(num,"is not a prime number")

# **Python Program to Print all Prime Numbers between an Interval**

We have already read the concept of prime numbers in the previous program. Here, we are going to print the prime numbers between given interval.

**See this example:**

1. #Take the input from the user:
2. lower = int(input("Enter lower range: "))
3. upper = int(input("Enter upper range: "))
5. **for** num **in** range(lower,upper + 1):
6. **if** num > 1:
7. **for** i **in** range(2,num):
8. **if** (num % i) == 0:
9. **break**
10. **else**:
11. **print**(num)

# **Python Program to Find the Factorial of a Number**

**What is factorial?**

Factorial is a non-negative integer. It is the product of all positive integers less than or equal to that number for which you ask for factorial. It is denoted by exclamation sign (!).

For example:

1. 4! = 4x3x2x1 = 24

The factorial value of 4 is 24.

**Note:** The factorial value of 0 is 1 always. (Rule violation)

**See this example:**

1. num = int(input("Enter a number: "))
2. factorial = 1
3. **if** num < 0:
4. **print**("Sorry, factorial does not exist for negative numbers")
5. **elif** num == 0:
6. **print**("The factorial of 0 is 1")
7. **else**:
8. **for** i **in** range(1,num + 1):
9. factorial = factorial\*i
10. **print**("The factorial of",num,"is",factorial)

# **Python Program to Display the multiplication Table**

In Python, you can make a program to display the multiplication table of any number. The following program displays the multiplication table (from 1 to 10) according to the user input.

**See this example:**

1. num = int(input("Show the multiplication table of? "))
2. # using for loop to iterate multiplication 10 times
3. **for** i **in** range(1,11):
4. **print**(num,'x',i,'=',num\*i)

# **Python Program to Print the Fibonacci sequence**

**Fibonacci sequence:**

The Fibonacci sequence specifies a series of numbers where the next number is found by adding up the two numbers just before it.

**For example:**

2. 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, **and** so on....

**See this example:**

1. nterms = int(input("How many terms you want? "))
2. # first two terms
3. n1 = 0
4. n2 = 1
5. count = 2
6. # check if the number of terms is valid
7. **if** nterms <= 0:
8. **print**("Plese enter a positive integer")
9. **elif** nterms == 1:
10. **print**("Fibonacci sequence:")
11. **print**(n1)
12. **else**:
13. **print**("Fibonacci sequence:")
14. **print**(n1,",",n2,end=', ')
15. **while** count < nterms:
16. nth = n1 + n2
17. **print**(nth,end=' , ')
18. # update values
19. n1 = n2
20. n2 = nth
21. count += 1

# **Python Program to Find the Sum of Natural Numbers**

**Natural numbers:**

As the name specifies, a natural number is the number that occurs commonly and obviously in the nature. It is a whole, non-negative number.

Some mathematicians think that a natural number must contain 0 and some don't believe this theory. So, a list of natural number can be defined as:

2. N= {0, 1, 2, 3, 4, .... **and** so on}
3. N= {1, 2, 3, 4, .... **and** so on}

**See this example:**

1. num = int(input("Enter a number: "))
3. **if** num < 0:
4. **print**("Enter a positive number")
5. **else**:
6. sum = 0
7. # use while loop to iterate un till zero
8. **while**(num > 0):
9. sum += num
10. num -= 1
11. **print**("The sum is",sum)

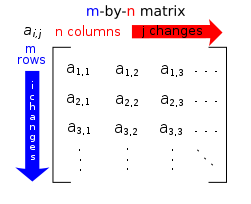
Python Program to Add Two Matrices

**What is Matrix?**

In mathematics, matrix is a rectangular array of numbers, symbols or expressions arranged in the form of rows and columns. For example: if you take a matrix A which is a 2x3 matrix then it can be shown like this:

1. 2       3          5
2. 8       12        7

**Image representation:**



In Python, matrices can be implemented as nested list. Each element of the matrix is treated as a row. For example X = [[1, 2], [3, 4], [5, 6]] would represent a 3x2 matrix. First row can be selected as X[0] and the element in first row, first column can be selected as X[0][0].

**Let's take two matrices X and Y, having the following value:**

1. X = [[1,2,3],
2. [4,5,6],
3. [7,8,9]]
5. Y = [[10,11,12],
6. [13,14,15],
7. [16,17,18]]

Create a new matrix result by adding them.

**See this example:**

1. X = [[1,2,3],
2. [4,5,6],
3. [7,8,9]]
5. Y = [[10,11,12],
6. [13,14,15],
7. [16,17,18]]
9. Result = [[0,0,0],
10. [0,0,0],
11. [0,0,0]]
12. # iterate through rows
13. **for** i **in** range(len(X)):
14. # iterate through columns
15. **for** j **in** range(len(X[0])):
16. result[i][j] = X[i][j] + Y[i][j]
17. **for** r **in** result:
18. **print**(r)

# **Python Program to Find Factorial of Number Using Recursion**

**Factorial:** Factorial of a number specifies a product of all integers from 1 to that number. It is defined by the symbol explanation mark (!).

For example: The factorial of 5 is denoted as 5! = 1\*2\*3\*4\*5 = 120.

**See this example:**

1. **def** recur\_factorial(n):
2. **if** n == 1:
3. **return** n
4. **else**:
5. **return** n\*recur\_factorial(n-1)
6. # take input from the user
7. num = int(input("Enter a number: "))
8. # check is the number is negative
9. **if** num < 0:
10. **print**("Sorry, factorial does not exist for negative numbers")
11. **elif** num == 0:
12. **print**("The factorial of 0 is 1")
13. **else**:
14. **print**("The factorial of",num,"is",recur\_factorial(num))

# **Python Program to Display Fibonacci Sequence Using Recursion**

**Fibonacci sequence:**

A Fibonacci sequence is a sequence of integers which first two terms are 0 and 1 and all other terms of the sequence are obtained by adding their preceding two numbers.

For example: 0, 1, 1, 2, 3, 5, 8, 13 and so on...

**See this example:**

1. **def** recur\_fibo(n):
2. **if** n <= 1:
3. **return** n
4. **else**:
5. **return**(recur\_fibo(n-1) + recur\_fibo(n-2))
6. # take input from the user
7. nterms = int(input("How many terms? "))
8. # check if the number of terms is valid
9. **if** nterms <= 0:
10. **print**("Plese enter a positive integer")
11. **else**:
12. **print**("Fibonacci sequence:")
13. **for** i **in** range(nterms):
14. **print**(recur\_fibo(i))