**KIT506 Programming Practical 6: Debugging Loops**

[ [↶ Close all sections](https://mylo.utas.edu.au/content/enforced/214790-AW_EAI_17S2_15468_0_0_0_1_1/kit506_pracs/KIT506_Prac_06.html?d2lSessionVal=1xyxT5ULsXqUuXVAAhs1zIzCY&ou=214790&d2l_body_type=3) ] Only visible sections will be included when printing this document.

**Aims:**

* to debug loop code.

The following examples illustrate some of the many ways that loops can fail to achieve what their creators intended.

**∨ 1 Creating a testing ground**

Create this skeleton program in Visual Studio. For each task below replace all code between the two WriteLines with the given declarations and statements. You may wish to save each program as you go for reference.

**public** **static** void Main() {

Console.WriteLine("Program has started");

*//Replace this with the code to test*

Console.WriteLine("Program about to end");

}

**∨ 2 Sum of the first *n* integers**

Paste the following code into the indicated part of the skeleton program. Compile it, run it, and then answer the questions below.

const int N = 4; *//sum up to this value*

int total = 0; *//stores sum*

*//Calculates the sum of the first N positive integers*

**for** (int i = 1; i <= N; i++)

{

total = i;//total+=i;

}

Console.WriteLine("Sum of the first " + N + " positive integers is " + total);

* What happens when the program runs?

[**Solution**](javascript:void(0);)

It prints the wrong answer. 1 + 2 + 3 + 4 = 10, but it displays 4.

* Where are good places to look for the cause of the error?

[**Solution**](javascript:void(0);)

Try the body first, because the behaviour of the loop is incorrect.

* What’s the problem?

[**Solution**](javascript:void(0);)

The line total = i; replaces the value of total rather than adding to it.

* What’s a suitable correction?

[**Solution**](javascript:void(0);)

Modify the body to be total += i; or total = total + i;.

Implement a correction and rerun to test.

**∨ 3 Countdown timer**

Now test the following loop code in the test program. Compile it, run it, and then answer the questions below.

*//Displays a countdown from 10 to 1, followed by Lift off!*

**for** (int t = 10; t < 0; t--)//t>0 otherwise won’t run

{

Console.WriteLine(t + "!");

}

Console.WriteLine("Lift off!");

* What happens when the program runs?

[**Solution**](javascript:void(0);)

It displays “Lift off!” but nothing else.

* Where are good places to look for the cause of the error? And what is the problem?

[**Solution**](javascript:void(0);)

The test is evaluating to **false** straight away, so the loop is never executed. This means that either we’ve initialised the loop variable to the wrong value

[**Solution**](javascript:void(0);)

No, the comment indicates we want to start it at 10 and that’s what we set it to.

or the test is incorrect.

[**Solution**](javascript:void(0);)

That’s it! The expression t < 0 will never be true because we start t at 10 and the **for**-loop never gets to execute the update.

* What’s a suitable correction?

[**Solution**](javascript:void(0);)

The original author probably meant to write t > 0, so we count down *as long as* t has not yet reached zero.

Implement a correction and rerun to test.

**∨ 4 Word list**

Now test the following loop code in the test program.

const string STOP = "end";

string word;

string list = "";

*//Construct list of words entered by the user, separated by commas*

*//User enters 'end' to stop ('end' is not added to the list)*

Console.Write("Enter a word: ");

word = Console.ReadLine();

**while** (word != STOP)

{

list += word + ",";

Console.Write("Enter a word: ");

// word = Console.ReadLine();

}

Console.WriteLine("Word list is: " + list);

Compile it, run it, and enter the following sequence of words when prompted:  
apple  
loop  
end

* What happens when the program runs?

[**Solution**](javascript:void(0);)

It never stops! Congratulations, you’ve just found the [address of Apple Inc](https://en.wikipedia.org/wiki/Apple_Campus). Close the console window to stop the program.

* Where are good places to look for the cause of the error? And what is the problem?

[**Solution**](javascript:void(0);)

If a loop never ends then it means that either the test can never become **false**

[**Solution**](javascript:void(0);)

No, the user *could* enter the word ‘end’. Try this now: rerun and type *end* when first prompted.

or the update is incorrect.

[**Solution**](javascript:void(0);)

Wait. What update? The value of word never changes. We ask the user for another word but we never actually read it from them!

* What’s a suitable correction?

[**Solution**](javascript:void(0);)

Add word = Console.ReadLine(); inside the loop just after we prompt the user.

Implement a correction and rerun to test.

**∨ 5 For each what?**

Try the following loop code in your test program.

List<string> items = **null**; *//items doesn't refer to anything*

*// null crashes*

*// List<string> items = new List<string>();*

*// List<string> items = new List<string> { "this", "that" };*

**foreach** (string item **in** items)

{

Console.Write("{0} is before ", item); *//replaces {0} with the value of item*

}

Console.WriteLine("the end.");

What happens when you attempt to run it?

[**Solution**](javascript:void(0);)

The program crashes, saying that the List is null. Of course, that’s what we set it to, so this appears to be a contrived error. But, in an application consisting of multiple classes, the List object we work with in one place may have been generated (or, like here, not) in another, so it *may* actually be **null** if the other part of the software didn’t do its job correctly.

Initialise the List with some values, or even just create a **new** empty List, and rerun the code.

[**Solution**](javascript:void(0);)

These illustrate both possible fixes:

List<string> items = **new** List<string>();

results in the program displaying “the end.”, while

List<string> items = **new** List<string> { "this", "that" };

produces the output:

this is before that is before the end.

**Tip:** The **foreach** loop is very robust. As long as the collection exists it will iterate over each element in it (or do nothing, if it’s empty), which eliminates many of the common mistakes that can occur when writing loops to work with collections.