

Beginning C Programming for Engineers

Project

Your team (1–3 people) is to write a program that plays the game, 4×4 Tic-tac-toe. This is the simple game played on a 4×4 board, in which two players alternately mark a square with his or her mark (typically, an “X” or an “O”). The first player to get four of his or her marks in a line along a row, column, or diagonal wins. The figure shows, from left to right, a final board position where smart player “X” has trounced on stupid player “O”.

X	O		
	X		
		X	O
O			X

Your objective is to write a program that plays “reasonably well.” If it can win in a turn, it will, otherwise it will attempt to block the human player from winning, if it can and must. Bonus points will be awarded to teams that develop programs that can never lose. The program should allow either the human player or the computer to go first. If the computer goes first, its first move should be random.

The human player can specify moves by entering the row, column of the position you wish to mark (though a “friendlier” method is also allowed). Your program must make sure this move is legal. After each move made by the program, print a representation of the board. When there is a winner, your program should indicate who won. Your program should also stop if the board filled up without a winner.

At the project review, one team member should be prepared to show the source code, compile it, and run it on a laptop. Additionally, each team will turn in:

1. a printed listing of the source code, as a separate document,
2. a flowchart for `main`, and
3. flowcharts for any two of your functions.

Evidence of design and attention to detail is much more important than attempting to use advanced features we haven’t covered in class. Points may be deducted for insufficient comments, poor use of functions, bugs, or a sloppy user interface. Pay attention to how nicely your program prints everything, such as prompts and the representations of the board. Use the grading criteria to gauge how much time you should spend on various parts of the project. As usual, we will assume student who choose to use Visual Studio™ have had prior programming experience, and will accordingly grade their projects against a higher standard.

Goals of this Project

The project is intended to help you put assimilate all that has been covered in class. Obviously, you will need to be able to write valid C statements. Less obviously, you will need to analyze a non-trivial problem, figuring out how to break it into simpler problems, solving those problems, and integrating those solutions into one uniform program. In short, you will need to *think like a programmer*.

In the past, many students who have not easily grasped the contents of this course have found that working on the project “brings everything together” for them. Diligently working on the project is also your best possible preparation for the final exam.

Teams and Collaboration

As with regular homework assignments, you are encouraged to form teams of up to three (3) classmates. Every member of the team will receive the same grade for the project.

The work submitted by a team is to be the labor solely of that team. Collaboration with members of other teams or people outside the class is prohibited. Illegal collaboration would include using programs you find on the web, in homework files, etc. Contracting out your program is prohibited.

Grading

The project will be graded out of a maximum possible of 100 points, determined by adding your score in each of the following areas:

30 points The program compiles under `gcc` and is a reasonable and sincere effort at solving the problem. The team must be able to answer questions about the program, displaying an understanding of the code.

10 points The flowchart submitted for `main` is reasonable and matches the actual `main` function.

10 points The flowcharts submitted for the other two functions are reasonable and match the actual functions.

10 points The program is well commented, sensibly uses functions, and generally make good use of the C programming language. The program does not use global variables or global arrays. A two-dimensional array is used to represent the board.

5 points The program prints a readable board, using `X`'s and `O`'s.

5 points The program allows moves to be entered, properly describing what should be typed in, and makes sure the move is legal.

5 points The human player can choose whether to go first or second.

5 points When the computer goes first, its move is random (any square may be chosen). The `srand` call is used *properly*.

10 points The program can play a game to completion, determining who won, or if the game has ended in a tie.

5 points The computer always makes a winning move, if possible.

5 points The computer always blocks a win by the human player, if possible.

If your team's program fails to compile, the grader will attempt to guess what proportion of these categories has been implemented, and assign a grade accordingly.