University of Illinois at Urbana-Champaign
Department of Computer Science

MP10 : Theseus and the Minotaur

CS 225 Data Structures
Spring Semester, 1999

Handed out: Sunday, April 25th, 1999
Due date: Wednesday, May 5th, 1999 at 11:59 PM

1 Introduction

In this MP, you will get to design your own solution to a programming problem. You will write a Makefile and a main function and maybe a class or two of your own. You will also get a chance to use any classes from the course library and any STL class you want.

2 The Files

To begin with, you will need to copy the given files into your own directory... naaah, there are no given files. For this MP you will have to create your own Makefile, your own main.C and your own... you got it, you have to do it all on your own.

There are, however, the test cases we will use and the correct outputs for those test cases. There is also a sample `Makefile` that shows how to compile your programs that use the STL, together with a sample program that shows how to use the `map` class from STL. You can get all of that from

```
~cs225/src/mp10                 (map example and sample Makefile)
~cs225/src/mp10/test            (test files)
```

3 Introduction

Those of you with a classical education may remember the legend of Theseus and the Minotaur. This is an unlikely tale involving a bull-headed monster, lovelorn damsels, balls of silk and an underground maze full of twisty little passages all alike. In line with the educational nature of this MP, we will now reveal the true story.

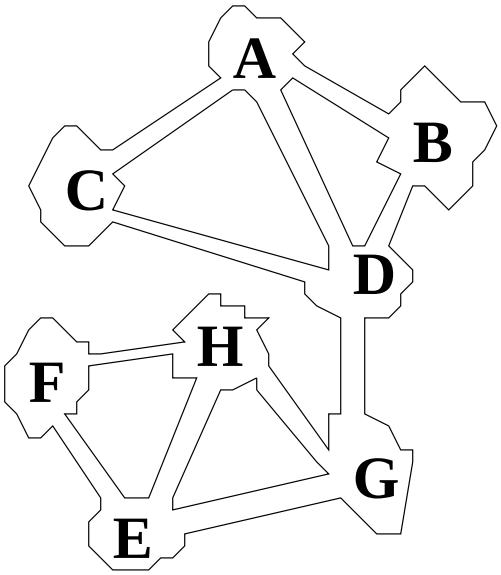
The maze was a series of caverns connected by passages. Theseus managed to smuggle into the labyrinth with him a supply of candles and a small tube of phosphorescent paint with which he could mark his way, or, more specifically, the exits he used. He knew that he would be lowered into a passage between two caverns, and that if he could find and kill the Minotaur he would be set free. His intended strategy was to move cautiously along a passage until he came to a cavern and then turn right (he was left-handed and wished to keep his sword away from the wall) and feel his way around the edge of the cavern until he came to an exit. If this was unmarked, he would mark it and enter it; if it was marked he would ignore it and continue around the cavern. If he heard the Minotaur in a cavern with him, he would light a candle and kill the Minotaur, since the Minotaur would be blinded by the light. If, however, he met the Minotaur in a passage he would be in trouble, since the size of the passage would restrict his movements and he would be unable

to either light a candle or fight adequately. When he entered a cavern that had been previously entered by the Minotaur he would light a candle and leave it there and then turn right (as usual) but take the Minotaur's exit.

In the meantime, the Minotaur was also searching for Theseus. He was bigger and slower-moving but he knew the caverns well and hence, unlikely as it may seem, every time he emerged from a passage into a cavern, so did Theseus, albeit usually in a different one. The Minotaur turned left when he entered a cavern and traveled clockwise around it until he came to an unmarked (by him) exit, at which point he would mark it and take it. If he sensed that the cavern he was about to enter had a candle burning in it, he would turn and flee back up the passage he had just used, arriving back at the previous cavern to complete his 'turn.'

4 Example

Consider the following labyrinth as an example



Assume that Theseus starts off between A and C going toward C, and that the Minotaur starts off between F and H going toward H. After entering C, Theseus will move to D, whereas the Minotaur, after entering H will move to G. Theseus will then move towards G while the Minotaur will head for D and Theseus will be killed in the corridor between D and G. If, however, Theseus starts off as before and the Minotaur starts off between D and G then, while Theseus moves from C to D to G, the Minotaur moves from G to E to F. When Theseus enters G he detects that the Minotaur has been there before him and heads for E, and not for H, reaching it as the Minotaur reaches H. The Minotaur is thwarted in his attempt to get to G and turns back, arriving in H just as Theseus, still 'following' the Minotaur arrives in F. The Minotaur tries E and is again thwarted and arrives back at H just as Theseus arrives in hot pursuit. Thus the Minotaur is slain in H.

Write a program that will simulate Theseus' pursuit of the Minotaur.

