Reading from and Writing to Files

Section 3.14 & 4.16

Files

- Data stored in variables is temporary

- We will learn how to write programs that can
  - Create files
  - Write to files
  - Read from files

Steps to Using Files

- There are five steps that must be taken in order to use files in C++
  1. Include header files
  2. Define a file stream object
     - variable to represent a file
  3. Open the file
  4. Check that the file opened correctly
  5. Use the file
  6. Close the file

1. Header files

- To access files you will need

```
#include <iostream>

#include <fstream>
```

2. File Stream Objects (Variable)

```
ifstream inputFile;

ofstream outputFile;

fstream inAndOut;
```

- One file per variable
- Can open many files at once

3. Opening Files

```
inputFile.open(“filename”)
```

- Same syntax for both input and output files
- Filename is a string literal
- Example:

```
ifstream inputFile;
inputFile.open(“input.txt”);
```

4. Check File Opened Correctly

- Make sure that it opened correctly

```
inputFile.open("input.txt");
if(!inputFile)
{
    cout << "Error opening input file ";
    exit(-1);
}
```

5. Using File Variables

- Use input file variable wherever you use **`cin`**

```
inputFile >> num;
```

- Use output file variable wherever you use **`cout`**

```
outputFile << num;
```

- Can read/write
    - **`double, char, int, string`**

6. Closing Files

- Any files that have been opened must be closed at the end of the program

```
inputFile.close();

outputFile.close();
```

Example: Writing to a File

- Write a program to ask the user for 5 integers and write each integer to the file numbers.txt, each integer on a new line.

- Where is the file?
    - It is in the same directory as your main.cpp

Example: Reading from a file

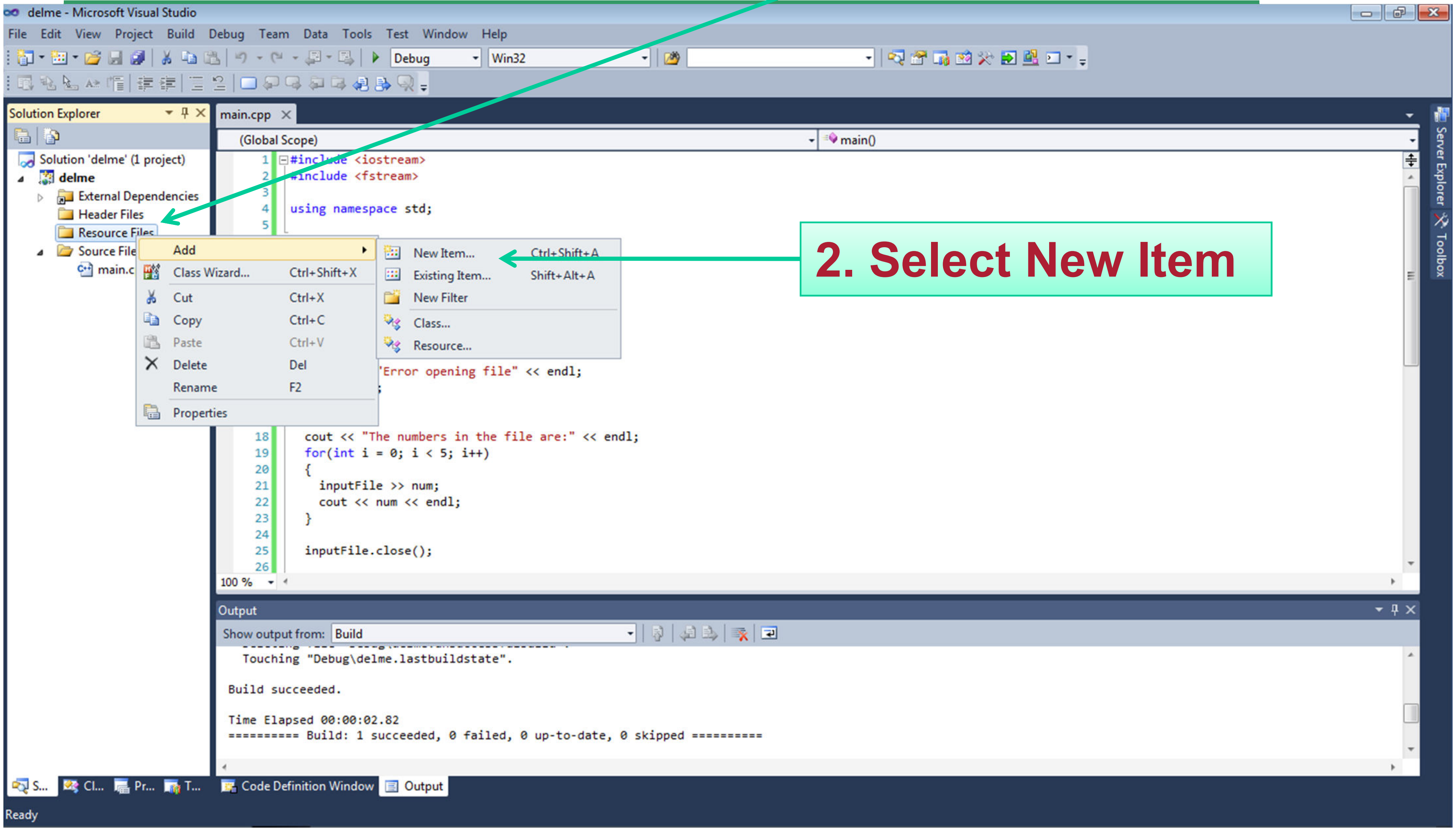
- Write a program to read 5 integers from a file named in.txt and display them to the screen.

