The game I want to do is Battleship. Battleship as it is normally played involves two people each placing a certain number of ships of varying sizes at secret locations on a grid. The size of the ship determines how many contiguous grid locations the ships fill. The players take turns guessing grid locations where they think the other player’s ships are, and notify the other player as to the results of the other player’s guesses. When the location of one’s ship is guessed, that ship is ”hit,” and when all of the locations filled by one ship are hit, that ship is ”sunk.” The players’ guesses of grid locations are denoted by a letter and a number. After a guess, a player is only told whether they missed, hit, or sank a ship. Different ships take a different number of hits to sink, depending on how much space they take up.

There’s an element of strategy both in hiding one’s ships and in guessing the locations of one’s opponent’s ships. People tend to try and hide their ships ”randomly,” aiming to put them far away from each other but not in a location that stands out, like the edges or the center. The thing about that is that people’s ships generally fall into particular patterns, making their placement less than random. To hide one’s ships in a truly difficult to find location, one should avoid the pseudo-random areas that on first glance look like no one would ever guess them in a million years. Going for outside those areas, places too close to the margin or the center to be considered on first glance a good hiding spot, makes one’s ships harder to find. Similarly, when guessing one’s opponent’s ships, first going for the obvious ”random” areas and then expanding outward from there is a good way to catch the ships of someone who thinks they’re a better Battleship player than they actually are.

A lot of Battleship is psyching your opponent out, making them think
that they should guess locations where your ships aren’t and steering them away from where your ships actually are. That’s part of what makes the game really fun. Most computer implementations, by contrast, are meant for a single player against a computer, which uses true randomness to select both its ship locations and its guesses. I find that this approach lacks something. When playing against a computer with a random strategy one might as well just guess in a regular search grid, because the computer hasn’t hidden the ships anywhere in particular and a ship is equally likely to be found in one location as in another.

I see a few options regarding how to translate Battleship into a computer game. There’s the question of whether it should stay a 2-player game or if I should make it single-player with the computer as the opponent. If I make it single-player, I’m probably going to have to simplify the game a little. It seems like it would be very difficult to train the computer to make smart, adaptable guesses about where the player’s ships are. Another way the computer could guess might be searching in a regular grid pattern, but it seems like that would be very easy for the player to detect and work around. Alternatively, the computer could search for the player’s ships by randomly selecting grid squares. Although it covers more ground than the regular-grid-pattern method, it’s still extremely ineffective against an actual human’s intelligent hiding of ships, and would probably make the game too easy for the player.

If I put aside the part of the game that would have the computer guessing about the locations of the single player’s ships, the simplified version of the game becomes merely the computer hiding the ships and the player trying to find them. In this case I suppose the ships’ locations would be randomized somehow, and the player would guess grid squares. The computer would say ”miss,” ”hit,” and ”sink,” depending on the result of the guess. When the computer is deciding the ships’ locations, depending on how many grid squares each ship takes up the program will need to account for one ship taking up multiple squares next to each other. The computer would have to keep track of which squares belong to which ships and remember to say ”hit” or ”sink” depending on how much of the ship has already been hit.

Making the game a two-player game would almost be simpler, because there would be no need to have the computer arrange ships on its own or
spontaneously guess anything. Each player would have a turn hiding each of their ships, probably by naming the grid squares where each ship would start and end. The computer would need to store those locations. Maybe the program would display grids for each person’s turn in the manner of the board game. The player whose turn just ended turns away from the computer and the current player steps up. Two grids are shown: one with the locations of all the player’s own ships and their opponent’s guesses right and wrong, and one with all the player’s own guesses right or wrong. The player would input a guess and the computer would say whether it was a miss, hit, or sink and update the grid. Then it would tell the current player to leave and show the opponent their updated grids.

On reflection, I think that if I were to implement this game I would try to do the two-player version. It seems like it would be more fun to play than the one-player version, because this game is, at least probably implemented by any program that I could write, too simple for a computer to present any kind of challenge to a human player with a strategy. The computer’s role becomes storing and presenting relevant information at the right time, rather than making actual decisions through randomization or any other process. It seems like an interesting challenge.