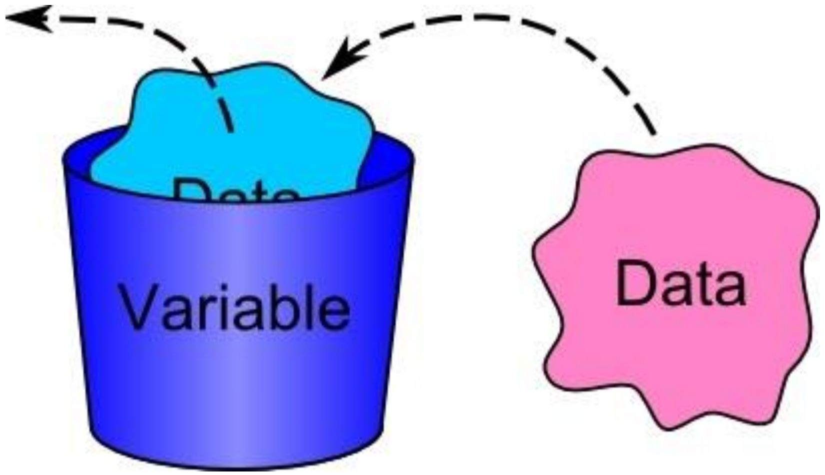
VARIABLES

VARIABLES

If you want to write useful code, you’ll need to be able to store and label pieces of information. That’s what variables do.

Variables are great for all sorts of things— from tracking your score in a game to performing calculations and holding lists of items.

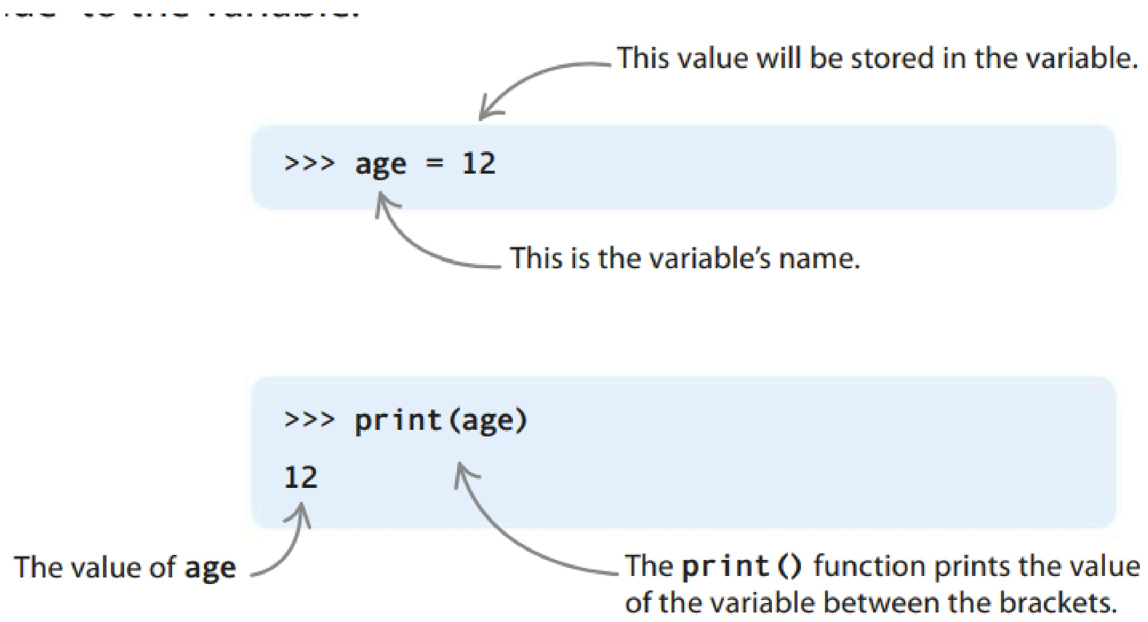


How to create a variable

A variable needs a name. Think of a name that will remind you what's inside the variable.

Then decide what you want to store in the variable. This is the variable's value.

Type the name, followed by an equals sign, followed by the value.
We call this "assigning a value" to the variable.



