Accepting Input and Displaying Output

- Learn how to **take input from a user** and system In Python.
- Accept an integer, float, character, and string input from a user.
- Convert the user input to a different data type.
- Learn fancier **output formatting**.

Python Input function to accept input from a user

In Python, we have the following two functions to handle input from a user and system.

1. input(prompt) to accept input from a user.
2. print() to display output on the console.

Python 3 has a built-in function input() **to accept user input**.

In Python 2, to accept user input we can use the following two functions: –

1. input([prompt])
2. raw_input([prompt])

The input() function reads a line entered on a console by an input device such as a keyboard and convert it into a string and returns it. As a new developer, It is essential to understand what is input in Python.

What is the input?

The Input is nothing but some value from a system or user. For example, if you want to perform an addition of two numbers on the calculator you need to provide two number to the calculator, those two number is nothing but an input provided by the user to a calculator program.

There are different types of Input, and that comes in various ways. For example: –

- **Input stems from the keyboard**. i.e., the user entered some value using a keyboard.
- **Input Using Mouse Click or movement**, i.e. you clicked on the radio button or some drop-down list and chosen an option from it.

In Python, there are various ways for reading input from the user from the command line environment or through the user interface. In both cases, the user is sending input from Keyboard or mouse.

Python example to accept input from a user

Let see how to accept employee data from a user using the `input()` function and display it using the `print()` function.

```
name = input("Enter Employee Name ")
salary = input("Enter salary ")
company = input("Enter Company name ")

print("\n")
print("Printing Employee Details")
print("Name", "Salary", "Company")
print(name, salary, company)
```

Output:

```
Enter Employee Name Jon

Enter salary 12000

Enter Company name Google
```

```
Printing Employee Details

Name Salary Company
```

Jon 12000 Google

How input() function works in Python

Python input() function syntax

```
input([prompt])
```

Here the **prompt** argument is optional. The prompt argument is used to display a message to the user. For example, the prompt is, "Please enter a value." When input() function executes, program flow stops until a user enters some value.

Whatever you enter as input, the input() function converts it into a string. If you enter an integer value, still it will convert it into a string. If you want to number input from a user, you need to perform **type conversion** on the input value.

Let's understand this with an example.

Program to check input type in Python

```
number = input("Enter number ")
name = input("Enter name ")

print("\n")
print("Printing type of a input value")
print("type of number", type(number))
print("type of name", type(name))
```

Output:

```
Enter number 26

Enter name Jessa



Printing type of a input value
```

type of number <class 'str'>

type of name <class 'str'>

As you know whatever you enter as input, the input() function always converts it into a string.

Accept an Integer input from User

Let's see how to accept an integer value from a user in Python. We need to convert an input string value into an integer using a int() function.

```
# program to calculate addition of two input numbers

first_number = int(input("Enter first number "))
second_number = int(input("Enter second number "))

print("\n")
print("First Number:", first_number)
print("Second Number:", second_number)
sum1 = first_number + second_number
print("Addition of two number is: ", sum1)
```

Output:

```
Enter first number 20
```

```
Enter second number 40



First Number: 20

Second Number: 40

Addition of two number is:  60
```

Now if you print the type of first_number you should get integer type. type(first_number) will return <class 'int'>

Accept float input from User

Let's see how to accept float value from a user in Python. You need to convert user input to the float number using the float() function as we did for the integer value.

```
float_number = float(input("Enter float number "))
print("\n")
print("input float number is: ", float_number)
print("type is:", type(float_number))
```

Output:

```
Enter float number 29.5

input float number is:  29.5

type is: <class 'float'>
```

Accept List as an Input from a user in Python

Like string and integer, we can also accept list and array as an input from a user. For example, accept a list of numbers from the user.

Get multiple input values from a user in one line

In Python, It is possible to get multiple values from the user in one line. i.e., In Python, we can accept two or three values from the user in one input() call.

