- a function for printing a maze out via a suitable pointer-to-`struct` argument, with each cell's character doubled to make the maze easier to view on the screen.

The required output for this stage for `test0.txt` is:

```
Stage 1
=======
maze has 5 rows and 9 columns
##..###############
##.............##
########..########
##........##....##
##..########..####
```

In this stage your elemental `struct` for each cell might only contain one variable, the type of that cell.

Stage 2 – Determining Reachable Regions (marks up to 12/20)

Each legal move of the robot takes it either one step vertically or one step horizontally, from one viable cell to an immediately adjacent viable cell. A viable cell is *reachable* if the robot can reach it starting at any of the gaps in the top row of the maze, and then following (any number of) legal moves.

Develop an algorithm for determining (and storing with your structure) a flag that records the reachability of every viable cell in the maze, including any exit gaps in the last row. The output of this stage is again a map of the maze, but with reachable cells shown as doubled `'+'` characters, and non-reachable cells as doubled `'-'` characters. If any of the exit gaps is reachable, then the maze as a whole can be reported as having a solution. The required output from this stage for `test0.txt` is

```
Stage 2
=======
maze has a solution
##++###############
##++++++++++++++##
########++########
##+++++++++##----##
##++########--####
```

Note the region in this maze that is non-reachable, including one of the exit gaps. (The alternative message to be used, if all of the exit gaps are non-reachable, is `"maze does not have a solution"`.) Further examples showing the full output that is required are provided on the LMS, and you should study them carefully so that you understand the details of what is required in this stage.

Note that the output from this stage is *in addition* to the output of Stage 1.

Stage 3 – Calculating Costs (marks up to 16/20)

Now add a further variable to the `struct` that represents each cell of the maze, and for each reachable cell, compute into that variable the minimum cost of any path from any entry gap in the top row through until that cell, counting one unit of cost for each cell that is traveled through, and with a cost of zero assigned at each top-row gap. The cost of every second reachable cell should be printed using two digits; other cells should be printed as before. If the cost of a reachable cell is greater than 99, then only the last two digits of the number should be printed. The required output from this stage for `test0.txt` is