This value will be stored in the variable.

```
>>> age = 12
```

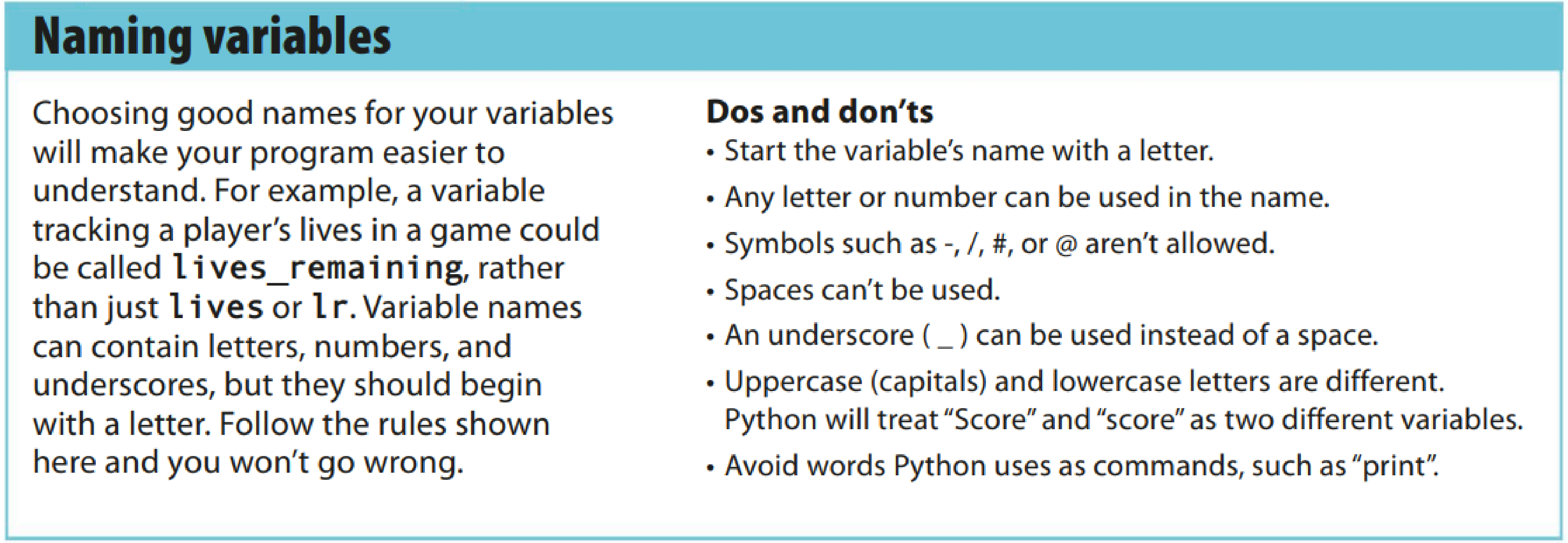
This is the variable's name.

```
>>> print(age)
12
```

The value of age

The print () function prints the value of the variable between the brackets.

Naming your variables



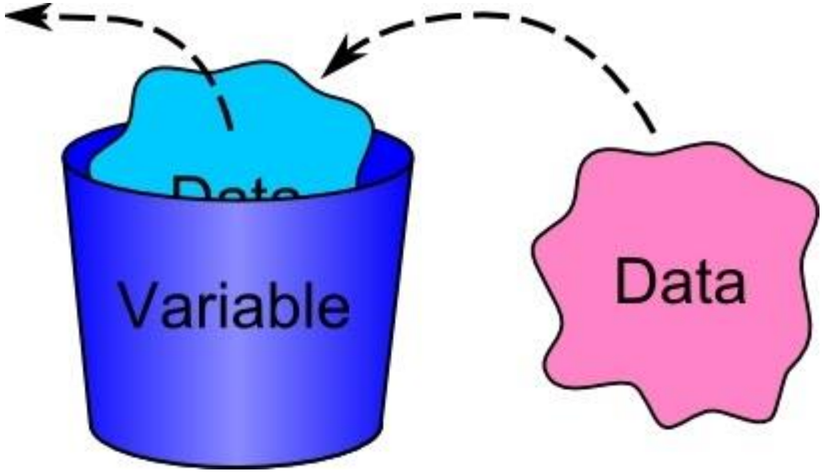
Naming variables

Choosing good names for your variables will make your program easier to understand. For example, a variable tracking a player's lives in a game could be called `lives_remaining`, rather than just `lives` or `lr`. Variable names can contain letters, numbers, and underscores, but they should begin with a letter. Follow the rules shown here and you won't go wrong.

Dos and don'ts

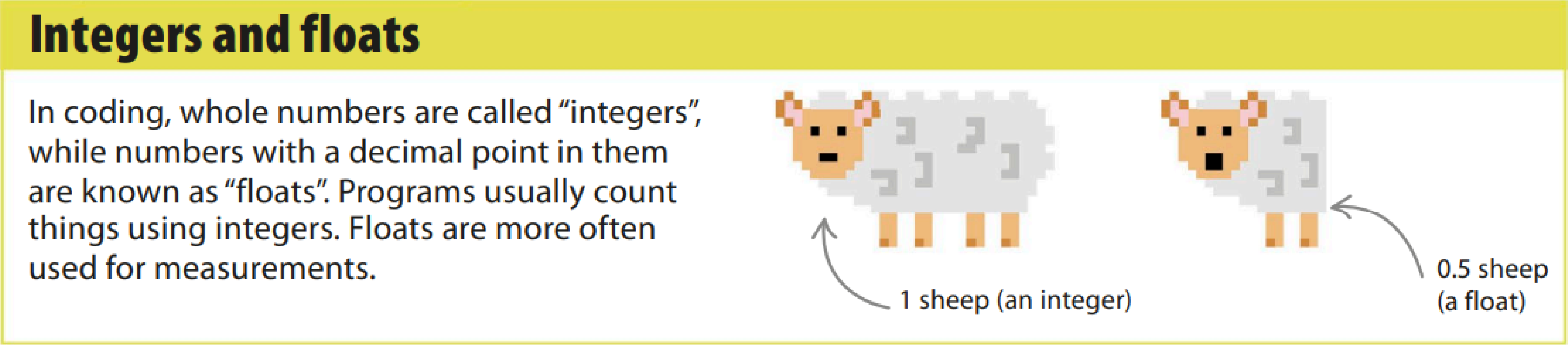
- Start the variable's name with a letter.
- Any letter or number can be used in the name.
- Symbols such as -, /, #, or @ aren't allowed.
- Spaces can't be used.
- An underscore (_) can be used instead of a space.
- Uppercase (capitals) and lowercase letters are different. Python will treat "Score" and "score" as two different variables.
- Avoid words Python uses as commands, such as "print".