- Modify the program to also display the average of the 5 integers.

Create the Input File

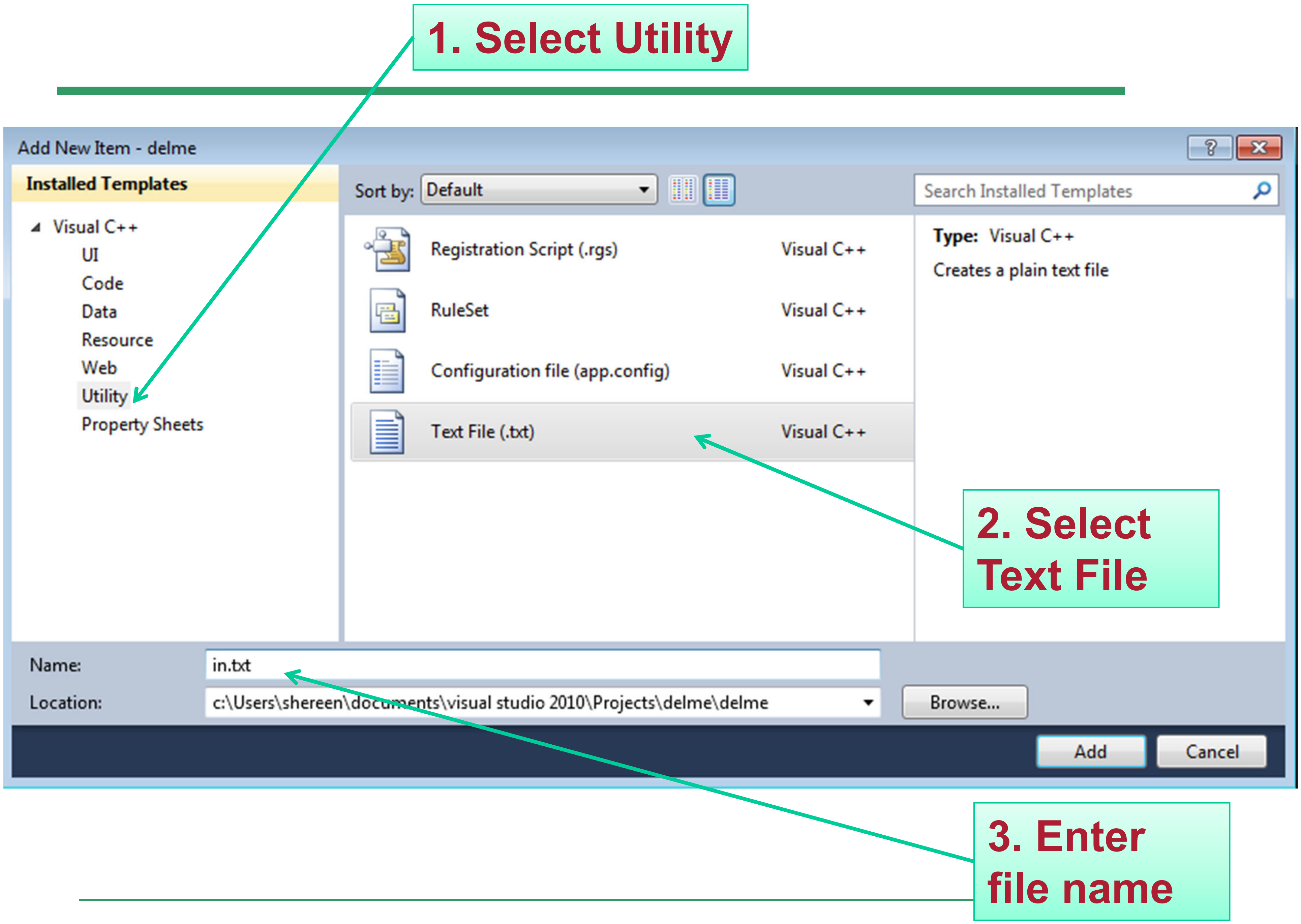
1. Right Click

2. Select New Item



```
#include <iostream>
#include <fstream>

using namespace std;

'Error opening file" << endl;
;

	cout << "The numbers in the file are:" << endl;
	for(int i = 0; i < 5; i++)
	{
	  inputFile >> num;
	  cout << num << endl;
	}

	inputFile.close();
```

```
