# Infinite Loops

This lesson explains how infinite loops might arise in a while loop using an example.

**We'll cover the following**



## What are Infinite Loops #

One common programming mistake is to create an **infinite** loop. An **infinite** loop refers to a loop, which under certain valid (or at least plausible) input, will never **exit**.

> **Note:** Beginner programmers" should be careful to examine all the *possible* inputs into a *loop* to ensure that for each such set of inputs, there is an **exit** condition that will eventually be reached.

*Compilers*, *debuggers*, and other *programming tools* can only help the programmer so far in detecting **infinite** loops.

> **Note:** In the sufficiently general case, it is not possible to automatically detect an infinite loop. This is known as the **halting** problem.

While the halting problem is not solvable in the fully general case, it is possible to determine whether a loop will **halt** for some *specific* cases.

### Example of Infinite loop #

Down below is an example of an *infinite* loop.

```
<?php
$a = 1;
// the while condition will always be met as it will always return true
```

```
while ($a)
{
    echo "Infinite loop\n";

}
?>
```

When executed, the code above will print **“Infinite loop”** without stopping because the condition statement in the `while` loop will always resolve to **true**. There is no point at which it’ll return **false** hence we’ll get stuck in an *infinite* loop, and the code will execute forever.

Now that you know all there is to know about the for loop, let’s solve a quiz in the next lesson.