

Inputs and Outputs with Python Turtles

Textbook

Inputs and Outputs with Python Turtles



What is an Input?

Let's say we wanted to get input from our user. Some examples of inputs we give:

- Telling the TV which show we want to watch. The show is the input.
- Telling the fast food kiosk which hamburger we want. The choice is the input.
- Entering a username or password. These are inputs.

Inputs from the user are an essential part of making a useful program.

Creating an Input in Python

Inputs are made with an `input()` statement.

We assign the input statement to a variable. Using a variable allows us to easily use the value from the user later in our program. In this example we assigned it to the variable named `response`.

```
1 import turtle
2 turtle.getscreen()
3
4 response = input("What is your name?")
```

Now, whatever the user types is a value we stored in the variable named `response`.

The question `What is your name?` will appear in the [console](#) (the black box area under your turtle screen).

To test it out when you run your code, you will need to click over on the black area and type in your response to input. Press the **enter** key to submit your input and continue your program.

String

In the input statement above, the text `"What is your name?"` is surrounded by quotation marks. Words or statements are a kind of data type in Python called a [string](#). They need to be surrounded by quotation marks.

The value inputted by the user is always accepted as a string as well.

Printing the Input

To see the response the user entered, we can print it by using a `print()` statement. This print statement generates an [output](#).

To print the input, place the variable inside the print statement. In the following example, the input is assigned to a variable named `response`.

```
1 import turtle
2 turtle.getscreen()
3
4 response = input("What is your name?")
5 print(response)
```

This print statement shows in the [console](#). The console is kind of like a notebook for programmers. The user doesn't usually see it, so it's just information for you, the programmer to see.

Sequencing

[Sequencing](#) is the order that code appears in the code editor. So if you ask for an input *before* importing turtles and starting a screen, you will need to answer the input before the turtle screen will show.

If you put the input statement below the code for turtles, the turtles screen will run first.

Data Collection Technologies

Have you ever wondered how people gather information to make smart decisions or solve problems? They collect data! Collecting data means gathering information, and technology makes this process much easier and more powerful. When we collect data, we also need to view, organize, and analyze it to understand what it tells us.

We use different tools to gather data. For science, probes can measure things like temperature or sound. Handheld devices like phones or tablets are great for surveys or quick notes. Mapping systems (GIS) use GPS to collect location data. Even running a computer program many times can create data from its results.

One simple way your Python programs can collect data is using the `input()` function. When your program uses `input()`, it pauses and asks the user (that's you!) to type something. Whatever the user types and presses Enter is then "collected" by the program as data. For example, if you write `name = input("What's`

`your name? ")` , your program is collecting the user's name as data. This is a direct way for your program to get information from the outside world, just like a scientist uses a probe to get temperature readings. Once data is gathered, technology helps us view and organize it, maybe in tables or graphs. Then, we analyze it to find patterns and draw conclusions. Choosing the right tool for collecting data is important because it helps ensure we get the right information to begin with.

Checkpoint

Inputs and Outputs with Python Turtles

Practice creating inputs and printing outputs to the console!

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Create a variable named `turtle_name` . Assign it to an input that asks `What is the turtle's name?` .
3. Print out the variable named `turtle_name` .

Requirements:

- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Create a variable named `turtle_name` . Assign it to an input that asks `What is the turtle's name?` .
- Print out the variable named `turtle_name` .

Questions (10)

1. How is user input typically received in Python?

MULTIPLE CHOICE

Choose the correct answer:

- A. Using the `print()` statement.
- B. Through the `turtle` module.
- C. By using the `input()` statement.
- D. Via the `import()` statement.

2. Why is it important to assign the result of the `input()` statement to a variable?

MULTIPLE CHOICE

Choose the correct answer:

- A. It enhances the appearance of the console output.
- B. It allows the program to store and use the user's input later.
- C. It prevents errors in the code.
- D. It improves the speed of code execution.

3. What data type are inputs automatically accepted as?

Choose the correct answer:

- A. string
- B. integer
- C. float
- D. boolean

4. How can you display the user's input in the console?

Choose the correct answer:

- A. Using the turtle module.
- B. With the import statement.
- C. By using the print() statement.
- D. Through the input() statement.

5. What role does sequencing play in Python programming?

Choose the correct answer:

- A. It determines the order of code execution.
- B. It imports external libraries.
- C. It controls the appearance of the console.
- D. It defines the type of data.

6. In the context of the turtle module, what is the console?

Choose the correct answer:

- A. The area where the turtle moves.
- B. The black box under the turtle screen.
- C. A separate window for user input.
- D. The area where the code is entered.