  Touching "Debug\delme.lastbuildstate".

Build succeeded.

Time Elapsed 00:00:02.82
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



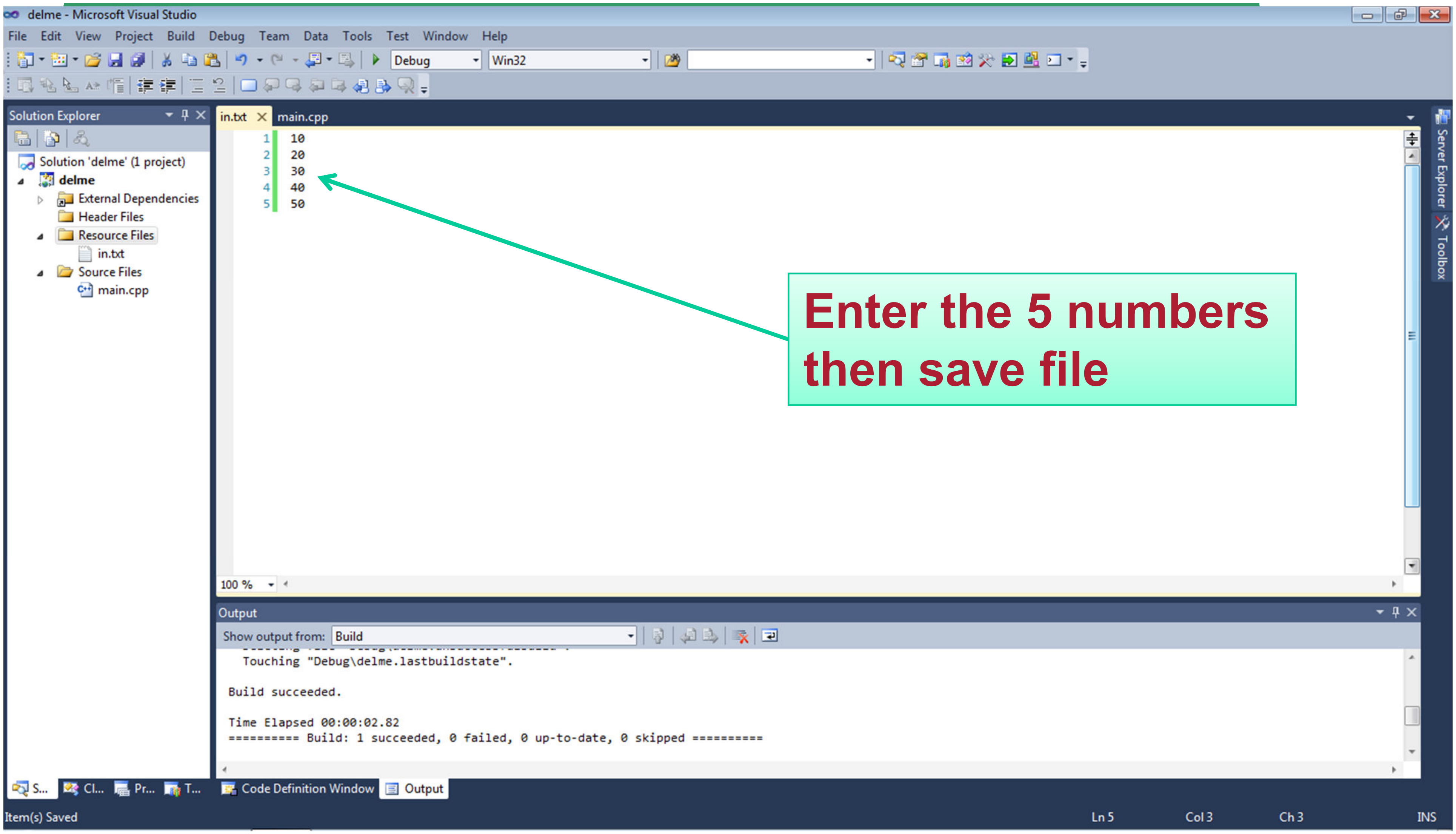
Registration Script (.rgs)	Visual C++
RuleSet	Visual C++
Configuration file (app.config)	Visual C++
Text File (.txt)	Visual C++

Type: Visual C++

Creates a plain text file

Name: in.txt

Location: c:\Users\shereen\documents\visual studio 2010\Projects\delme\delme