For example, in a single execution of the input() function, we can ask the user hi/her name, age, and phone number and store it in three different variables. Let' see how to do this.

```
name, age, phone = input("Enter your name, Age, Percentage separated by space ").split()
print("\n")
print("User Details: ", name, age, phone)
```

Output:

```
Enter your name, Age, Percentage separated by space Jessa 26 75.50


User Details:  Jessa 26 75.50
```

Accept multiline input from a user

As you know, the input() function does not allow the user to enter put lines separated by a newline. If the user tries to enter multiline input, it prints back only the first line. Let's see how to gets multiple line input.

As you know, the **input()** function does not allow the user to enter put lines separated by a newline. If the user tries to enter multiline input, it prints back only the first line. Let's see how to gets multiple line input.

We can use for loop. In each iteration, we can get input string from the user and then .join() them using \n, or you can also concatenate each input string using + operator separated by \n.

```
MultiLine = []
print("Tell me about yourself")
while True:
    line = input()
    if line:
        MultiLine.append(line)
    else:
        break
finalText = '\n'.join(MultiLine)
print("\n")
print("Final text input")
print(finalText)
```

Output:

```
Tell me about yourself

My Name is Jessa

I am a software engineer
```

Format output strings by its positions

```
firstName = input("Enter First Name ")
lastName = input("Enter Last Name ")
organization = input("Enter Organization Name ")
```

```
print("\n")
print('{0}, {1} works at {2}'.format(firstName, lastName,
organization))
print('{1}, {0} works at {2}'.format(firstName, lastName,
organization))
print('FirstName {0}, LastName {1} works at
{2}'.format(firstName, lastName, organization))
print('{0}, {1} {0}, {1} works at {2}'.format(firstName,
lastName, organization))
```

Output:

```
Enter First Name Ault

Enter Last Name Kelly

Enter Organization Name Google

Ault, Kelly works at Google

Kelly, Ault works at Google

FirstName Ault, LastName Kelly works at Google

Ault, Kelly Ault, Kelly works at Google
```

Accessing output string arguments by name

```
name = input("Enter Name ")
marks = input("Enter marks ")

print("\n")
print('Student: Name:  {firstName}, Marks:
{percentage}%'.format(firstName=name, percentage=marks))
```

Output:

```
Enter Name Jhon

Enter marks 74
```

Student: Name: Jhon, Marks: 74%

Output text alignment specifying a width

```
text = input("Enter text ")

print("\n")
# left aligned
print('{:<25}'.format(text)) # Right aligned
print('{:>25}'.format(text))
# centered
print('{:^25}'.format(text))
```

Output:

```
Enter text This is a sample text



This is a sample text

    This is a sample text

  This is a sample text
```

Specifying a sign while displaying output numbers

```
positive_number = float(input("Enter Positive Number "))
negative_number = float(input("Enter Negative Number "))

print("\n")
# sign '+' is for both positive and negative number
print('{:+f}; {:+f}'.format(positive_number, negative_number))

# sign '-' is only for negative number
print('{:f}; {:-f}'.format(positive_number, negative_number))
```

Output:

```
Enter Positive Number 25.25

Enter Negative Number -15.50

+25.250000; -15.500000
```

25.250000; -15.500000

Python Input() vs raw_input()

input() function works differently between Python 3 and Python 2. Both the input() and raw_input() function accept input from user.

The difference between the input() and raw_input() functions relevant only when you are using python 2.

In Python 2

- The main difference between those two functions is input() function automatically converts user input to appropriate type. i.e., If a user-entered string input() function converts it into a string, and if a user entered a number it converts to an integer.

- **raw_input()** convert every user input to string.

raw_input() function of python 2 is renamed to input() in Python 3.x and original input() function is removed from Python 3.

Let's see how to use python raw_input() in Python 2.

