

Variables and Multiple Python Turtles

Textbook

Variables and Multiple Python Turtles



What if you want to make multiple turtles on your page? You do this by creating variables.

Each turtle is stored in a variable. A [variable](#) is a building block of programming that allows you to store data. Think of it like a "box" that can store anything you put in it.

In Python, you use variables to save information for later in your program. Initializing a variable means giving it a starting value. The variable's value can change multiple times while your program runs.

1. You can name "the box", or variable anything you want and you can put any type of data inside. **The name of the variable must either be one word, or connected with underscores.**

```
pet = "dog" or my_pet = "dog"
```

2. You do not need to specify anything before declaring a variable in Python, unlike other programming languages.
3. You use the equal sign `=` to assign the data to the variable or "put it in the box."

Create a Variable



Let's create a variable that will hold a Python turtle. Let's name our turtle `my_turtle`. This is done by assigning `turtle.Turtle()` to your variable.

```
1 import turtle
2 turtle.getscreen()
3 my_turtle = turtle.Turtle()
```

Now we have just created a variable named `my_turtle`. In other words, you created another turtle named `my_turtle`. Notice that you must have the same capitalization as in the example.

Now let's move our turtle forward!

```
1 import turtle
2 turtle.getscreen()
3 my_turtle = turtle.Turtle()
4
5 my_turtle.forward(100)
```

But wait, why are there two turtles on my screen? This is because of the default variable.

Default Variable in Turtles

We have actually been using a turtle variable this whole time! Consider the following code.

```
1 import turtle
```

```
2 turtle.getscreen()  
3  
4  
5 turtle.forward(100)
```

The code `turtle.forward(100)` is telling the variable named `turtle` to move forward. But where did we create the variable? It is already created in the turtles library. When we import the turtle library on the first line, we are importing the variable that has been created in the library.

So within the Python turtles library, there is a variable named `turtle` that we can use. Or we can create our own variables and use them. This is why the following code has two turtles on the page.

```
1 import turtle  
2 turtle.getscreen()  
3 my_turtle = turtle.Turtle()  
4  
5 turtle.left(90)  
6 turtle.forward(100)  
7 my_turtle.forward(100)
```

Abstraction

Variables are like shortcuts for complex information in coding. Instead of typing the whole thing every time, we just use the variable name. This simplifying technique is called [abstraction](#).

We use abstraction in daily life too, like how a red light means "stop," or pressing "call" on your phone is a shortcut to entering a number and connecting to your friend, or how a simple contact name in your phone holds their phone number, address, and picture.

Abstraction makes things easier by using simple symbols to represent more complicated information.

Creating a variable is an example of abstraction. Now we can just use the variable name instead of retyping the code it holds every time.

Rules for Naming Variables

When deciding what to name your variables, try to name them something that refers to what it's doing. It is [best practice](#) to name your variables something that refers to what the variable will hold.

Here are a few more rules when it comes to naming variables.

- Variables cannot be more than one word long (We get around this rule by adding an underscore character `_` to join multiple words together. For example `first_name` is a valid variable name. This is called snake case.
- A variable name must start with a letter or an underscore character `_`
- A variable name cannot start with a number
- A variable name can only use capital A-Z, lowercase a-z, numbers 0-9, and the underscore character `_`. No special characters are allowed.

Checkpoint

Variables with Python Turtles

1. Make sure you are in a Python file with the extension .py.
2. Import the turtle library.
3. Generate the turtle screen.
4. Create a variable named `speedy` and assign it to a turtle from the Python turtle library.

Requirements:

- Import the turtle library.
- Generate the turtle screen.
- Create a variable named `speedy` and assign it to a turtle from the Python turtle library.

Questions (10)

1. How do you create a variable named "my_turtle" that holds a Python turtle?

MULTIPLE CHOICE

Choose the correct answer:

- A. `turtle.getscreen(my_turtle)`
- B. `my_turtle = turtle.Turtle()`
- C. `variable.my_turtle = turtle.Turtle()`
- D. `my_turtle = turtle.createTurtle()`

2. Why does the code `turtle.forward(100)` result in turtle movement, even though we didn't explicitly create a turtle variable?

MULTIPLE CHOICE

Choose the correct answer:

- A. It won't work--it will throw an error.
- B. The turtle library automatically creates a default variable named "turtle".
- C. The forward function doesn't require a specific turtle variable.
- D. All variables are named turtle in Python.

3. How is the concept of abstraction illustrated in the passage?

Choose the correct answer:

- A. By using variables to represent more complex information.
- B. Ignoring unnecessary details in programming.
- C. Creating turtles with minimal code.
- D. Avoiding the use of variables for simplicity.

4. What is the purpose of the following code snippet?

```
my_turtle = turtle.Turtle()
```

Choose the correct answer:

- A. Declaring a variable named my_turtle.
- B. Initializing a turtle screen.
- C. Importing the turtles library.
- D. Declaring a variable named turtle.