7. Why is it important to consider sequencing when working with user input and turtles?

MULTIPLE CHOICE

Choose the correct answer:

- A. It determines the order in which code is executed.
- B. It prevents errors in the console.
- C. It influences the color of the code editor.
- D. It affects the appearance of the turtle screen.

8. Debug the following code:

DEBUG CODE

Code to Debug:

```
1 favorite = input(What is your favorite color?)
2
3 print(favorite)
```

9. Debug the following code:

DEBUG CODE

Code to Debug:

```
1 favorite = input("What is your favorite color?")
2
3 print(favrite)
```

10. Debug the following code:

DEBUG CODE

Code to Debug:

```
1 favorite = input("What is your favorite color?")
2
3 print = (favrite)
```

Challenges (5)

1. Username and Password

Create a program that asks the user for a username and password. The program will then print the username and password to the console.

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Create a variable named `username` . Assign it to an input that says `Enter username` .
3. Create a variable named `password` . Assign it to an input that says `Enter password` .
4. On a separate line of code, print the variable named `username` .
5. On a separate line of code, print the variable named `password` .

Requirements:

- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Create a variable named `username` . Assign it to an input that says `Enter username` .
- Create a variable named `password` . Assign it to an input that says `Enter password` .
- On a separate line of code, print the variable named `username` .
- On a separate line of code, print the variable named `password` .

2. Breakfast, Lunch, and Dinner

What did you eat today? Create a program where you can answer that question!

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Create a variable named `breakfast` . Assign it to an input that says `What did you have for breakfast?` .
3. Create a variable named `lunch` . Assign it to an input that says `What did you have for lunch?` .
4. Create a variable named `dinner` . Assign it to an input that says `What did you have for dinner?` .
5. On a separate line of code, print the variable named `breakfast` .
6. On a separate line of code, print the variable named `lunch` .
7. On a separate line of code, print the variable named `dinner` .

Requirements:

- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Create a variable named `breakfast` . Assign it to an input that says `What did you have for breakfast?` .
- Create a variable named `lunch` . Assign it to an input that says `What did you have for lunch?` .
- Create a variable named `dinner` . Assign it to an input that says `What did you have for dinner?` .
- On a separate line of code, print the variable named `breakfast` .
- On a separate line of code, print the variable named `lunch` .
- On a separate line of code, print the variable named `dinner` .

3. How are You Feeling?

Create a program that asks the user how they are feeling *before* running the turtle program. Then it will run the turtles program. After the turtles program runs, it will print out the user's response.

1. Create a variable named `emotion` . Assign it to an input that says `How are you feeling today?` .
2. Include the necessary code to start up a python screen (import the library and generate a screen.)
3. Draw a circle with a radius of 20.
4. Print the variable named `emotion` .

Requirements:

- Create a variable named `emotion` . Assign it to an input that says `How are you feeling today?` .
- Include the necessary code to start up a python screen (import the library and generate a screen.)
- Draw a circle with a radius of 20.
- Print the variable named `emotion` .

4. User Custom Title

Create a turtles program where the user can decide what the title of the turtles screen is!

1. Create a variable named `response` . Assign it to an input that says `Enter Title` .
2. Include the necessary code to start up a Python screen (import the library and generate a screen).
3. Add the code to determine the title of the turtle screen. Use the variable named `response` .
4. Move the turtle forward 100.

Requirements:

- Create a variable named `response` . Assign it to an input that says `Enter Title` .
- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Add the code to determine the title of the turtle screen. Use the variable named `response` .
- Move the turtle forward 100.

5. Write on the Turtle Screen

Did you know that your turtle can write? Your turtle can write on the screen using the following command:

```
turtle.write( )
```

You can either put a string directly into the command, or add a variable in. For this challenge, get an input from the user, assign it to a variable, and write the variable on the screen.

1. Create a variable named `name` . Assign it to an input that says `Enter Turtle Name` .
2. Include the necessary code to start up a Python screen (import the library and generate a screen).
3. Use the `turtle.write()` command. Place the variable named `name` in the parentheses to print the user input.

Requirements:

- Create a variable named `name` . Assign it to an input that says `Enter Turtle Name` .
- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Use the `turtle.write()` command. Place the variable named `name` in the parentheses to print the user input.

