



Python Input and Variables

Made by Danny Zou with help from
Nate Bomar



Review!

Pythonic Strings:

```
"this is a string"
```

```
'this is a string'
```

```
'this is also a string'
```

```
'this is also a string'
```

```
'''this is a multi-line string  
it can go onto several lines!  
just make sure to close it afterwards!'''
```

```
'this is a multi-line string\nit can go onto several lines!\njust make sure to close it afterwards!'
```

More Review!

Print Method

```
print('this is a string')
```

this is a string

```
print("""this is a printed  
multiple line string""")
```

this is a printed
multiple line string

```
print(34)  
print('This is a printed integer.')
```

34
This is a printed integer.

Variables

This is how you make a variable!

```
1  john = 5
2  joanne = "hello"
3  ron = True
```

This is why variables are useful!

Code:

```
main.py ×  
1  john = 5  
2  joanne = "hello"  
3  ron = True  
4  
5  for i in range(0, 5):  
6      john = john + 1  
7  print(john)
```

Output:

```
Console  Shell  Markdown  
  
6  
7  
8  
9  
10  
✦
```

With a variable, you can store information

Input

How do we get information from the user? Let's ask W3Schools

Python input() Function

[< Built-in Functions](#)

Example

Ask for the user's name and print it:

```
print('Enter your name:')  
x = input()  
print('Hello, ' + x)
```

[Try it Yourself »](#)

Definition and Usage

The `input()` function allows user input.

Let's Try It!

No variables for now!

```
[9]: input()
```

```
hi
```

```
[9]: 'hi'
```

Now With Variables!

```
joe = input()
```

```
joe is a computer programmer and everyone feels bad for him, his job sucks
```

```
print(joe)
```

```
joe is a computer programmer and everyone feels bad for him, his job sucks
```


What if we wanted an Integer?

Let's try it!

```
main.py x
1  joe = input()
2  print(joe)
```

```
Console  Shell  Markdown
5
5
> []
```

Can you tell what's wrong here?

The problem

The Pythonic `type()` function returns the type of object given to it.

```
main.py x
1  joe = input()
2  print(joe)
3
4  print(type(joe))
```

```
Console Shell Markdown
5
5
<class 'str'>
❏
```

joe isn't an integer after all! It's a string!

How do we fix this?

Typecasting

Typecasting means to change from one type of object to another. Let's convert a string to an integer!

```
main.py ×
1  joe = input()
2  print(joe)
3
4  print(type(joe))
5
6  joe = int(joe)
7
8  print(type(joe))
```

```
Console  Shell  Markdown
5
5
<class 'str'>
<class 'int'>
> []
```

Typecasting Explained

There are certain built-in Python functions that enable you to change objects from one type to another. With strings, integers, floats, and other primitive types, these make life much easier!

`int()` converts things to integers

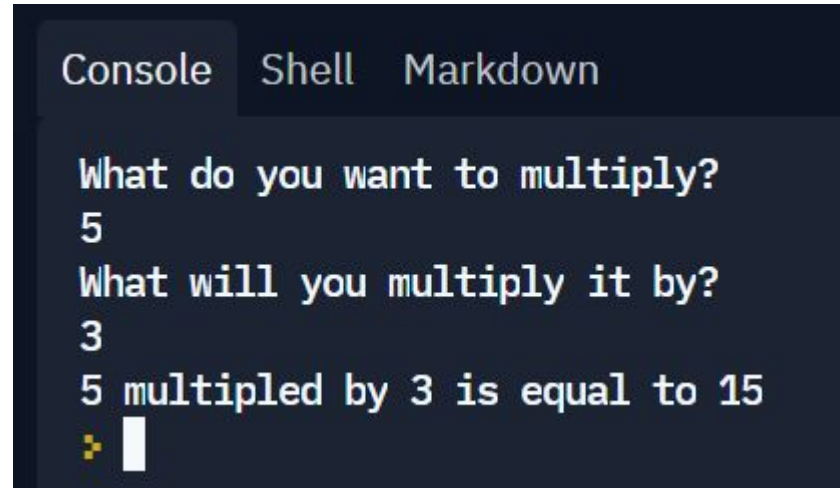
`str()` converts things to strings

`float()` converts things to floats

Lab

Take in two numbers from user input and multiply them, returning the product.

Lab Output:

A screenshot of a terminal window with a dark background. At the top, there are three tabs: 'Console' (which is selected and highlighted), 'Shell', and 'Markdown'. The terminal shows the following text: 'What do you want to multiply?' followed by the user input '5'. Then it asks 'What will you multiply it by?' followed by the user input '3'. Finally, it displays the output '5 multiplied by 3 is equal to 15'. At the bottom left, there is a small yellow icon of a cursor and a white rectangular cursor line.

```
Console Shell Markdown
What do you want to multiply?
5
What will you multiply it by?
3
5 multiplied by 3 is equal to 15
>
```