5 Input

Your program should read its input from the standard input (using `cin` as the input stream). The input starts with the definition of the labyrinth. The labyrinth definition will contain a series of

cavern descriptions, one per line. Each line will contain the word `FROM`, then a cavern name (a single word), followed by the word `TO` and then a list of caverns reachable from it (in counterclockwise order). No cavern will be connected to itself. The cavern descriptions will not be ordered in any way. The description of a labyrinth will be terminated by a line starting with the word END, followed by two pairs of cavern names. The first pair indicates the passage in which Theseus starts, and the second in which the Minotaur starts. The travel in a starting passage is toward the cavern whose identifier is the second character in the pair.

A final encounter is possible for each test input.

6 Output

Output will consist of one line. That line will specify who gets killed and where. Note that if the final encounter takes place in a passage it should be specified from Theseus' point of view. Follow the format shown in the examples below exactly.

There are some special cases that need to be resolved:

1. If Theseus and the Minotaur start in the same tunnel, then Theseus gets killed. It does not matter whether they faced the same direction or the opposite directions.

2. If Theseus lights a candle in some cavern A and enters a tunnel towards some other cavern B, and in the same turn the Minotaur follows a tunnel from B to A, then Theseus gets killed because they meet in the middle of the tunnel, before the Minotaur has a chance to see the candle in A and turn back.

7 Examples

The content of `test/test.0` is:

```
FROM Acavern TO Bcavern Ccavern Dcavern
FROM Dcavern TO Bcavern Acavern Ccavern Gcavern
FROM Fcavern TO Hhall Eroom
FROM Gcavern TO Hhall Eroom Dcavern
FROM Bcavern TO Acavern Dcavern
FROM Eroom TO Fcavern Gcavern Hhall
FROM Hhall TO Fcavern Eroom Gcavern
FROM Ccavern TO Acavern Dcavern
END Acavern Ccavern Fcavern Hhall
```

Your program will be called like this:

```
a.out < test/test.0
```

Your program should print the following result:

```
Theseus is killed between Dcavern and Gcavern
```

The content of `test/test.1` is:

```
FROM Acavern TO Bcavern Ccavern Dcavern
FROM Dcavern TO Bcavern Acavern Ccavern Gcavern
FROM Fcavern TO Hhall Eroom
FROM Gcavern TO Hhall Eroom Dcavern
FROM Bcavern TO Acavern Dcavern
FROM Eroom TO Fcavern Gcavern Hhall
FROM Hhall TO Fcavern Eroom Gcavern
FROM Ccavern TO Acavern Dcavern
END Acavern Ccavern Dcavern Gcavern
```

Your program will be called like this:

```
a.out < test/test.1
```

Your program should print the following result:

```
The Minotaur is slain in Hhall
```

You must match the output EXACTLY! If your output does not match exactly, you will receive no points for that example. Your grade for this MP will be based almost entirely on passing the examples.

DO NOT try to cheat by patching the program so that it works only for the test inputs (i.e. if I read the input, I see which one of the inputs I have read, then I print the correct line without ever actually solving the problem). This would be considered an infraction of academic integrity and penalized as described in the "Code of Policies and Regulations Applying to All Students", Rule 33.

8 PSP

Don't forget your PSP work!

9 Handing in your code

To hand in your MP10 code, you must hand in the files that you wrote AND any files from our course library that you use. Additionally, you must submit the `Makefile` you use for compiling your MP 10. Your `Makefile` must produce an executable named "a.out" when `make` is run.

Because of the limitations of the `handin` program, you must copy your `Makefile` into a file named `myMakefile` and then hand that in, together with your .C and .h files.

For example, suppose I wrote main.C, Maze.C and Maze.h and I used the InfoGraph class (which in turn uses Graph and Array, which in turn uses Assert) and `map` from STL. I also have this `Makefile` that compiles everything and produces `a.out`. Then, before handing in, I do the following:

```
cp Makefile myMakefile
handin cs225 mp10 myMakefile main.C Maze.h Maze.C InfoGraph.h InfoGraph.C Graph.h
Graph.C Array.h Array.C Assert.h Assert.C
```

Notice that I do not hand in **map** because it is a part of STL.

Make sure that you check `handin10.log` file after you hand in. If your code did not compile because you forgot to hand in one of the files, you get a zero for this MP because it does not compile. If your Makefile produces an executable, but its name is not `a.out`, you get no points for any of the test cases.

10 Helpful Hints

You might want to use Graph or InfoGraph classes in this MP. If you do so, notice that the order in which you insert edges into the graph is important. A newly inserted Edge is always inserted at the end of the departing list of its source vertex and at the end of the entering edge list of its target vertex. In order to preserve the order of tunnels that exit the cavern, you may want to use a directed graph and insert every edge once in each direction. then you need to worry only about the order of departing edges for each vertex.

You might want to use the `map` class from STL in order to translate from cavern names to caverns (vertices, if you use a graph). Study the `map` example carefully and that should be enough for the purposes of this MP.

11 Credits

Modified from an ACM Programming Contest Finals problem, with permission from ACM.