



Command Line Interaction

Contents

1. Input
2. `input()` Function
3. `Input()` Returns String Data
4. Type Conversion
5. Function Composition
6. Escape Characters
7. Multiple Lines

Input

Enter your age:

Input

```
Enter your age: 32
```

```
In one year, you will be 33!
```

Input

We need:

1. A way to retrieve what the user types
2. A way to store the data



Use a variable

Use the `input()` function

input() function

We need:

1. A way to retrieve what the user types
2. A way to store the data



Use a variable

Use the `input()` function

input() Function

```
name = input('What is your name: ')\nprint(name)
```

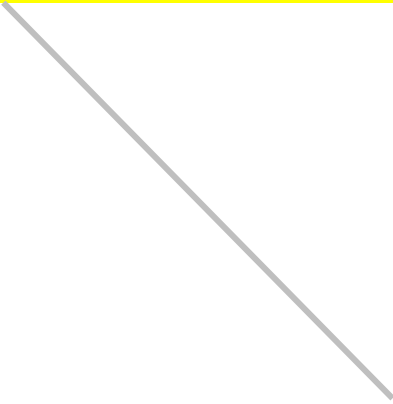
Use the `input()` function to retrieve text from the user

The prompt that user sees before they start to type.

input() Function

```
name = input('What is your name: ')
```

```
print(name)
```

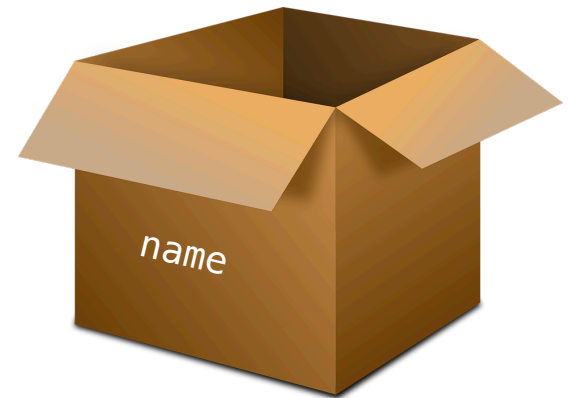


Program pauses on the first line until the user presses the enter key.

input() Function

```
name = input('What is your name: ')\nprint(name)
```

Whatever the user types
is then stored in a variable
called name

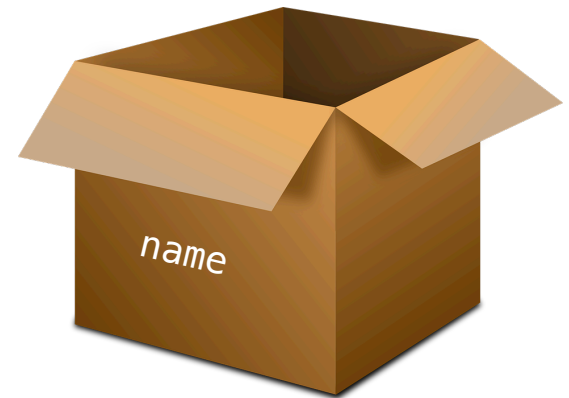


input() Function

```
name = input('What is your name: ')
```

```
print(name)
```

The variable is printed
just like any other variable



input() Returns String Data


```
name = input('What is your name: ')\nprint(name)\nprint(type(name))
```



<class 'str'>

input() Returns String Data


```
name = input('What is your name: ')\nprint('You said your name is ' + name + '.')
```



Since name is a string, it can be concatenated

input() Returns String Data

```
age = input('How old are you (in years): ')  
print('You are ' + age + ' years old.')  
print(type(age))
```



Even though the user enters a number, it is still returned and stored as a string.

input() Returns String Data

```
age = input('How old are you (in years): ')  
age = age + 1
```

TypeError: Must be str, not int

Type Conversion

```
age = input('How old are you (in years): ')
age = int(age) + 1
```

Alternative:

```
age = int(input('How old are you (in years): '))
age = age + 1
```

Function Composition

```
age = int(input('How old are you (in years): '))  
age = age + 1
```



The diagram consists of two lines originating from the code above. One line starts at the closing parenthesis of the `int(input(...))` expression in the first line and extends diagonally down and to the left. The other line starts at the `age` variable in the second line (`age + 1`) and extends diagonally down and to the right. These two lines converge towards the word 'Function' centered below the space between the two code lines.

Function

Composed functions are evaluated from inside out.

Function Composition

```
number = int(input('Enter a number: '))
```

```
number = int('88')
```

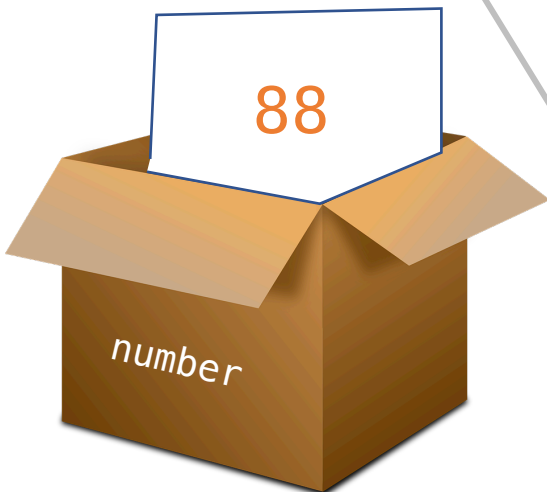
```
number = 88
```

Python first prints the prompt
and waits for user to enter text

User's text given to `int()`
function

`int()` function converts text
to integer

Number assigned to variable



Escape Character

In Python strings, the backslash "\" is a special character, also called the "escape" character.

Escape Character

```
print('You're a coder.')
```

```
print('You're a coder')
```

SyntaxError: invalid syntax

Fix with an escape character:

```
print('You\'re a coder.')
```

Escape Character

Escape character is also used in representing certain whitespace characters:

<code>'\t'</code>	tab
<code>'\n'</code>	newline
<code>'\r'</code>	carriage return

Example:

```
print('You\t're a coder.\nI\'m too.')
```

```
You're a coder.  
I am too.
```

Escape Character

Prefixing a special character with "\" turns it into an ordinary character.

Example:

```
print('Newline is \\n')
```

Newline is \n

Multiple Lines

Use triple single quotes ''' or triple double quotes """ to enclose the strings that span multiple lines.

Example:

```
print('''  
Haiku of a dragonfly:  
  
    Hover dart hover,  
A dazzlement of colour,  
    Joy of the summer.  
''')
```