

While Loops with Python Turtles

Textbook

While Loops with Python Turtles



The [while loop](#) runs a chunk of code while a condition is met. If the condition is no longer met, then the code chunk will stop running.

```
1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter + 1
9
```

Let's explore what's happening in this code.

1. We created a variable named `counter` and assigned it to a value of `1`.

2. We created a `while loop` that will run as long as the variable named `counter` is less than `5`.
3. Inside the while loop, we move the turtle forward 50 and then rotate left 90 degrees.
4. Inside the while loop, we update the variable named `counter` by adding 1 to it. So each time the loop runs, the variable named `counter` increases by 1.
5. Since the variable named `counter` increases by 1 each time the loop runs, it turns into 2, then 3, then 4, then 5. This means the loop will run a total of 4 times. Once the loop runs 4 times, the value of `counter` is 5, which does not meet the condition of `counter < 5`.

Reassigning Variables

As you can see in this code, it's possible to update and reassign variables in your code. Just like a box, you can replace whatever is inside a variable with new items or values.

```
1 import turtle
2 turtle.getscreen()
3
4 my_variable = 2
5
6 my_variable = 5
7
8 print(my_variable)
```

This will print out 5 to the console because the variable has been reassigned to the value of 5.

CAUTION! Watch Out for Infinite Loops!



You need to be very careful with using while loops. If you don't include a way for the condition to stop being met, the code will just run and run without an exit. **The programmer must include a way to quit the loop.**

```

1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter + 1

```

For example, in the above code segment, if we didn't increase the value of the variable named `counter` each time the loop runs, the condition `counter < 5` will always be met, forever. This causes an infinite loop which crashes your program.

These are done with adding or subtracting from a variable INSIDE the while loop. If you don't, your code will continue to run forever which is not good for your computer and leads to program crashes.

We will learn some ways to make sure the variable inside the while loop updates through incrementing and decrementing.

Increment Operator

Let's discuss increment operators with the same code example.

```

1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter+1

```

See the code line that looks like this?

```
counter = counter + 1
```

Every time the loop runs, the variable named `counter` adds 1 to it. This means that the variable named `counter` becomes a larger and larger integer until the condition is met (`counter < 5`). Once the condition is met, the loop will quit. **This stops an infinite loop from happening.**

You can write an incrementor in two ways. Both are correct.

```
counter = counter + 1
```

```
counter += 1
```

Decrement Operator

A decrement operator works the same way as an increment operator, except it subtracts values.

```
counter = counter - 5
```

```
counter -= 5
```

How to Stop an Infinite Loop



If an infinite loop happens, don't worry! You will need to quit your web browser and open a new window. Carefully add an incrementor or decrementor from a variable inside your loop before running the loop again. Think carefully about how to make sure the loop has a specified time and place to stop running.

Remember that using addition or subtraction inside the loop will tell your while loop when to stop.

If you don't tell it to stop in your program, it won't.

Break

Sometimes, you may want to stop a while loop before the condition is no longer met. You can do this by using the `break` statement. When the `break` is reached inside the loop, it immediately stops the loop, no matter what the condition is.

```
1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 10:
6     turtle.forward(50)
7     turtle.left(90)
8     counter += 1
9     if counter == 5:
10         break
11
```

Try it!

In this example, the loop will stop when `counter` reaches 5, even though the original condition was for the loop to run until `counter` is less than 10. The `break` lets you control exactly when the loop stops.

Checkpoint

While Loops with Python Turtles

Draw a square in Python Turtles using a while loop.

1. Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
2. Create a variable named `counter` and assign it to 1.
3. Create a while loop that checks to see if the variable named `counter` is less than 5.
4. Inside the while loop, move the turtle forward 50 and rotate to the left 90.
5. Inside the while loop, increase the variable named `counter` by 1.