Data types



Numbers
(Integers and floats)

Integers and floats



Integers and floats

In coding, whole numbers are called "integers", while numbers with a decimal point in them are known as "floats". Programs usually count things using integers. Floats are more often used for measurements.

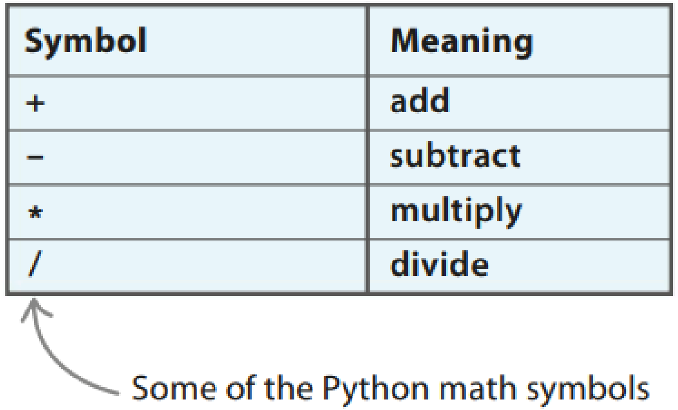
1 sheep (an integer)

0.5 sheep
(a float)

Integers and floats

Using numbers

Variables can be used to store numbers and do sums. You can use them with symbols to do calculations, just like you do in maths. Some of these symbols will be familiar, but watch out for the symbols meaning "multiply" and "divide"—they're slightly different from the ones you use in class.



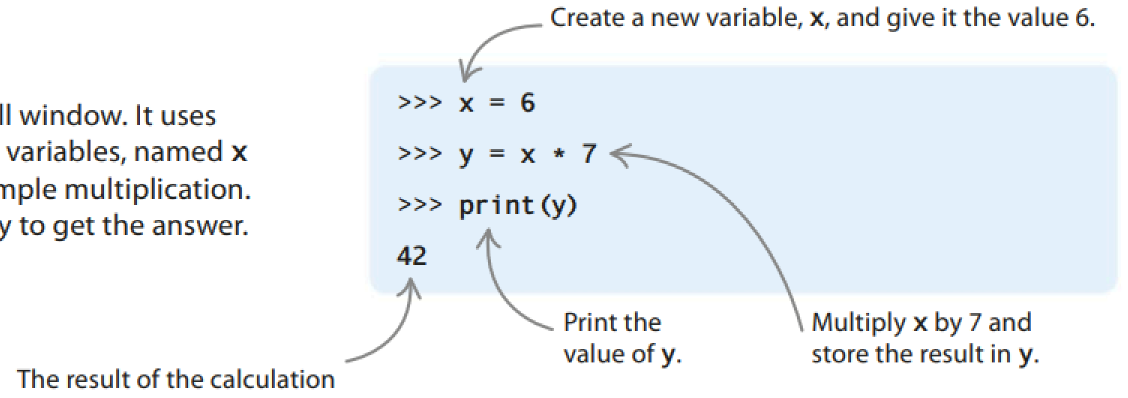
Symbol	Meaning
+	add
–	subtract
*	multiply
/	divide

Some of the Python math symbols

Using Integers and floats

1 A simple calculation

Type this code in a shell window. It uses numbers stored in two variables, named **x** and **y**, to carry out a simple multiplication. Hit the enter/return key to get the answer.



Create a new variable, **x**, and give it the value 6.

```
>>> x = 6
>>> y = x * 7
>>> print(y)
42
```

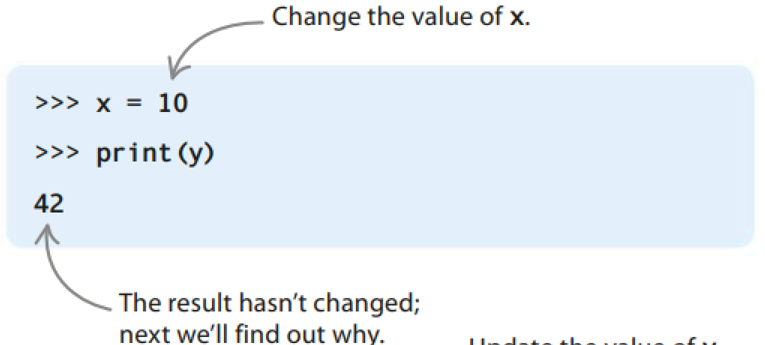
Print the value of **y**.

