**Standard Input and Output Python 3**

# The input () function:

- The purpose of **input()** function is to read input from the standard input (the keyboard, by default).

It accepts all user input as a string.

- The user may enter a number or a string but the **input()** function treats them as strings only.

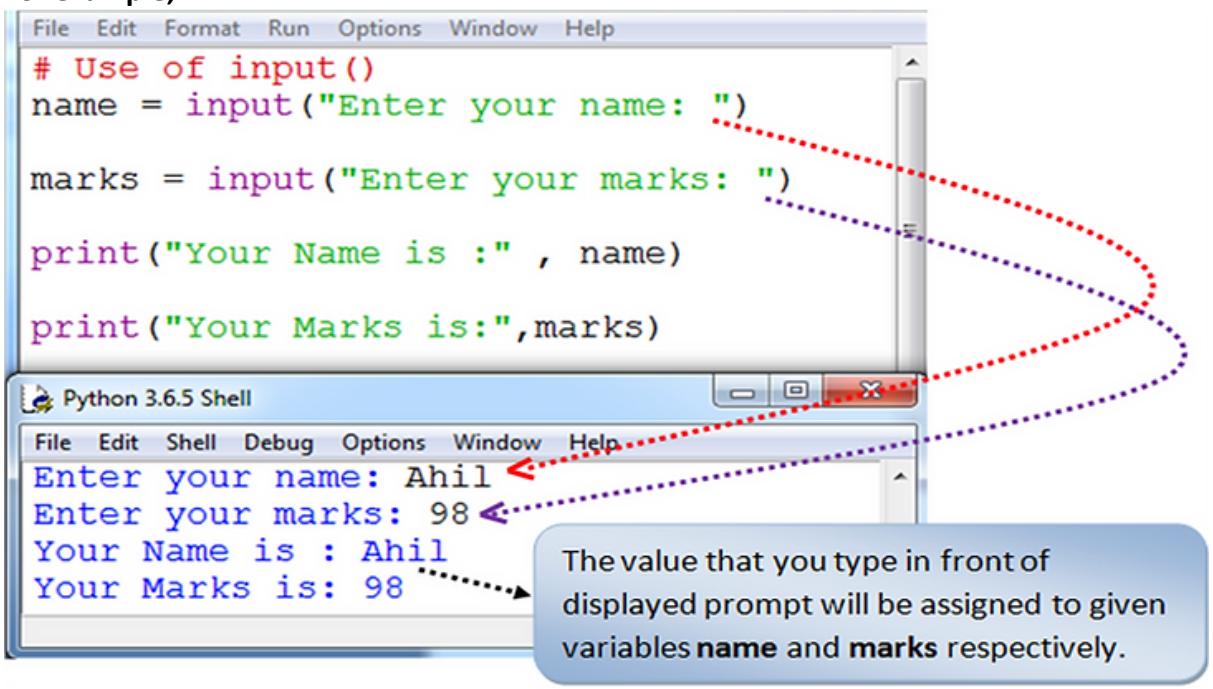
The general format is:

