User Input

How to assign variables with user input.

So far, we've covered how to assign variables values directly in a Python file. However, we often want a user of a program to enter new information into the program.

How can we do this? As it turns out, another way to assign a value to a variable is through user input.

While we output a variable's value using `print()`, we assign information to a variable using `input()`. The `input()` function requires a prompt message, which it will print out for the user before they enter the new information. For example:

```
likes_snakes = input("Do you like snakes? ")
```

In the example above, the following would occur:

1. The program would print `"Do you like snakes? "` for the user.
2. The user would enter an answer (e.g., `"Yes! I have seven pythons as pets!"`) and press enter.
3. The variable `likes_snakes` would be assigned a value of the user's answer.

Try constructing a statement to collect user input on your own!

Fill in the Code

Fill in the blanks in the code to complete a statement that asks a user "What is your favorite flightless bird?" and then stores their answer in the variable `favorite_flightless_bird`.

```
favorite_flightless_bird = input (
"What is your favorite flightless bird?" )
```

You got it!

Not only can `input()` be used for collecting all sorts of different information from a user, but once you have that information stored as a variable you can use it to simulate interaction:

```
>>> favorite_fruit = input("What is your favorite fruit? ")
What is your favorite fruit? mango

>>> print("Oh cool! I like " + favorite_fruit + " too, but I think my favorite fruit is apple.")
Oh cool! I like mango too, but I think my favorite fruit is apple.
```

These are pretty basic implementations of `input()`, but as you get more familiar with Python you'll find more and more interesting scenarios where you will want to interact with your users.


Back
Next