Multiply **x** by 7 and store the result in **y**.

The result of the calculation

2 Change a value

To change the value of a variable, you just assign a new value to it. In your code, change the value of **x** to 10 and run the calculation again. What do you expect the result to be?



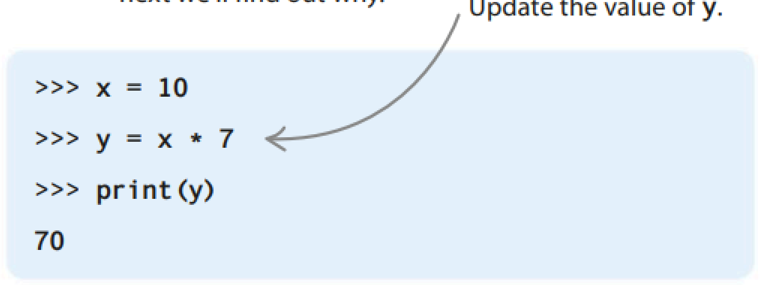
Change the value of **x**.

```
>>> x = 10
>>> print(y)
42
```

The result hasn't changed; next we'll find out why.

3 Update the value

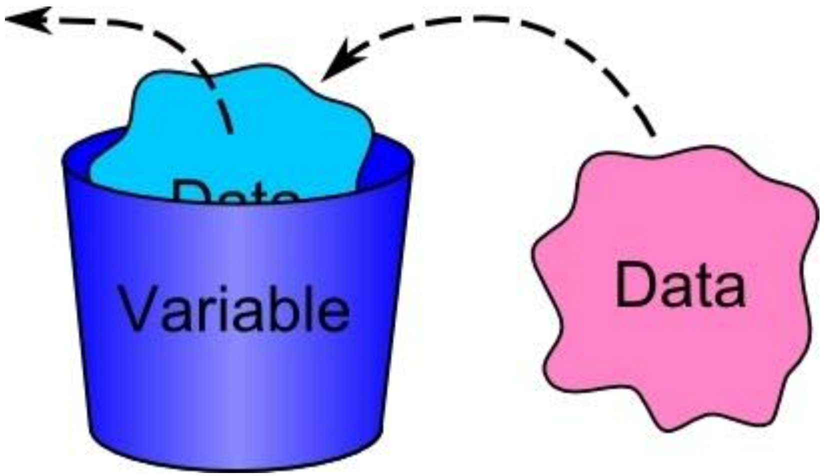
The value of **y** needs to be updated to get the correct result. Type these lines. Now the code assigns the new value to **y** after **x** has been changed. If you update the value of one variable in your own programs, always check to see if you need to update any others.



Update the value of **y**.

```
>>> x = 10
>>> y = x * 7
>>> print(y)
70
```

Data types



Strings

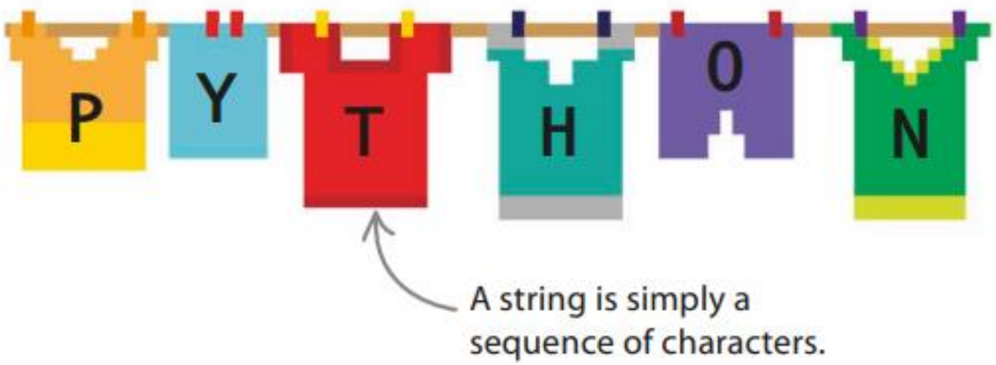
(text)

Strings

We use the word "string" for any data made up of a sequence of letters or other characters.

Words and sentences are stored as strings.

Every character that you can type on your keyboard, and even those you can't, can be stored in a string.



