Jump Statements

There a few keywords we can use to add further control flow to our loops. Typically, they work with a series of *nested loops*, where one loop is written entirely within the body of another loop. These keywords are often used to limit while loops and prevent them from creating infinite loops.

BREAK

At any point within a loop block, you can end it by using the break keyword.

```
while (playerIsAlive)
{
// this code will keep running

if (playerIsAlive == false)
{
    // eventually if this stopping condition is true,
    // it will break out of the while loop
    break;
}
// rest of the program will continue
```

You've already seen the **break** keyword—it's the same keyword that is used in switch statements.

CONTINUE

The continue keyword is used to bypass portions of code. It will ignore whatever comes after it in the loop and then will go back to the top and start the loop again.

```
int bats = 10;
for (int i = 0; i <= 10; i++)
{
   if (i < 9)
   {
      continue;
   }</pre>
```

```
// this will be skipped until i is no longer less than 9
Console WriteLine(i);
}
```

Here, the program starts in the for loop, then hits the if statement. Since there is a continue in the if statement, it will bypass the Console.WriteLine() statement until the condition in the if statement is no longer true. So while the loop starts at 0, nothing will print to the console until i is equal to 9.

RETURN

The return keyword is another way to exit a loop, specifically loops that are used within a method. When a return is used within such a loop, it breaks out of the loop and returns control to the point in the program where the method was called.

```
UnlockDoor
bool doorIsLocked
while (doorIsLocked
  // eventually if this stopping condition is true,
  // it will break out of the while loop
   // this return statement will break out of the entire method
   return true
```

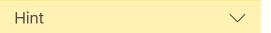
You should only use return if you need to exit a method, because it will break out of allloops. If you only want to break out of one loop and not exit a method, use break.

✓Instructions

1.

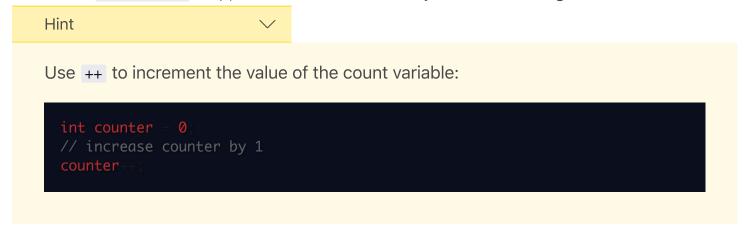
You've decided to go back to the pomodoro application. This time, you don't want the alarm to ring endlessly. If it rings more than 3 times, it should shut off.

Create a variable that will keep track of how many times the alarm has gone off.



Create an int variable to hold the count value. Start it equal to zero, since the alarm has not gone off yet.

2. Inside the do...while loop, increase the count every time the alarm goes off.



3. The program should break out of the loop if the count variable reaches three.

Write a statement that checks if the count variable has reached three, and when it does, have it break out of the do...while loop.



Use an if statement to check if the variable is equal to three. Place the break keyword inside the if statement and make sure the if statement is inside of the loop.

```
while (playerIsAlive)
{
    // this code will keep running

    if (playerIsAlive == false)
        // eventually if this stopping condition is true,
        // it will break out of the while loop
        break;
}
// rest of the program will continue
```