**FAQs**

If the answer isn't here then post a message to the Discussion Forum.

1.*I don’t know where to start!*

Get a very basic, simplified version of the program working first, then extend it step by step. For example, first work out how to display the background grid, with a very simple maze. Display the walls using squares, with a data structure used to represent the content of each square (wall, empty, marker).

2. *How do you position the robot, a marker, and walls in the grid?*The simplest way is to just put code in the program to position everything, but this has the disadvantage of having to edit and recompile to change any of the positions. Better would be to either use command line arguments or read a data file, or a combination of both, to input the positions so that the program code does not need to change.  
  
3. *Do I have to write complicated code so that the robot can always find the exit, wherever anything is positioned?*No, the basic questions are not really about search and movement algorithms, and follow on from the exercise questions by using functions, loops, if statements, variables, and arrays. The aim is to get a working program to display a maze and use a basic strategy to move the robot around (e.g., move forward, if cannot move forward turn left or right, until the exit is found).  
The other questions can be seen as progressively harder challenges if you want to take them on.   
  
4. *The robot is moving so fast I can't see what is happening!*  
Use the sleep function added to the drawing program: void sleep(int milliseconds). Calling the function will pause the animation for the number of milliseconds specified, e.g., sleep(1000) will sleep for 1 second (1000 milliseconds). Sleeping for 100-300 milliseconds is probably good enough, as you still want the robot to move fairly quickly.    
  
5. *The Close Window button on the drawing window doesn't respond.*  
 Type Ctrl-c (hold the control key down and press c) to stop the program.  
  
6. *The size of my .zip directory containing the finished coursework exceeds the Moodle upload size limit. Can I upload a text file containing the URL from which the .zip directory can be safely downloaded (e.g. my personal cloud storage)?*

Your zip file is way too large, it should typically only be a few kilobytes, in size! You have probably added compiled code or other files by mistake.

No, you should only upload the source code + documentation in the .zip file to Moodle, plus any data files or image files needed to run the program, and that cannot possibly be more than a few megabytes. Don't include compiled code, or large images.

7. *What is good practice for code comments?*

Comments should be kept minimal and should always **add** information that is not apparent from reading the code. The code itself should be as readable as possible, with variables and functions having clear descriptive names, and good layout used. In general, comments are a liability as they need to be kept synchronised with the code as it is edited, and can easily get out of date.

You should consider comments for:

* References/citations to other work you are making use of.
* Where necessary, providing more information about the design or use of an algorithm, minimising the dependencies on specific lines or versions of code.

If you think a comment is needed to explain a complicated section of code, then make the code more readable! One way of doing this is to split the code into a collection of smaller functions, where each function has a clear name that explains its purpose. 

Never include comments like "this is a variable called length", or any that repeat what the code says. Don't provide big multi-line blocks of comments, or use comments to break up source code files into sections (use multiple source code files instead).

8. *Can I use additional code libraries that are not part of the standard C library?*

No, you should just use the standard C libraries that come with the compiler, and the DrawApp graphics.h and graphics.c files. Any other code you need you have to write yourself. You should not need anything extra anyway!

Your code must compile and run using the standard gcc installation without any additions, except graphics.h/c.