Requirements:

- Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
- Create a variable named `counter` and assign it to 1.
- Create a while loop that checks to see if the variable named `counter` is less than 5.
- Inside the while loop, move the turtle forward 50 and rotate to the left 90.
- Inside the while loop, increase the variable named `counter` by 1.

Questions (10)

1. What is the purpose of the while loop in the Python Turtles code?

MULTIPLE CHOICE

Choose the correct answer:

- A. To repeat a code chunk as long as a condition is met.
- B. To repeat a code chunk only 2 times.
- C. To speed up code.
- D. To create an infinite loop.

2. In the following code, what does the variable "counter" represent?

MULTIPLE CHOICE

```
import turtle
turtle.getscreen()
counter = 1
while counter < 5:
    turtle.forward(50)
    turtle.left(90)
    counter = counter + 1
```

Choose the correct answer:

- A. The number of degrees the turtle rotates.
- B. The distance the turtle moves forward.
- C. The variable that the while loop is checking during it's conditional statement.
- D. The radius of the turtle's movement.

3. What happens if the condition in the while loop is never met?**Choose the correct answer:**

- A. The code inside the loop will never run.
- B. The code inside the loop will run indefinitely.
- C. The loop will terminate immediately.
- D. The turtle will not be visible on the screen.

4. How does the variable "counter" change inside the while loop?

```
import turtle
turtle.getscreen()
counter = 1
while counter < 5:
    turtle.forward(50)
    turtle.left(90)
    counter = counter + 2
```

Choose the correct answer:

- A. It is multiplied by 2 each time.
- B. It is divided by 2 each time.
- C. It is incremented by 2 each time.
- D. It is decremented by 2 each time.

5. What can happen if you don't include a way for the condition in the while loop to stop being met?**Choose the correct answer:**

- A. The code will run faster.
- B. The turtle will move in a straight line.
- C. An infinite loop will occur.
- D. The program will display an error.

6. Which of the following is a correct example of a decrement operator? Select 2.**Select all that apply:**

- A. `counter = counter - 1`
- B. `counter -= 1`
- C. `counter += 1`
- D. `counter = counter + 1`

7. What is the consequence of an infinite loop in programming?

Choose the correct answer:

- A. The program will run faster.
- B. The program will terminate immediately.
- C. The program will just keep running and running, which isn't good for the computer.
- D. The turtle will move in a straight line.

8. How can you stop an infinite loop in your program?

Choose the correct answer:

- A. Quit the web browser and open a new window.
- B. Delete the entire code.
- C. Panic.
- D. Increase the screen size.

9. How do you remove an infinite loop from code?

Choose the correct answer:

- A. Carefully add an incrementor or decrementor inside the loop.
- B. Delete the entire code.
- C. Add a variable.
- D. Only use for loops.

10. Debug the following code:

Code to Debug:

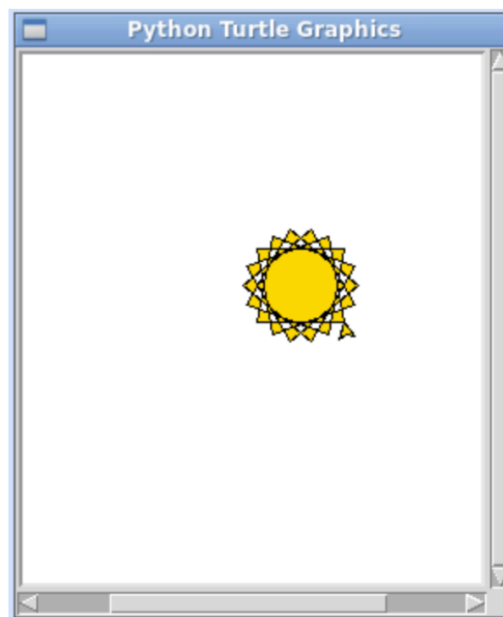
```
1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter+10
```

Challenges (5)

1. Sun

Draw a sun with a while loop.

1. Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
2. Use `begin_fill()` and `fillcolor()` . Make the sun "gold" . Place the `end_fill` command after the loop finishes.
3. Create a variable named `counter` and assign it to 1.
4. Create a while loop that checks to see if the variable named `counter` is less than 20.
5. Inside the while loop, move the turtle forward 50 and rotate to the left 100.
6. Inside the while loop, increase the variable named `counter` by 1.



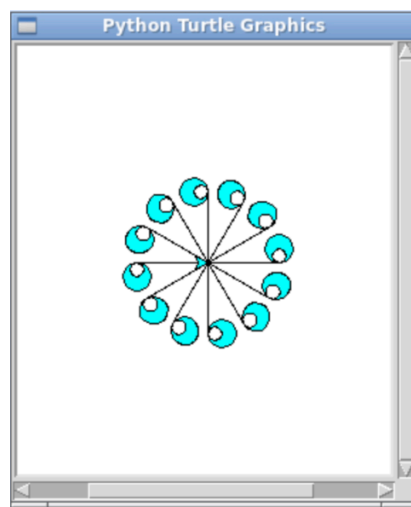
Requirements:

- Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
- Use `fillcolor()` and `begin_fill()` . Make the sun "gold" . Place the `end_fill` command after the loop finishes.
- Create a variable named `counter` and assign it to 1.
- Create a while loop that checks to see if the variable named `counter` is less than 20.
- Inside the while loop, move the turtle forward 50 and rotate to the left 100.
- Inside the while loop, increase the variable named `counter` by 1.

2. Peacock

Draw some peacock feathers using while loops in Python Turtles!

1. Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
2. Use `fillcolor()` and `begin_fill()` . Make the shape "aqua" . Place the end_fill command after the loop finishes.
3. Create a variable named `counter` and assign it to 1.
4. Create a while loop that checks to see if the variable named `counter` is less than 13.
5. Inside the while loop, move the turtle forward 50, draw a circle with a radius of 10, draw another circle with a radius of 5, move backward 50, and rotate to the left 30.
6. Inside the while loop, increase the variable named `counter` by 1.



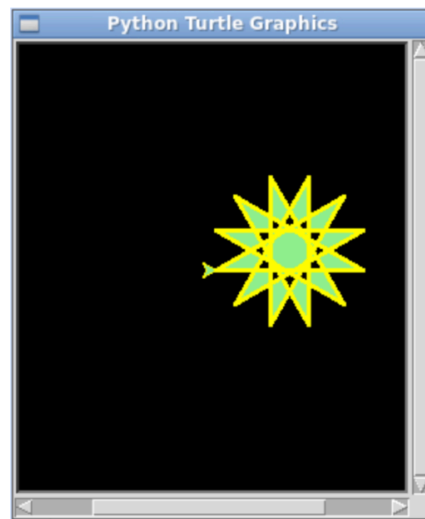
Requirements:

- Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
- 1. Use `fillcolor()` and `begin_fill()` . Make the shape "aqua" . Place the end_fill command after the loop finishes.
- Create a variable named `counter` and assign it to 1.
- Create a while loop that checks to see if the variable named `counter` is less than 13.
- Inside the while loop, move the turtle forward 50, draw a circle with a radius of 10, draw another circle with a radius of 5, move backward 50, and rotate to the left 30.
- Inside the while loop, increase the variable named `counter` by 1.

3. Decrement While Loop

Try creating a while loop that uses a decrementor.

1. Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
2. Use `fillcolor()` and `begin_fill()` . Make the shape "light green" . Place the `end_fill` command after the loop finishes.
3. Set the background color to `black` . Use `bgcolor()` .
4. Set the pen color to `yellow` . Use `pencolor()` .
5. Set the pen size to 3. Use `pensize()` .
6. Create a variable named `counter` and assign it to 13.
7. Create a while loop that checks to see if the variable named `counter` is greater than 1.
8. Inside the while loop, move the turtle forward 100 and rotate to the left 150.
9. Inside the while loop, DECREASE the variable named `counter` by 1.