5. How can you make a turtle variable named henry move forward ?

Choose the correct answer:

- A. turtle.forward()
- B. my_turtle.move_forward(100)
- C. henry.forward(100)
- D. henry.move(my_turtle, 100)

6. What is NOT a valid variable name according to the rules mentioned in the passage?

Choose the correct answer:

- A. name@variable
- B. myname
- C. first_name
- D. second_variable2

7. Debug the following code:

[DEBUG CODE](#)

Code to Debug:

```
1 import turtle
2 turtle.getscreen()
3 fiona = turtle.turtle()
4
5 turtle.left(20)
6 turtle.forward(50)
7 fiona.forward(70)
```

8. Debug the following code:

[DEBUG CODE](#)

Code to Debug:

```
1 import turtle
2 turtle.getscreen()
3 sam = turtle.Turtle()
4
5 turtle.left(30)
6 turtle.forward(80)
7 sam.forward(10)
```

9. Debug the following code:

[DEBUG CODE](#)

Code to Debug:

```
1 import turtle
2 turtle.getscreen()
3 awesome_turtle = turtle.Turtle()
```

10. Debug the following code:

[DEBUG CODE](#)

Code to Debug:

```
1 import turtle
2 turtle.getscreen()
3 awesome_turtle = turtle.Turtle()
4
5 awesome_turtle.right(90)
6 awesome_turtle.forward(100)
```

Challenges (5)

1. Scooter the Turtle

Let's say you just got a pet turtle from the store! He's green with yellow stripes. You decided to name him scooter.

1. Make sure you are in a Python file with the extension .py.
2. Import the turtle library.
3. Generate the turtle screen.
4. Create a variable named `scooter` and assign it to a turtle from the Python turtle library.

Requirements:

- Import the turtle library.
- Generate the turtle screen.
- Create a variable named `scooter` and assign it to a turtle from the Python turtle library.

2. Bigfoot the Tortoise

Do you know what the differences are between a turtle and a tortoise?

Tortoises have more rounded and domed shells where turtles have thinner, more water-dynamic shells. Turtle shells are more streamlined to aid in swimming. One major key difference is that tortoises spend most of their time on land and turtles are adapted for life spent in water.

Let's say your particular turtle was actually a tortoise. You decided to name your tortoise bigfoot.

1. Make sure you are in a Python file with the extension .py.
2. Import the turtle library.
3. Generate the turtle screen.
4. Create a variable named `bigfoot` and assign it to a turtle from the Python turtle library.

Image from Reddit r/coolguides

Requirements:

- Import the turtle library.
- Generate the turtle screen.
- Create a variable named `bigfoot` and assign it to a turtle from the Python turtle library.

3. Really Long Turtle Name

Make a turtle variable with a super long name! Remember that names with multiple words need to be connected with an underscore. Name your turtle variable `shelly_sue_sammie_speedy_stripy_sister`

1. Make sure you are in a Python file with the extension .py.
2. Import the turtle library.
3. Generate the turtle screen.
4. Create a variable named `shelly_sue_sammie_speedy_stripy_sister` and assign it to a turtle from the Python turtle library.

Requirements:

- Import the turtle library.
- Generate the turtle screen.
- Create a variable named `shelly_sue_sammie_speedy_stripy_sister` and assign it to a turtle from the Python turtle library.

4. Name Your Own Pet

Create a turtle and name it whatever you want! If you had a pet turtle, what would you name it?

1. Make sure you are in a Python file with the extension .py.
2. Import the turtle library.
3. Generate the turtle screen.
4. Create a variable with whatever name you want and assign it to a turtle from the Python turtle library. Remember it needs to be one word long or connected by underscores for_example.

Requirements:

- Import the turtle library.
- Generate the turtle screen.
- Create a variable with whatever name you want and assign it to a turtle from the Python turtle library. Remember it needs to be one word long or connected by underscores for_example.

5. Two Turtles

Make two different turtle variables! Move them in different directions.

1. Make sure you are in a Python file with the extension .py.
2. Import the turtle library.
3. Generate the turtle screen.
4. Create a variable named `greenie` and assign it to a turtle from the Python turtle library.
5. Create a variable named `blue_dude` and assign it to a turtle from the Python turtle library.

Requirements:

- Import the turtle library.
- Generate the turtle screen.
- Create a variable named `greenie` and assign it to a turtle from the Python turtle library.
- Create a variable named `blue_dude` and assign it to a turtle from the Python turtle library.

Answer Keys & Solutions

Checkpoint Solutions

Variables with Python Turtles

```
1 import turtle
2 turtle.getscreen()
3 speedy = turtle.Turtle()
```

Questions

1. How do you create a variable named "my_turtle" that holds a Python turtle?

MULTIPLE CHOICE

Correct Answer:

- A. `turtle.getscreen(my_turtle)` ✗ Incorrect
- B. `my_turtle = turtle.Turtle()` ✓ Correct
- C. `variable.my_turtle = turtle.Turtle()` ✗ Incorrect
- D. `my_turtle = turtle.createTurtle()` ✗ Incorrect

Explanation:

A variable assigned a value with an equals sign

2. Why does the code `turtle.forward(100)` result in turtle movement, even though we didn't explicitly create a turtle variable?

