Comparing Loops

You may have noticed that there are lots of similarities between different types of loops, and you're right!

We just showed how we can use a `foreach` loop to iterate through an array. But we can also use a `for` loop to iterate through an array:

```
string[] items = { "potion", "dagger", "shield", "plant" };

for (int i = 0; i < items.Length; i++)
{
  Console.WriteLine(items[i]);
}
```

We could even write a complicated `while` loop that starts a counter at 0, then compares that counter to the length of the `items` array. If the counter is less than the array, the loop will continue. Otherwise, it will stop looping through the statements and the program will move on to the next line of code.

```
int i = 0;
while (i < items.Length)
{
  Console.WriteLine(items[i]);
  i++;
}
```

Since a `foreach` loop does the same thing as the other two but is more concise, it is less prone to errors and the better choice in this circumstance.

```
string[] items = { "potion", "dagger", "shield", "plant" };

foreach (string item in items)
{
  Console.WriteLine(item);
}
```

In general,

`while` loops are good when you know your stopping condition, but not when you know how many times you want a program to loop or if you have a specific collection to loop through.

`do...while` loops are only necessary if you definitely want something to run once, but then stop if a condition is met.

`for` loops are best if you want something to run a specific number of times, rather than given a certain condition.

`foreach` loops are the best way to loop over an array, or any other collection.

☑Instructions

1.

You want to build an app that blocks websites, so you find some code online and copy and paste it into your text editor. You notice that it uses a `while` loop to iterate through a `websites` array, but you know a better way to do that!

Re-write the loop so that it uses a loop that better suits the objective.

Hint

A `foreach` loop is the best choice to loop through a collection. Here's what a `foreach` loop looks like:

```
string[] melody = { "a", "b", "c", "c", "b" };

foreach (string note in melody)
{
  PlayMusic(note);
}
```