Answer Keys & Solutions

Checkpoint Solutions

Inputs and Outputs with Python Turtles

```
1 turtle_name = input("What is the turtle's name?")
2
3 import turtle
4 turtle.getscreen()
5
6 print(turtle_name)
```

Questions

1. How is user input typically received in Python?

MULTIPLE CHOICE

Correct Answer:

- A. Using the print() statement. ✗ Incorrect
- B. Through the turtle module. ✗ Incorrect
- C. By using the input() statement. ✓ Correct
- D. Via the import() statement. ✗ Incorrect

Explanation:

The input statement collects the information from the user.

2. Why is it important to assign the result of the input() statement to a variable?

MULTIPLE CHOICE

Correct Answer:

- A. It enhances the appearance of the console output. ✗ Incorrect
- B. It allows the program to store and use the user's input later. ✓ Correct
- C. It prevents errors in the code. ✗ Incorrect
- D. It improves the speed of code execution. ✗ Incorrect

Explanation:

Assigning the value to a variable makes it easier to use in other places in the program.

3. What data type are inputs automatically accepted as?

MULTIPLE CHOICE

Correct Answer:

- | | |
|------------|-------------|
| A. string | ✓ Correct |
| B. integer | ✗ Incorrect |
| C. float | ✗ Incorrect |
| D. boolean | ✗ Incorrect |

Explanation:

Inputs are always accepted as this data type. "here's an example"

4. How can you display the user's input in the console?

MULTIPLE CHOICE

Correct Answer:

- | | |
|------------------------------------|-------------|
| A. Using the turtle module. | ✗ Incorrect |
| B. With the import statement. | ✗ Incorrect |
| C. By using the print() statement. | ✓ Correct |
| D. Through the input() statement. | ✗ Incorrect |

Explanation:

This process is called printing to the console.

5. What role does sequencing play in Python programming?

MULTIPLE CHOICE

Correct Answer:

- | | |
|---|-------------|
| A. It determines the order of code execution. | ✓ Correct |
| B. It imports external libraries. | ✗ Incorrect |

C. It controls the appearance of the console.

✗ Incorrect

D. It defines the type of data.

✗ Incorrect

Explanation:

Sequencing is the order that commands appear in the code

6. In the context of the turtle module, what is the console?

MULTIPLE CHOICE

Correct Answer:

A. The area where the turtle moves.

✗ Incorrect

B. The black box under the turtle screen.

✓ Correct

C. A separate window for user input.

✗ Incorrect

D. The area where the code is entered.

✗ Incorrect

Explanation:

The console is for the programmer to see, not the user

7. Why is it important to consider sequencing when working with user input and turtles?

MULTIPLE CHOICE

Correct Answer:

A. It determines the order in which code is executed.

✓ Correct

B. It prevents errors in the console.

✗ Incorrect

C. It influences the color of the code editor.

✗ Incorrect

D. It affects the appearance of the turtle screen.

✗ Incorrect

Explanation:

Sequencing is the order that commands appear in the code

8. Debug the following code:

DEBUG CODE

Incorrect Code:

```
1 favorite = input(What is your favorite color?)
2
3 print(favorite)
```

Correct Solution:

```
1 favorite = input("What is your favorite color?")
2
3 print(favorite)
```

Explanation:

This code is missing quotation marks

9. Debug the following code:

DEBUG CODE

Incorrect Code:

```
1 favorite = input("What is your favorite color?")
2
3 print(favrite)
```

Correct Solution:

```
1 favorite = input("What is your favorite color?")
2
3 print(favorite)
```

10. Debug the following code:

DEBUG CODE

Incorrect Code:

```
1 favorite = input("What is your favorite color?")
2
3 print = (favrite)
```

Correct Solution:

```
1 favorite = input("What is your favorite color?")
2
3 print(favorite)
```

Explanation:

This code is missing a parentheses

Challenges

1. Username and Password

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 username = input("Enter username")
5 password = input("Enter password")
6
7 print(username)
8 print(password)
```

2. Breakfast, Lunch, and Dinner

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 breakfast = input("What did you have for breakfast?")
5 lunch = input("What did you have for lunch?")
6 dinner = input("What did you have for dinner?")
7
8 print(breakfast)
9 print(lunch)
10 print(dinner)
```

3. How are You Feeling?

Solution:

```
1 emotion = input("How are you feeling today?")
2
3 import turtle
4 turtle.getscreen()
5
6 turtle.circle(20)
7
8
9 print(emotion)
```

4. User Custom Title

Solution:

```
1 response = input("Enter Title")
2
3 import turtle
```

```
4 turtle.getscreen()
5 turtle.title(response)
6
7 turtle.forward(100)
```

5. Write on the Turtle Screen

Solution:

```
1 name = input("Enter Turtle Name")
2
3 import turtle
4 turtle.getscreen()
5
6
7 turtle.write(name)
```