```
10
20
30
40
50
```

Enter the 5 numbers then save file

```
  Touching "Debug\delme.lastbuildstate".

Build succeeded.

Time Elapsed 00:00:02.82
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Practice

- Write a program that will read the following file and find the largest value. The file will contain 100 integers. Output the largest value to the screen.
- Part of the file (data.txt):

```
59
98
99
77
66
73
85
```

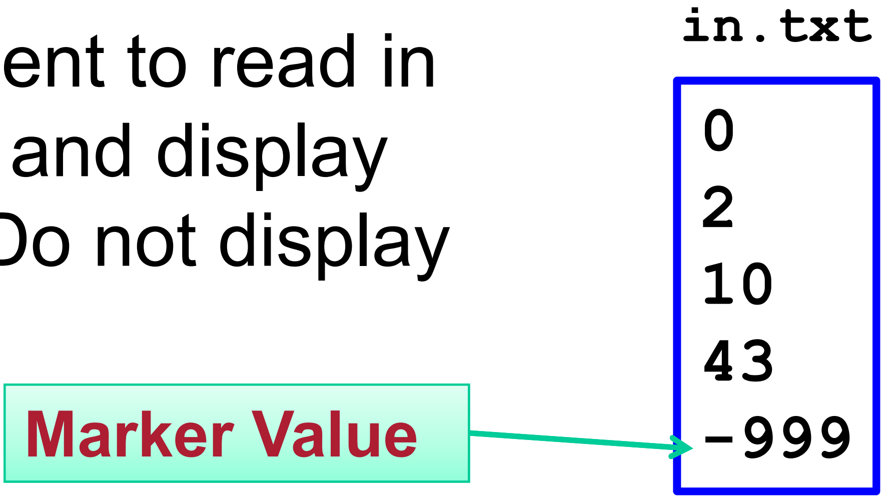
Practice

- Change the previous program so that the data is displayed both to the screen and to a file named output.txt

When to Stop

- What if we don’t know the number of items in the file?

- Marker : read until some value
  - Write the code segment to read in the numbers in in.txt and display them to the screen. Do not display the marker value!

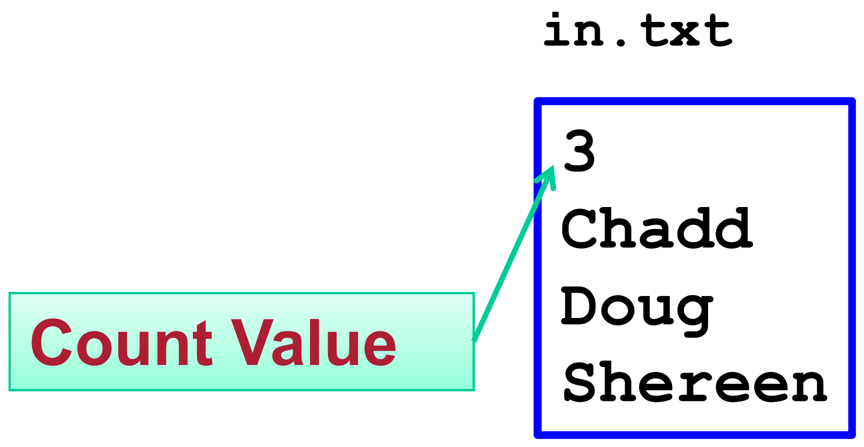


in.txt

```
0
2
10
43
-999
```

When to Stop

- Count: First integer tells us how much data to read
  - Write the code segment to read in the strings in the file in.txt and display them to the screen. Do not display the count value!



in.txt

```
3
Chadd
Doug
Shereen
```