Requirements:

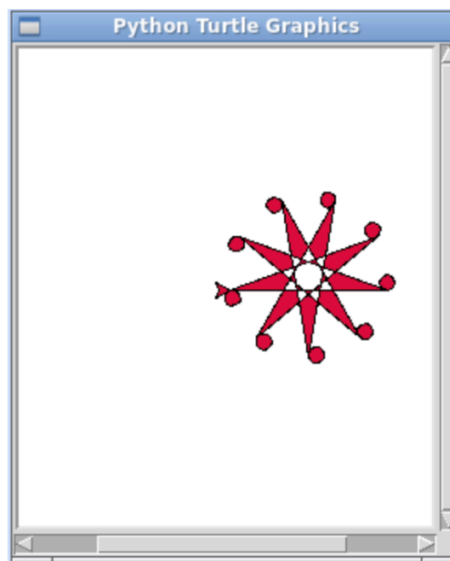
- Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
- 1. Use `fillcolor()` and `begin_fill()` . Make the shape "light green" . Place the `end_fill` command after the loop finishes.
- Set the background color to `black` . Use `bgcolor()` .
- Set the pen color to `yellow` . Use `pencolor()` .
- Set the pen size to 3. Use `pensize()` .
- Create a variable named `counter` and assign it to 13.
- Create a while loop that checks to see if the variable named `counter` is greater than 1.
- Inside the while loop, move the turtle forward 100 and rotate to the left 150.

- Inside the while loop, DECREASE the variable named `counter` by 1.

4. Ferris Wheel

Draw a ferris wheel using Python Turtles!

1. Include the necessary code to start up a python screen (import the library and generate a screen.)
2. Use `fillcolor()` and `begin_fill()` . Make the shape "crimson" . Place the `end_fill` command after the loop finishes.
3. Create a variable named `counter` and assign it to 1.
4. Create a while loop that checks to see if the variable named `counter` is less than 10.
5. Inside the while loop, move the turtle forward 100, draw a circle with a radius of 5, and rotate to the right 200.
6. Inside the while loop, increase the variable named `counter` by 1.



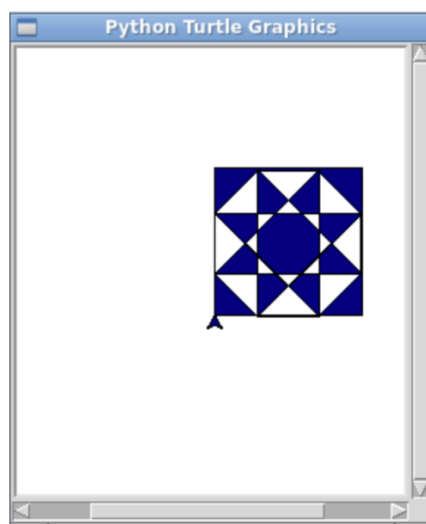
Requirements:

- Include the necessary code to start up a python screen (import the library and generate a screen.)
- Use `fillcolor()` and `begin_fill()` . Make the shape "crimson" . Place the `end_fill` command after the loop finishes.
- Create a variable named `counter` and assign it to 1.
- Create a while loop that checks to see if the variable named `counter` is less than 10.
- Inside the while loop, move the turtle forward 100, draw a circle with a radius of 5, and rotate to the right 200.
- Inside the while loop, increase the variable named `counter` by 1.

5. Quilting Square

Draw a quilting square using Python Turtles! Use 3 separate loops.

1. Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
2. Use `fillcolor()` and `begin_fill()` . Color the shape "navy" . Place the `end_fill` command after the loop finishes.
3. Create a 3 variables. Name them `counter` , `second_counter` , and `third_counter` . Assign each to 1.
4. Create 3 while loops. The first will check to see if the variable named `counter` is less than 9. The second will check to see if the variable named `second_counter` is less than 9. The third will check to see if the variable named `third_counter` is less than 5.
5. Inside the first while loop, move the turtle forward 100 and rotate to the left 135.
6. Inside the second while loop, move the turtle forward 41 and rotate to the right 45.
7. Inside the third while loop, move the turtle forward 100 and rotate to the right 90.
8. Inside each loop, increase the variable named `counter` , `second_counter` , or `third_counter` by 1.
9. After the first loop, rotate the turtle to the left 90 degrees. (Outside the loops. Check indentation.)
10. After the second loop, move the turtle backward 28. (Outside the loops. Check indentation.)



Requirements:

- Include the necessary code to start up a Python screen. (Import the library and generate a screen.)
- Use `fillcolor()` and `begin_fill()` . Make the shape "navy" . Place the `end_fill` command after the loop finishes.
- Create a 3 variables. Name them `counter` , `second_counter` , and `third_counter` . Assign each to 1.
- Create 3 while loops. The first will check to see if the variable named `counter` is less than 9. The second will check to see if the variable named `second_counter` is less than 9. The third will check to see if the variable named `third_counter` is less than 5.
- Inside the first while loop, move the turtle forward 100 and rotate to the left 135.
- Inside the second while loop, move the turtle forward 41 and rotate to the right 45.

- Inside the third while loop, move the turtle forward 100 and rotate to the right 90.
- Inside each loop, increase the variable named `counter` , `second_counter` , or `third_counter` by 1.
- After the first loop, rotate the turtle to the left 90 degrees. (Outside the loops. Check indentation.)
- After the second loop, move the turtle backward 28. (Outside the loops. Check indentation.)

Answer Keys & Solutions

Checkpoint Solutions

While Loops with Python Turtles

```
1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter + 1
```

Questions

1. What is the purpose of the while loop in the Python Turtles code?

MULTIPLE CHOICE

Correct Answer:

- A. To repeat a code chunk as long as a condition is met. ✓ Correct
- B. To repeat a code chunk only 2 times. ✗ Incorrect
- C. To speed up code. ✗ Incorrect
- D. To create an infinite loop. ✗ Incorrect

Explanation:

While loops run as long as a certain condition is met.

2. In the following code, what does the variable "counter" represent?

MULTIPLE CHOICE

Correct Answer:

- A. The number of degrees the turtle rotates. ✗ Incorrect
- B. The distance the turtle moves forward. ✗ Incorrect
- C. The variable that the while loop is checking during it's conditional statement. ✓ Correct

D. The radius of the turtle's movement.

✗ Incorrect

Explanation:

The counter variable updates each time the loop runs

3. What happens if the condition in the while loop is never met?

MULTIPLE CHOICE

Correct Answer:

A. The code inside the loop will never run.

✓ Correct

B. The code inside the loop will run indefinitely.

✗ Incorrect

C. The loop will terminate immediately.

✗ Incorrect

D. The turtle will not be visible on the screen.

✗ Incorrect

Explanation:

A while loop runs only while a certain condition is met. If it's not met, it will not run.

4. How does the variable "counter" change inside the while loop?

MULTIPLE CHOICE

Correct Answer:

A. It is multiplied by 2 each time.

✗ Incorrect

B. It is divided by 2 each time.

✗ Incorrect

C. It is incremented by 2 each time.

✓ Correct

D. It is decremented by 2 each time.

✗ Incorrect

Explanation:

Increment means increasing

5. What can happen if you don't include a way for the condition in the while loop to stop being met?

MULTIPLE CHOICE

Correct Answer:

A. The code will run faster.

✗ Incorrect

B. The turtle will move in a straight line.

✗ Incorrect

C. An infinite loop will occur.

✓ Correct

D. The program will display an error.

✗ Incorrect

Explanation:

If a while loop doesn't have a way to stop the condition from being met, it will just keep running.