***input([prompt])***

**Here,**

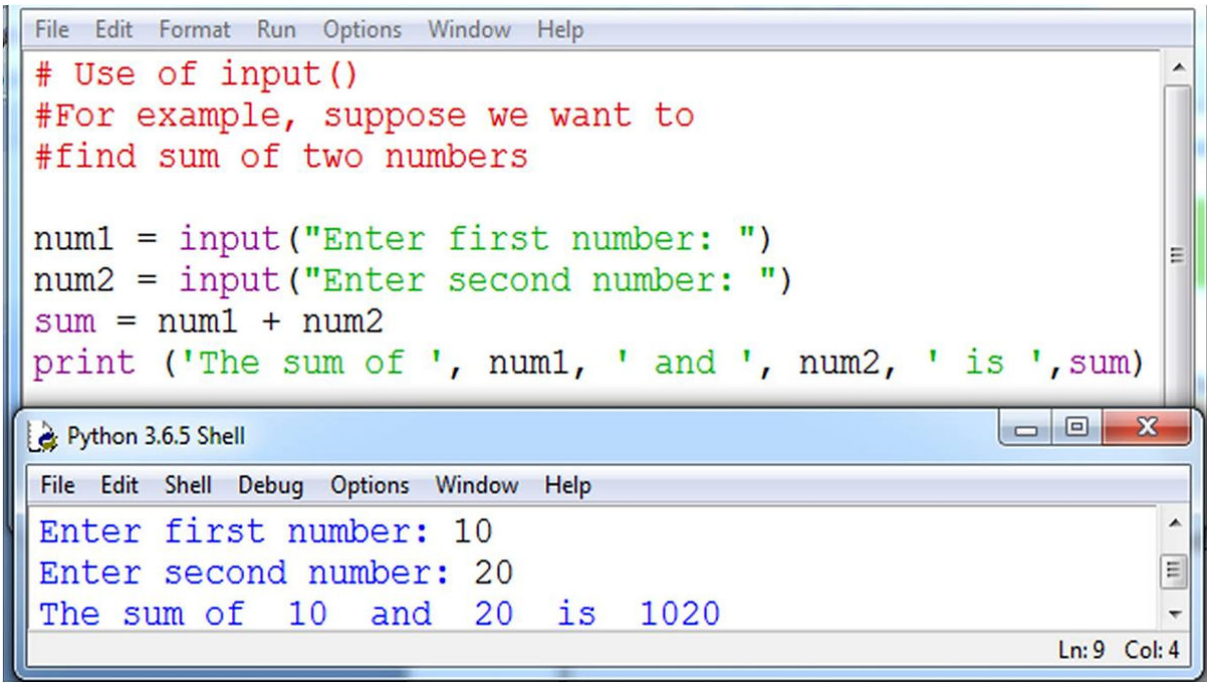
• The prompt is an optional parameter and where prompt is the string you wish to display on the screen.

**For example,**




**Inputting Numeric Values**

Suppose we want to find the sum of two numbers and the program is:




When we execute the above program, the program doesn't add the two numbers together; it just puts one right after the other as 1020

**The solution to the above problem is:**
Python offers two functions ***int()*** and ***float()*** to be used with **input ( )** to convert the values received through **input ( )** into ***int*** and ***float*** types.

**Accept an Integer input from User**
We need to convert an input string value into an integer using an ***int()*** function.

```
# Both input values are converted into integer type
num1 = int (input("Enter first number: "))
num2 = int(input("Enter second number: "))
sum = num1 + num2
print ('The sum of ', num1, ' and ', num2, ' is ',sum)
```

**OUTPUT:**
**Enter the first number: 10**
**Enter second number: 30**
**The sum of 10 and 30 is 40**

**Accept a float input from User**
We need to convert an input string value into a float using a ***float()*** function.

```
# Both input values are converted into float type
num1 = float (input("Enter first number: "))
num2 = float(input("Enter second number: "))
sum = num1 + num2
print ('The sum of ', num1, ' and ', num2, ' is ',sum)
```

**OUTPUT:**
**Enter the first number: 10.5**
**Enter second number: 30.5**
**The sum of 10 and 30 is 41.0**

**Get multiple input values from a user in one line**
In Python, we can accept two or three values from the user in one **input()** call.
**name,Class,Marks = input("Enter your Name,Class,Marks separated by space: ").split()**
**print("User Details: ",name, Class,Marks)**

**OUTPUT:**
**Enter your Name,Class,Marks separated by space: Ashaz 4 98**
**User Details: Ashaz 4 98**

# The print () function

- The print ( ) function allows you to print out the value of a variable and strings in parenthesis.
- The print() function will print as strings everything in a comma-separated sequence of expressions, and it will separate the results with single blanks by default.
- The print() function prints strings in between quotes (either single or double). If a print statement has quotes around text, the computer will print it out just as it is written.
- To print multiple items, separate them with commas. The print() function inserts a blank between objects.
- The print() function automatically appends a newline to output. To print without a newline, use end separator after the last object.