Python 2's raw_input() function to accept input from a user

If you are using python 2, then we need to use raw_input() instead of input() function.

```
string = raw_input("Enter a String ")
print "input string is: ", string
```

Output:

```
Enter a String Python is a beautiful language

input string is: Python is a beautiful language
```

```
name = input("Enter Employee Name ")
salary = input("Enter salary ")
company = input("Enter Company name ")

print("\n")
print("Printing Employee Details")
print("Name", "Salary", "Company")
print(name, salary, company)
```

Output:

```
Enter Employee Name Jon

Enter salary 12000

Enter Company name Google
```

```
Printing Employee Details

Name Salary Company
```

Jon 12000 Google

How input() function works in Python

Python input() function syntax

```
input([prompt])
```

Here the **prompt** argument is optional. The prompt argument is used to display a message to the user. For example, the prompt is, “Please enter a value.” When input() function executes, program flow stops until a user enters some value.

Whatever you enter as input, the input() function converts it into a string. If you enter an integer value, still it will convert it into a string. If you want to number input from a user, you need to perform **type conversion** on the input value.

Let’s understand this with an example.

Program to check input type in Python

```
number = input("Enter number ")
name = input("Enter name ")

print("\n")
print("Printing type of a input value")
print("type of number", type(number))
print("type of name", type(name))
```

Output:

```
Enter number 26

Enter name Jessa


Printing type of a input value
```

type of number <class 'str'>

type of name <class 'str'>

Accept an Integer input from User

Let's see how to accept an integer value from a user in Python. We need to convert an input string value into an integer using a int() function.

```
# program to calculate addition of two input numbers

first_number = int(input("Enter first number "))
second_number = int(input("Enter second number "))

print("\n")
print("First Number:", first_number)
print("Second Number:", second_number)
sum1 = first_number + second_number
print("Addition of two number is: ", sum1)
```

Output:

```
Enter first number 20

Enter second number 40
```

```
First Number: 20
Second Number: 40
```

Addition of two number is: 60

Accept float input from User

Let's see how to accept float value from a user in Python. You need to convert user input to the float number using the float() function as we did for the integer value.

```
float_number = float(input("Enter float number "))
print("\n")
print("input float number is: ", float_number)
print("type is:", type(float_number))
```

Output:

```
Enter float number 29.5
input float number is:  29.5
```

type is: <class 'float'>

Get multiple input values from a user in one line

In Python, It is possible to get multiple values from the user in one line. i.e., In Python, we can accept two or three values from the user in one input() call.

For example, in a single execution of the input() function, we can ask the user hi/her name, age, and phone number and store it in three different variables. Let' see how to do this.

```
name, age, phone = input("Enter your name, Age, Percentage separated by space ").split()
print("\n")
print("User Details: ", name, age, phone)
```

Output:

```
Enter your name, Age, Percentage separated by space Jessa 26 75.50


User Details:  Jessa 26 75.50
```

Flow Control Statements

An else statement can be combined with an if statement.

An else statement contains the block of code that executes if the conditional expression in the if statement resolves to 0 or a FALSE value.

The else statement is an optional statement and there could be at most only one else statement following if .

Syntax The syntax of the if...else statement is −

```
if expression:

statement(s)

else:

statement(s)
```

Ex:-

```
var1 = 100
 if var1:
        print "1 - Got a true expression value"
print var1
else:
         print "1 - Got a false expression value"
print var1
```

The elif Statement

The elif statement allows you to check multiple expressions for TRUE and execute a block of code as soon as one of the conditions evaluates to TRUE.

Similar to the else, the elif statement is optional. However, unlike else, for which there can be at most one statement, there can be an arbitrary number of elif statements following an if.

syntax

```
if expression1:
 statement(s)
 elif expression2:
 statement(s)
 elif expression3:
 statement(s)
 else: statement(s)
```

Ex:

```
var = 5
```

```
if var == 10:
   print "1 - Got a true expression value"
   print var
elif var == 15:
   print "2 - Got a true expression value"
   print var
elif var == 20:
   print "3 - Got a true expression value"
   print var
else:
   print "4 - Got a false expression value"
   print var

print "Good bye!"
```