6. Which of the following is a correct example of a decrement operator? Select 2.

SELECT MULTIPLE

Correct Answers:

A. `counter = counter - 1`

✓ Correct

B. `counter -= 1`

✓ Correct

C. `counter += 1`

✗ Incorrect

D. `counter = counter + 1`

✗ Incorrect

Explanation:

Decrement means decreasing

7. What is the consequence of an infinite loop in programming?

MULTIPLE CHOICE

Correct Answer:

A. The program will run faster.

✗ Incorrect

B. The program will terminate immediately.

✗ Incorrect

C. The program will just keep running and running, which isn't good for the computer.

✓ Correct

D. The turtle will move in a straight line.

✗ Incorrect

Explanation:

The loop won't stop.

8. How can you stop an infinite loop in your program?

Correct Answer:

- A. Quit the web browser and open a new window. ✓ Correct
- B. Delete the entire code. ✗ Incorrect
- C. Panic. ✗ Incorrect
- D. Increase the screen size. ✗ Incorrect

Explanation:

Restarting the web browser will stop the infinite loop

9. How do you remove an infinite loop from code?

Correct Answer:

- A. Carefully add an incrementor or decrementor inside the loop. ✓ Correct
- B. Delete the entire code. ✗ Incorrect
- C. Add a variable. ✗ Incorrect
- D. Only use for loops. ✗ Incorrect

Explanation:

While loops need an incrementor or a decrementor to work correctly.

10. Debug the following code:

Incorrect Code:

```
1 import turtle
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter+10
```

Correct Solution:

```
1 import turtle
```

```
2 turtle.getscreen()
3
4 counter = 1
5 while counter < 5:
6     turtle.forward(50)
7     turtle.left(90)
8     counter = counter+10
```

Explanation:

There's a spelling mistake in this code.

Challenges

1. Sun

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4
5 turtle.fillcolor("gold")
6 turtle.begin_fill()
7
8
9 counter = 1
10 while counter < 20:
11     turtle.forward(50)
12     turtle.left(100)
13     counter = counter + 1
14
15 turtle.end_fill()
```

2. Peacock

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.fillcolor("aqua")
5 turtle.begin_fill()
6
7 counter = 1
8 while counter < 13:
9     turtle.forward(50)
10    turtle.circle(10)
11    turtle.circle(5)
12    turtle.backward(50)
13    turtle.left(30)
14    counter = counter + 1
15
16 turtle.end_fill()
```

3. Decrement While Loop

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.bgcolor("black")
5 turtle.pencolor("yellow")
6 turtle.pensize(3)
7
8 turtle.fillcolor("light green")
9 turtle.begin_fill()
10
11 counter = 13
12 while counter > 1:
13     turtle.forward(100)
14     turtle.left(150)
15     counter = counter - 1
16
17 turtle.end_fill()
```

4. Ferris Wheel

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.fillcolor("crimson")
5 turtle.begin_fill()
6
7 counter = 1
8 while counter < 10:
9     turtle.forward(100)
10    turtle.circle(5)
11    turtle.right(200)
12    counter = counter + 1
13
14 turtle.end_fill()
```

5. Quilting Square

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.fillcolor("navy")
5 turtle.begin_fill()
```

```
6
7 counter = 1
8 while counter < 9:
9     turtle.forward(100)
10    turtle.left(135)
11    counter = counter + 1
12
13 turtle.left(90)
14
15 second_counter = 1
16 while second_counter < 9:
17     turtle.forward(41)
18     turtle.right(45)
19     second_counter = second_counter + 1
20
21 turtle.backward(28)
22
23 third_counter = 1
24 while third_counter < 5:
25     turtle.forward(100)
26     turtle.right(90)
27     third_counter = third_counter + 1
28
29 turtle.end_fill()
```