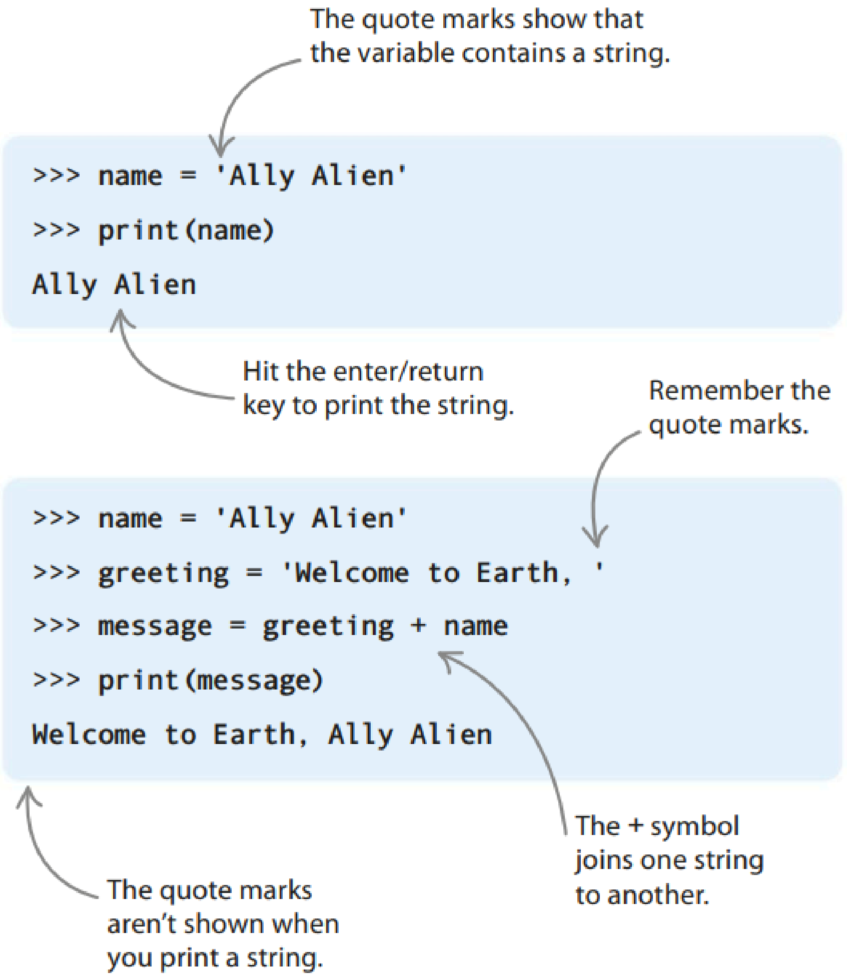
Using Strings

1 Strings in variables

Strings can be put into variables. Type this code into the shell window. It assigns the string `'Ally Alien'` to the variable **name** and then displays it. Strings must always have quotation marks at the beginning and end.

2 Combining strings

Variables become really useful when you combine them to make new variables. If you add two strings together, you can store the combination in a new variable. Try this out.



The quote marks show that the variable contains a string.

```
>>> name = 'Ally Alien'
>>> print(name)
Ally Alien
```

Hit the enter/return key to print the string.

Remember the quote marks.

```
>>> name = 'Ally Alien'
>>> greeting = 'Welcome to Earth, '
>>> message = greeting + name
>>> print(message)
Welcome to Earth, Ally Alien
```

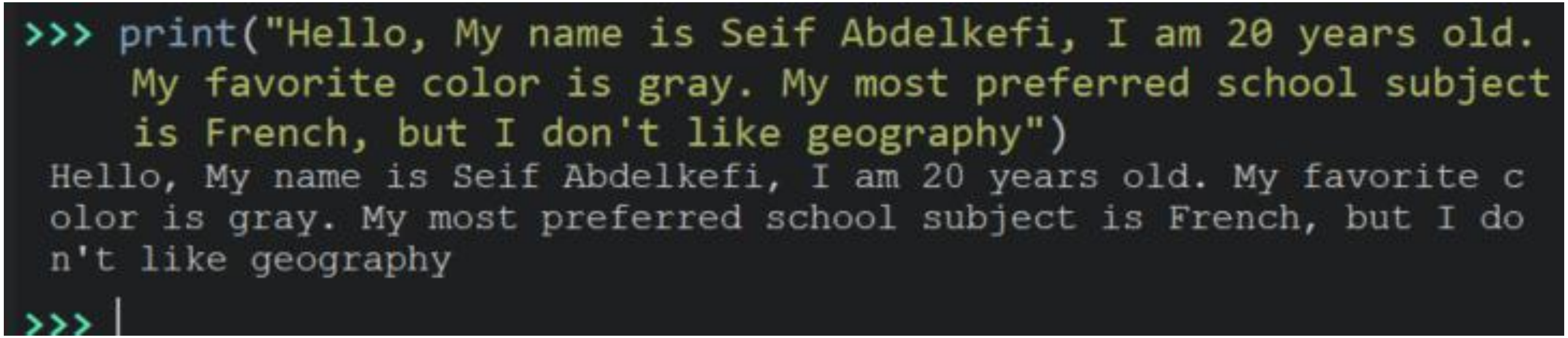
The + symbol joins one string to another.

The quote marks aren't shown when you print a string.