MULTIPLE CHOICE

Correct Answer:

- A. It won't work--it will throw an error. ✗ Incorrect
- B. The turtle library automatically creates a default variable named "turtle". ✓ Correct
- C. The forward function doesn't require a specific turtle variable. ✗ Incorrect
- D. All variables are named turtle in Python. ✗ Incorrect

Explanation:

The turtle library already has a variable created

3. How is the concept of abstraction illustrated in the passage?

MULTIPLE CHOICE

Correct Answer:

- A. By using variables to represent more complex information. ✓ Correct
- B. Ignoring unnecessary details in programming. ✗ Incorrect
- C. Creating turtles with minimal code. ✗ Incorrect
- D. Avoiding the use of variables for simplicity. ✗ Incorrect

Explanation:

We can use a variable name, which is much simpler than typing in all that the variable represents.

4. What is the purpose of the following code snippet?

MULTIPLE CHOICE

Correct Answer:

- A. Declaring a variable named my_turtle. ✓ Correct
- B. Initializing a turtle screen. ✗ Incorrect
- C. Importing the turtles library. ✗ Incorrect
- D. Declaring a variable named turtle. ✗ Incorrect

Explanation:

`turtle.Turtle()` is the code to create a new turtle

5. How can you make a turtle variable named henry move forward ?

MULTIPLE CHOICE

Correct Answer:

- A. `turtle.forward()` ✗ Incorrect

B. `my_turtle.move_forward(100)`

✗ Incorrect

C. `henry.forward(100)`

✓ Correct

D. `henry.move(my_turtle, 100)`

✗ Incorrect

Explanation:

Just like `turtle.forward`, replace the variable name of `turtle` to `henry`.

6. What is NOT a valid variable name according to the rules mentioned in the passage?

MULTIPLE CHOICE

Correct Answer:

A. `name@variable`

✓ Correct

B. `myname`

✗ Incorrect

C. `first_name`

✗ Incorrect

D. `second_variable2`

✗ Incorrect

Explanation:

No special characters allowed

7. Debug the following code:

DEBUG CODE

Incorrect Code:

```
1 import turtle
2 turtle.getscreen()
3 fiona = turtle.turtle()
4
5 turtle.left(20)
6 turtle.forward(50)
7 fiona.forward(70)
```

Correct Solution:

```
1 import turtle
2 turtle.getscreen()
3 fiona = turtle.Turtle()
4
5 turtle.left(20)
6 turtle.forward(50)
```

```
7 fiona.forward(70)
```

Explanation:

The variable named fiona is missing a capital letter somewhere

8. Debug the following code:[DEBUG CODE](#)**Incorrect Code:**

```
1 import turtle
2 turtle.getscreen()
3 sam = turtle_Turtle()
4
5 turtle.left(30)
6 turtle.forward(80)
7 sam.forward(10)
```

Correct Solution:

```
1 import turtle
2 turtle.getscreen()
3 sam = turtle.Turtle()
4
5 turtle.left(30)
6 turtle.forward(80)
7 sam.forward(10)
```

9. Debug the following code:[DEBUG CODE](#)**Incorrect Code:**

```
1 import turtle
2 turtle.getscreen()
3 awesome_turtle = turtle.Turtle(
```

Correct Solution:

```
1 import turtle
2 turtle.getscreen()
3 awesome_turtle = turtle.Turtle()
```

Explanation:

This code is missing a closing parenthesis

10. Debug the following code:[DEBUG CODE](#)

Incorrect Code:

```
1 import turtle
2 turtle.getscreen()
3 awesome_turtle = turtle.Turtle()
4
5 awesome_turtle.right(90)
6 awesome_turtle.forward(100
```

Correct Solution:

```
1 import turtle
2 turtle.getscreen()
3 awesome_turtle = turtle.Turtle()
4
5 awesome_turtle.right(90)
6 awesome_turtle.forward(100)
```

Explanation:

This code is missing a parenthesis

Challenges

1. Scooter the Turtle

Solution:

```
1 import turtle
2 turtle.getscreen()
3 scooter = turtle.Turtle()
```

2. Bigfoot the Tortoise

Solution:

```
1 import turtle
2 turtle.getscreen()
3 bigfoot = turtle.Turtle()
```

3. Really Long Turtle Name

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 shelly_sue_sammie_speedy_stripy_sister = turtle.Turtle()
```

4. Name Your Own Pet

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 penelope = turtle.Turtle()
```

5. Two Turtles

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 blue_dude = turtle.Turtle()
5 greenie = turtle.Turtle()
```