The general form of **print ()** function is:

**print** ( )

**print** (value/expr)

**print** (value, ...., sep=' ', end='\n', file=sys.stdout, flush=False)

Where,
• **sep:** The optional parameter sep is a separator between the output values. We can use a character, integer or a string as a separator. The default separator is space.
• **end:** This is also optional and it allows us to specify any string to be appended after the last value. The default is a new line.
• The **file** is the object where the values are printed and its default value is sys.stdout (screen).
• **flush** force it to "flush" the buffer, meaning that it will write
everything in the buffer to the terminal, even if normally waits before doing so.
**Uses and Examples of print() function in Python**
**1)** The print ( ) function can be used to print a blank line.
**For example:**

```
print('Learn python from')
print() # it print a blank line in between two print statement
print('GLA University')
```
**OUTPUT:**
```
Learn python from

GLA University
```

**2)** The print() function prints the value of variable.
**For example:**
```
N1 = 20
N2 = 30
Sum = N1 + N2
print(Sum) # prints a value
```
The above print statement receives a variable **Sum**

**3)** The print() function prints the value of an expression.
**For example:**
```
N1 = 20
N2 = 30
print(N1 + N2) # prints a value of an expression i.e., N1+N2
```
The above print statement prints a value of an expression
**OUTPUT:**
```
50
```
**4)** The print() function prints two values separated with comma (,)
**For example:**
```
Ver = 3.9
lang = 'Python'
print (lang, Ver) # prints two values separated with comma (,)
```
**OUTPUT:**
```
Python 3.9
```
**5)** The print() function prints a message
**For example:**
```
N1 =20
N2 = 30
sum = N1 + N2
print ('The sum of two numbers is')
# print statement receives a string message
print (sum)
```
**OUTPUT:**
```
The sum of two numbers is
50
```
**6)** The print() function prints a value, message or both message and values and vice versa.
**For example:**
```
N1 = 20
N2 = 30
Sum = N1 + N2
```

**print ('The sum of two numbers is', Sum)**
**# print is alternate way**
**print(Sum, "is the sum of", N1, "and", N2)**
**print() with Concatenate Operator (+)**

- The print() function can print two strings either using comma (,) operator, or a concatenate operator (+).
- When we use a concatenate operator (+) between two strings, both the strings are joined without leaving any space.
- Do not use the concatenate operator (+) with the combination of numeric and string data.
- If you do so then the Python interpreter will produce an error as Can't convert 'int' object to str implicitly, i.e., cannot concatenate 'str' and 'int' objects.

**For example:**
**print("I Learn Python", "with hyperskill.org")**
**# Print with a space separator.**
**will produce:**
**I Learn Python with hyperskill.org**
**Similarly, by using the concatenate operator (+), the print() function is:**
**print("I Learn Python" + "with hyperskill.org")**
**# Print without a separator.**
**will produce:**
**I Learn Pythonwith hyperskill.org**
**Print with Keyword Parameter *sep* and *end***
The print() function has two keyword arguments **sep** and **end** to handle the complex jobs for us to print standard output. Keyword parameters must be listed at the end of the parameter list. These are:
***sep:***

- Defines the string that is to be placed between every two values printed.
- By default, ***sep*** inserts a space (‘ ’).
- If the ***sep*** option is missing, space is printed between the values received by print() function.

**For example:**
**# It specifies the separator between the outputted arguments.**
**print('H','e','l','l','o',sep='-')**
**will produce:**
**H-e-l-l-o**
**print(**"My"**, "name", "is", "Stephen", sep="*")**
**will produce:**
**My*name*is*Stephen**
***end:***

- This separator defines the string to be printed at the end of the print() function.
- By default, end inserts a new line ('\n').
- If you don't supply the end option, a newline is printed by default.
- The string assigned to the end keyword argument can be of any length

**For example:**
**print("My name is Stephen", end = ' ')**

**print("and learn python")**
**will produce:**
**My name is Stephen and learn python**
If we wish to print the first two lines in one line and last two lines in another line with a comma (,) and space separator, then the code will be written as:

```
print("My name is A. Khan", end = ', ')
print("Class-XII A.")
print("I am in Sr. Sec. School", end = ', ')
print("Delhi, India.")
```

**will produce:**
**My name is A. Khan, Class-XII A.**
**I am in Sr. Sec. School, Delhi, India.**