Introduction

This programming challenge is intended to provide an enjoyable opportunity for you to demonstrate your ability to solve a software problem.

While there is no set time limit for completing the challenge, it is anticipated that the task should take no more than 2-3 hours.

We hope you find it a fun and interesting challenge as well as a useful way to demonstrate some of your software development skills.

Problem description

The challenge is to develop a software application that is able to calculate and output information about features of a provided map.

A map is an ASCII text file where each character represents a location that can either be water, land, tree or mountain.

The characters for each map feature are defined in the following table:

Map Feature	**ASCII Character**	**Description**	**ASCII value (Dec)**	**ASCII Value (Hex)**
water	.	Full stop / dot / period	46	2E
land	+	Plus sign	43	2B
tree	*	Asterisk	42	2A
mountain	^	Caret / circumflex	94	5E
building	@	At-sign	64	40

The software application must read the map text file, calculate the number of landmasses contained in the map file and output the result.

A landmass is defined as one or more connected locations where each location contains a landmass character. A landmass character can be either a land, tree, mountain or building character. Locations are connected if in a 3x3 grid around a landmass character any of the 8 surrounding locations contains another landmass character.

For example, the following 4 land characters are connected as part of a landmass:

```
.+.
++.
..+
```

For each landmass:

- Any of the landmass characters can mark the edge of a landmass.
- A landmass can contain any combination of landmass characters.
- A landmass may reach the edge of a map but does not wrap around to the other side of the map. Therefore, a landmass that reaches the edge of the map is terminated at the edge of the map.

An example connected landmass is shown here:

```
. . . . . .
```

```
....++++....+...
...**+++..+++...
....^+++++++....
....^^^++++.....
.....^++**+.....
.......+++......
........+.......
```

Input

The input data into your application will be a provided in a map text file.

The input filename will be provided as a command line parameter to your software application executable on the command line.

For example, if your application is named “mapSolver.exe” it will be executed on the command line as “mapSolver mapFile1.txt”.

Each map text file contains a single map and all maps are the same size.
The size of each map is always 80 characters (width) x 50 characters (height).

Output

The output from your application can be either to the console, a separate text file or to a GUI display.

The output should contain your name and state the number of landmasses that are in the map input file.

An example output from the application on the console may look like:

The task will be considered complete if the correct number of detected landmasses is shown in the output.

Although this is not required to complete the task, additionally you may include any other information that you have calculated, for example the number of each landmass characters in each detected landmass.

Feel free to be creative!

Programming Languages and Development Environments

The task must be completed in Windows using any version of Microsoft Visual Studio or Microsoft Visual Studio Express edition.

The task can be completed using any of the programming languages that can be compiled using this development environment, however, preference will be given to candidates that complete the task using C++.

Microsoft Visual Studio 2010 Express edition can be downloaded and used free of charge from the following location http://www.microsoft.com/visualstudio/en-us/products/2010-editions/express

What to Provide

When you have completed the challenge please provide all of the source code that you have written in a zip file. This should include the Visual Studio solution files to allow the source files to be unzipped and built in Visual Studio easily.

Please also provide a compiled executable for your application in a separate zip file. This will be run and tested against some map files to see if your application correctly solves the problem.

In addition please provide a short description of the method/algorithm you designed to solve the problem.

Summary of what to provide:

1. Zip file containing source code (including Visual Studio Solution files)
2. Zip file containing a built and functioning executable
3. Short description of how you solved the problem

What if your application fails/produces an incorrect result?

The description of how you attempted to solve the problem will be used alongside any provided source code to allow us to assess how you approached the problem.