Time to code

Introduction paragraph

Make the shell print this message
“Hello, My name is Seif Abdelkefi, I am 20 years old. My favorite color is gray. My most preferred school subject is French, but I don't like geography ”

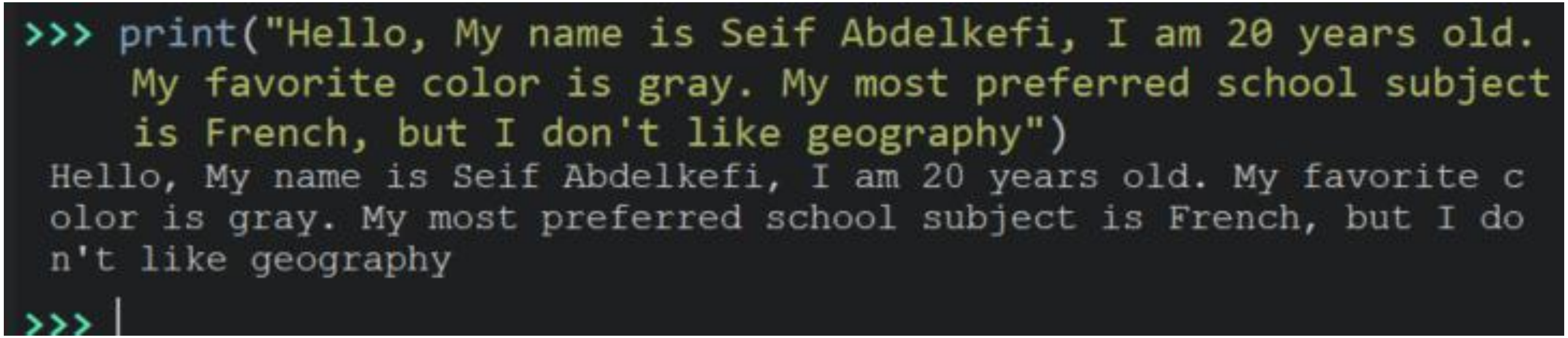


```
>>> print("Hello, My name is Seif Abdelkefi, I am 20 years old.
     My favorite color is gray. My most preferred school subject
     is French, but I don't like geography")
 Hello, My name is Seif Abdelkefi, I am 20 years old. My favorite c
 olor is gray. My most preferred school subject is French, but I do
 n't like geography

>>> |
```

Introduction paragraphwith variables

Make the shell print this message
“Hello, My name is Seif Abdelkefi, I am 20 years old. My favorite color is gray. My most preferred school subject is French, but I don't like geography ”



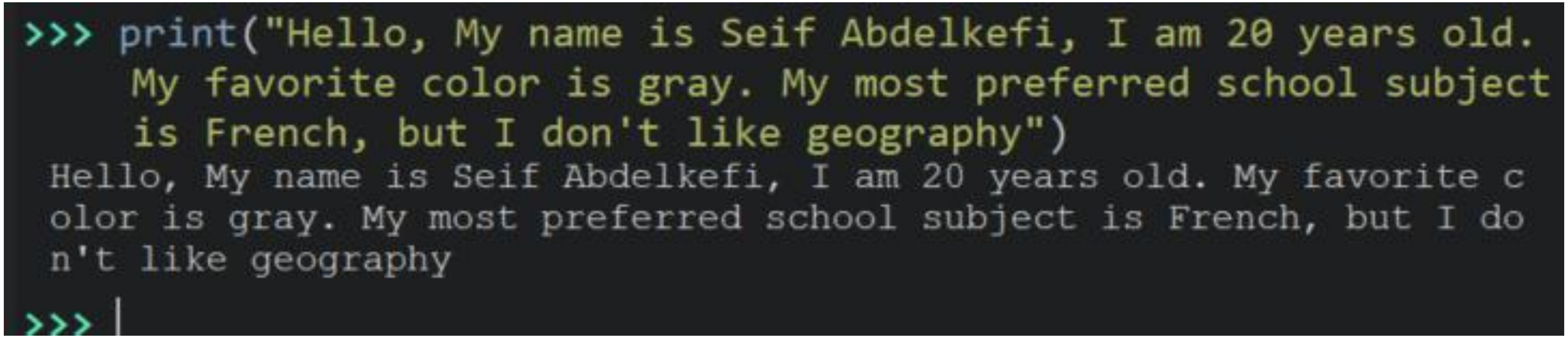
```
>>> print("Hello, My name is Seif Abdelkefi, I am 20 years old.
     My favorite color is gray. My most preferred school subject
     is French, but I don't like geography")
 Hello, My name is Seif Abdelkefi, I am 20 years old. My favorite c
 olor is gray. My most preferred school subject is French, but I do
 n't like geography

>>> |
```

Introduction paragraphwith variables

Make the shell print this message

“Hello, My name is Seif Abdelkefi, I am 20 years old. My favorite color is gray. My most preferred school subject is French, but I don't like geography ”



```
>>> print("Hello, My name is Seif Abdelkefi, I am 20 years old.
     My favorite color is gray. My most preferred school subject
     is French, but I don't like geography")
 Hello, My name is Seif Abdelkefi, I am 20 years old. My favorite c
 olor is gray. My most preferred school subject is French, but I do
 n't like geography
>>> 
```

Introduction paragraph with variables

Create 6 variables called:

- ✓ name
- ✓ age
- ✓ fav_color
- ✓ Fav_school_subject
- ✓ Worst_school_subject

Assign appropriate values to these variables.

Use the variables in the big print statement to print the same text.

