INPUT()

The input() method reads a line from input, converts into a string and returns it.

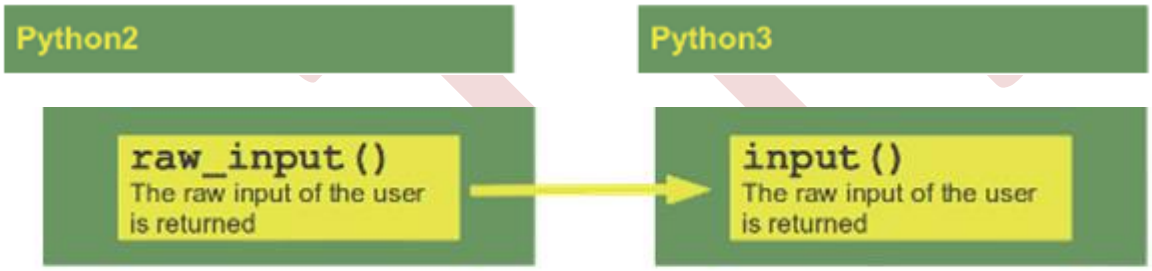
The syntax of input() method is:

input([prompt])

input() Parameters

The input() method takes a single optional argument:

prompt (Optional) - a string that is written to standard output (usually screen) without trailing newline



Return value from input()

The input() method reads a line from input (usually user),
converts the line into a string by removing the trailing newline and returns it.

EG=> static condition

```
# static condition

shirt = 500

oil = 100

pant = 700


print('Total Amount is ',(shirt+oil+pant))
```

EG=> dynamic

```
# dynamic

print('Enter user shirt cost : ')
shirt = int(input())
print('Enter user oil cost : ')
oil = int(input())
print('Enter user pant cost : ')
pant = int(input())

print('Total Amount is ',(shirt+oil+pant))


print('-----------------------------------------------------------------')

shirt = int(input('Enter user shirt cost : '))
oil = int(input('Enter user oil cost : '))
pant = int(input('Enter user pant cost : '))
gst = float(input('Enter G.S.T  :  '))

gstbill =(shirt+oil+pant)*gst
print()
print('Gst Amount : ',gstbill)
print()
print('Amount without Gst : ',(shirt+oil+pant))
print()
print('Total Amount is ',(shirt+oil+pant+gstbill))
```

ID () Function

The function accepts a single parameter and is used to **return** the identity of an object.
This identity must be unique and constant for this object during the lifetime.
Two objects with non-overlapping lifetimes may have the same id() **value**.

Return Value from id()

The id() function returns identity of the object. This is an integer which is unique for the given object and remains constant during its lifetime.

EG =>

```
str1 = "a"
print(id(str1))

str2 = "a"
print(id(str2))

# This will return True
print(id(str1) == id(str2))

# Use in Lists
list1 = ["a", "b", "c"]
print(id(list1[0]))
print(id(list1[2]))

# This returns false
print(id(list1[0]) == id(list1[2]))
```

Python Data Types

Python Data Types are used to define the type of a variable.

Data Types in Python

- Python Numbers
- Python List
- Python Tuple
- Python Strings
- Python Set
- Python Dictionary
- Conversion between data types

Python Numbers

Integers, floating point numbers and complex numbers fall under Python numbers category. They are defined as `int`, `float` and `complex` class in Python.

We can use the `type ()` function to know which class a variable or a value belongs to and the `isinstance ()` function to check if an object belongs to a class.

Eg=>

```
a = 5
print (a, "is of type", type(a))
a = 2.0
print (a, "is of type", type(a))
a = 1+2j
print (a, "is complex number?", isinstance (1+2j, complex))
```

Integers can be of any length; it is only limited by the memory available.

int 16 bytes

A floating-point number is accurate up to 15 decimal places. Integer and floating points are separated by decimal points. `1` is integer, `1.0` is floating point number.

Complex numbers are written in the form, `x + yj`, where `x` is the real part and `y` is the imaginary part.

Python List

List is an ordered sequence of items. It is one of the most used datatypes in Python and is very flexible. All the items in a list do not need to be of the same type.

Python Tuple

Tuple is an ordered sequence of items same as list. The only difference is that tuples are immutable. Tuples once created cannot be modified.

Python Strings

String is sequence of Unicode characters. We can use single quotes or double quotes to represent strings. Multi-line strings can be denoted using triple quotes, ''' or """.

Python Set

Set is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

Python Dictionary

Dictionary is an unordered collection of key-value pairs.

It is generally used when we have a huge amount of data. Dictionaries are optimized for retrieving data. We must know the key to retrieve the value.

Conversion between data types

We can convert between different data types by using different type conversion functions like int(), float(), str() etc