# break, continue keywords

**The goal for this exercise** is to understand the **break** and **continue** keywords, so that you can exhibit more complex control over the loops that you create.

There are times when you'll want to end a loop early, or skip the rest of the loop, and go to the next iteration of the loop (without ending the loop). For example, one way to write a loop that will print all even integers less than 300 (starting at 1), and that stops as soon as it finds the first odd number higher than 15, would be as follows:

|  |
| --- |
| int num = 1;  while (num <= 300)  {  // if we've found the first odd #,that's > 15  // exit the loop  if (num > 15 && num % 2 == 1)  break;  // If the number is odd, skip back to the  // while(num <= 300) line  if (num % 2 == 1)  continue;  //otherwise, print the number:  Console.WriteLine(num);  }  Console.WriteLine("All done!");  // (Yes, there are better ways to accomplish this :) ) |

Note the use of the break keyword – if we want to skip the rest of this iteration of the loop, **and stop running the loop altogether**, we use break (such as when we've found an odd number higher than 15 – we're done).

Note the use of the continue keyword – if we want to skip the rest of this iteration of the loop, **but keep running the loop**, we use continue (such as when skipping the even numbers).

You may be tempted to try using a pattern like the following:

|  |
| --- |
| **while (true)**  {  // if we've found the first odd #,that's > 15  **// 🡺 I'm relying on the break keyword to exit the loop 🡸**  if (num > 15 && num % 2 == 1)  break;  // If the number is odd, skip back to the  // while(num <= 300) line  if (num % 2 == 1)  continue;  //otherwise, print the number:  Console.WriteLine(num);  }  Console.WriteLine("All done!"); |

using a pattern like while(true), and then relying on break statements to exit the loop should generally be avoided, since the loop normally would give us (as programmers) valuable information about what the *intention* of the loop is. As a matter of fact, both of the above examples would be better written as follows:

|  |
| --- |
| // if the number is less than 15, keep going  while (num <= 15)  {  // If the number is odd, skip back to the  // while(num <= 15) line  if (num % 2 == 1)  continue;  //otherwise, print the even number:  Console.WriteLine(num);  }  Console.WriteLine("All done!"); |

Side-Note: If you've done the exercise(s) for the switch statement, you'll notice that the switch statement also makes use of the break keyword. In that situation, the meaning is similar - break will make sure that you break out of the switch, rather than going on to the next case. However, you'll end up using switch/break together so often that it may seem like in that case, break is merely part of the syntax of a switch statement (which isn't a bad way to think about switches). As you've seen above, break can also be used in other circumstances, so it's important to keep both purposes in your head – break can break out of a switch (as part of a switch's standard syntax), and break can be used to break out of a loop (in a more general-purpose sense).

**What you need to do for this exercise:**  For this exercise, you should understand the break and continue keywords. You should know which loop the break (or continue) keywords jumps to if it's used inside a nested loop (both if it's used in the inner nested loop, and the outer loop).

You should then fill in the Break\_Continue\_Keywords. RunExercise method (in the provided starter project) so that it will ask the user for 20 numbers, one at a time. As the program receives each number, it should do one of the following actions:

|  |  |
| --- | --- |
| If the number is: | The program should: |
| Less than 10 | Print a warning about it ignoring numbers less than 10, then ignore the number |
| 201, exactly | Print a message to the user thanking them for their time, print out the total of all the numbers they've input so far (including the 201 that the user just entered),and then leave the loop (even if they haven’t already entered 20 numbers) |
| Anything else | Add that number to the running total (the total starts at 0, and is the sum of all the numbers (that are >= 10) that they've entered so far ) |

Within that method, you should use the **break** and **continue** keyword, at least once each (just once is fine). Make sure that you use the keywords in a way that makes sense (i.e., matches the use specified in the above discussion, as opposed to just trying to throw them in somewhere, anywhere, to get credit for using it ☺)