Introduction paragraph with variables

Create 6 variables called:

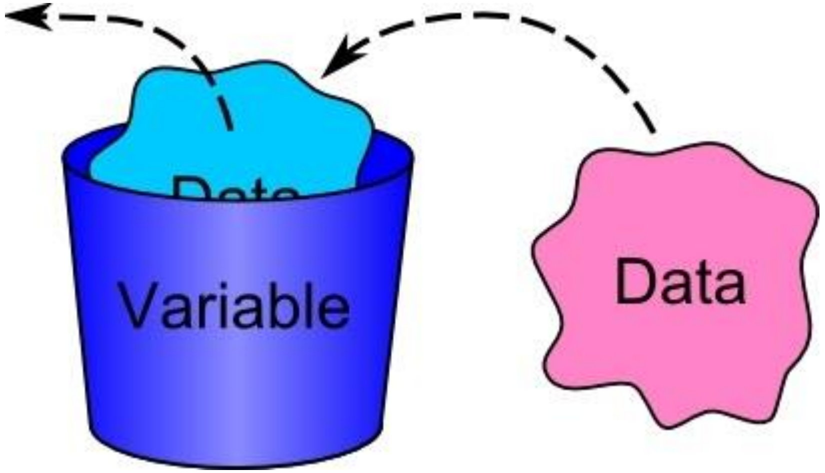
- ✓ name
- ✓ age
- ✓ fav_color
- ✓ Fav_school_subject
- ✓ Worst_school_subject

Assign appropriate values to these variables.

Use the variables in the big print statement to print the same text.

Only now whenever you want to change an information about yourself you do it directly from the variables ;)

Data types



Lists

Lists

When you want to store a lot of data, or perhaps the order of the data is important, you may need to use a list.

A list can hold many items together and keep them in order. Python gives each item a number that shows its position in the list.

You can change the items in the list at any time.

Lists

1 Multiple variables

Imagine you're writing a multiplayer game and want to store the names of the players in each team. You could create a variable for each player, which might look like this...

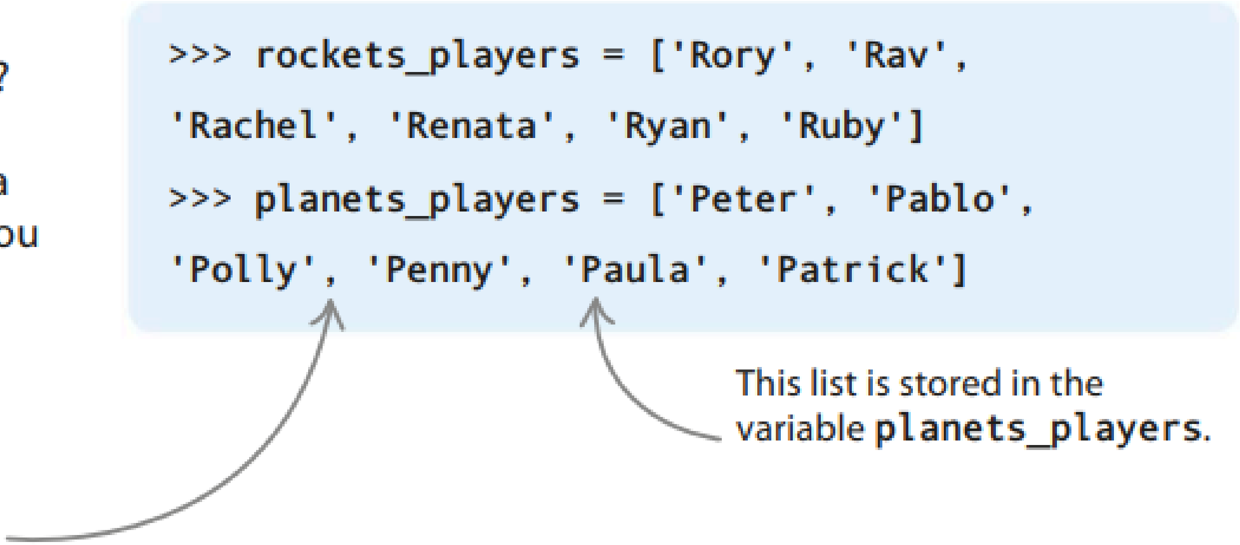
```
>>> rockets_player_1 = 'Rory'
>>> rockets_player_2 = 'Rav'
>>> rockets_player_3 = 'Rachel'
>>> planets_player_1 = 'Peter'
>>> planets_player_2 = 'Pablo'
>>> planets_player_3 = 'Polly'
```

With three players per team, you'd need six variables.

