Welcome to my minesweeper game. In this game I use multiple functions.
In the main function, I use file handling to display a welcome message for the player. Then, using playGame function to start the game. In the code I use dynamic memory allocation for my 2 dimensional arrays to easily use and free up memory conviniently for the program and the computer.

void playGame(void)
This function is used to start the game. To create space for dynamic 2d arrays we use memory allocation for arrays called minefield and hiddenfield.

void hiddenfieldCreator(int x, int y, int **hiddenfield)
Function that assigns empty values ('-') to the hiddenfield (field that the user sees)

void minefieldCreator(int x, int y, int numMines, int **minefield)
Function that assigns values to the original minefield, with mines and numbers (number of mines in surrounding cells)

void printField(int x, int y, int **hiddenfield)
Simple function to easily print out minefields after a move or anything else

void guess(int x, int y, int numMines, int **minefield, int **hiddenfield)
This is my richest and function containing most lines, as it is the function that is triggered when a user makes a move. Any other functions from automatic revealing of mines to losing or winning the game is employed in this function.

void checkProgress(int i, int j, int x, int y, int numMines, int **minefield,int **hiddenfield)
Function to check the progress of the player

void lose(int i, int j, int x, int y, int **minefield , int **hiddenfield)
Function to check if the user has lost and if so, terminate the game

void win(int x, int y, int numMines, int **hiddenfield , int **minefield)
Function to check if the user has won and if so, terminate the game

void adjecentCells1(int i, int j, int **hiddenfield, int **minefield)
AND
void adjecentCells2(int i, int j, int **hiddenfield, int **minefield)
Function that uses recursion to automatically reveal the surrounding cells that are not mines.
This function is devided into 2 parts, because the recursion would not work if going in opposite directions. For example i-1, j-1 and i+1,j+1 would create an infinite struggle for the recursion, pulling in opposite directions.