Lists

2 Put a list in a variable

...but what if there were six players per team? Managing and updating so many variables would be difficult. It would be better to use a list. To create a list, you surround the items you want to store with square brackets. Try out these lists in the shell.



```
>>> rockets_players = ['Rory', 'Rav',
'Rachel', 'Renata', 'Ryan', 'Ruby']
>>> planets_players = ['Peter', 'Pablo',
'Polly', 'Penny', 'Paula', 'Patrick']
```

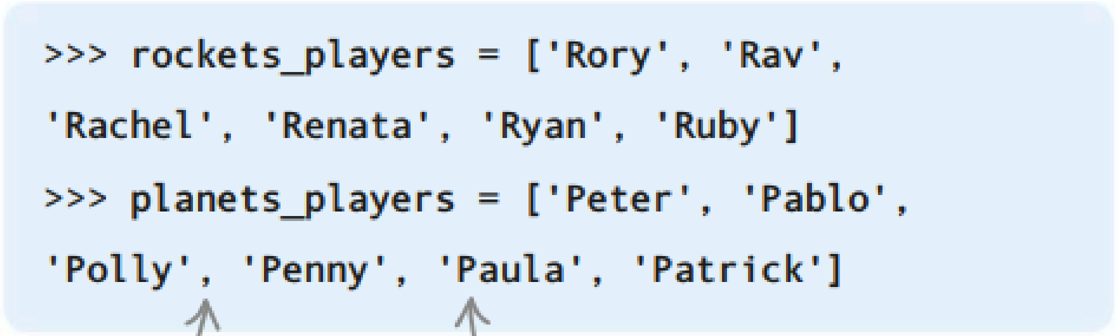
The list items must be separated by commas.

This list is stored in the variable `planets_players`.

Lists

2 Put a list in a variable

...but what if there were six players per team? Managing and updating so many variables would be difficult. It would be better to use a list. To create a list, you surround the items you want to store with square brackets. Try out these lists in the shell.



```
>>> rockets_players = ['Rory', 'Rav',
'Rachel', 'Renata', 'Ryan', 'Ruby']
>>> planets_players = ['Peter', 'Pablo',
'Polly', 'Penny', 'Paula', 'Patrick']
```

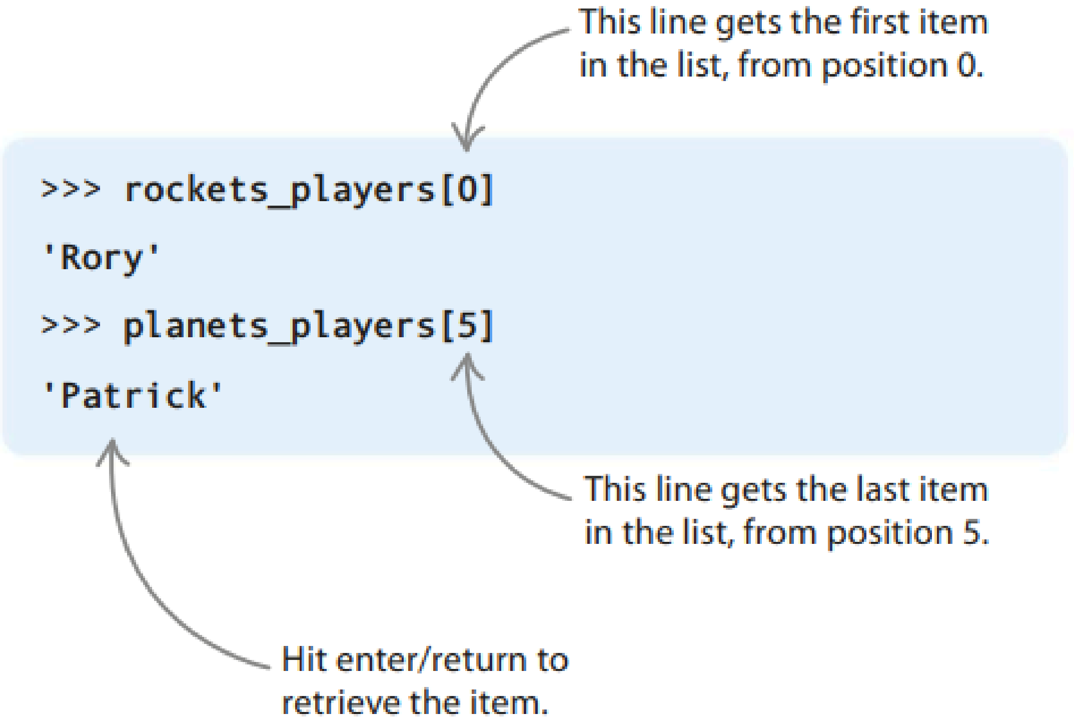
This list is stored in the variable `planets_players`.

The list items must be separated by commas.

Lists

3 Getting items from a list

Once your data is in a list, it's easy to work with. To get an item out of a list, first type the name of the list. Then add the item's position in the list, putting it inside square brackets. Be careful: Python starts counting list items from 0 rather than 1. Now try getting different players' names out of your team lists. The first player is at position 0, while the last player is at position 5.



```
>>> rockets_players[0]
'Rory'
>>> planets_players[5]
'Patrick'
```

This line gets the first item in the list, from position 0.

This line gets the last item in the list, from position 5.

Hit enter/return to retrieve the item.

Introduction paragraph with variables (+lists)

Create 6 variables called:
- ✓ name
- ✓ age
- ✓ fav_colors (list)
- ✓ Fav_school_subjects (list)
- ✓ Worst_school_subject

Assign appropriate values to these variables.

Use the variables in the big print statement to print the same text.

Only now whenever you want to change an information about yourself you do it directly from the variables ;)