Battleship Game Project

Mohamed Aloraby H00410900

Introduction:

The design and implementation of a Battleship game in C is demonstrated in this report. In the project, the player competes against an AI opponent in the classic Battleship game. The player attempts to sink the opponent's ships by taking turns attacking and positioning ships on a grid. The goal of the project is to use structured programming approaches and create a fully functional, playable game that develops problem solving abilities.

Game Design:

The game is played on a 5x5 grid for both the player and the AI. Each grid cell can represent water (~), a ship (S), a hit (X), or a miss (O).

These are the 4 different types of ships in the game and how many of each type of ship are in the game.

Ship Name	Units
Battleship	4
Cruiser	3
Submarine	2
Patrol Boat	1

<u>Flow of the game and its mechanics:</u>

1. The user is required to manually position each ship by entering a coordinate and placement also saying if it will be horizontal or vertical.

2. The AI randomly positions its ships on the board.

3. The game then enters a loop in which the player and AI swap attacking.

4. The player enters coordinates to attack the AI's board, and the results hit or miss are shown.

5. The AI randomly selects valid coordinates to attack the player.

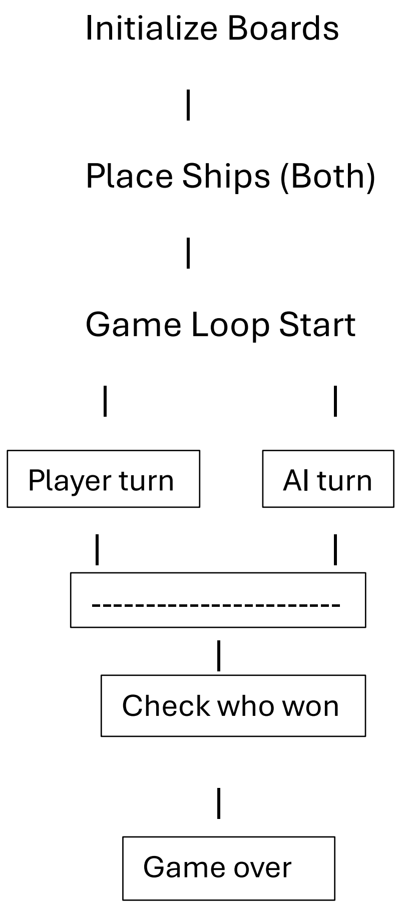
6. The game ends when one team has successfully sunk all of the opponent ships.

<u>Key implementation Details:</u>

- initializeBoards(): Sets up the boards with water (~).
- printBoard(): Displays the board with column headers A-E and row numbers 1-5.
- placeShipsManually(): Allows the player to input coordinates and directions to place ships.
- placeShipsRandomly(): Automatically places AI ships ensuring no overlap.
- playerTurn(): Accepts player input, checks for validity, and updates the AI tracking board.

- aiTurn(): Randomly selects a coordinate that hasn’t been used to attack the player’s board.
- checkWin(): Checks whether all ships on a board have been destroyed.
- A 2D character array for each board
- A ship struct for storing the ships name and sizes

Flow chart:



Instructions for playing:

Place each ship by entering:

A coordinate (e.g., B2)

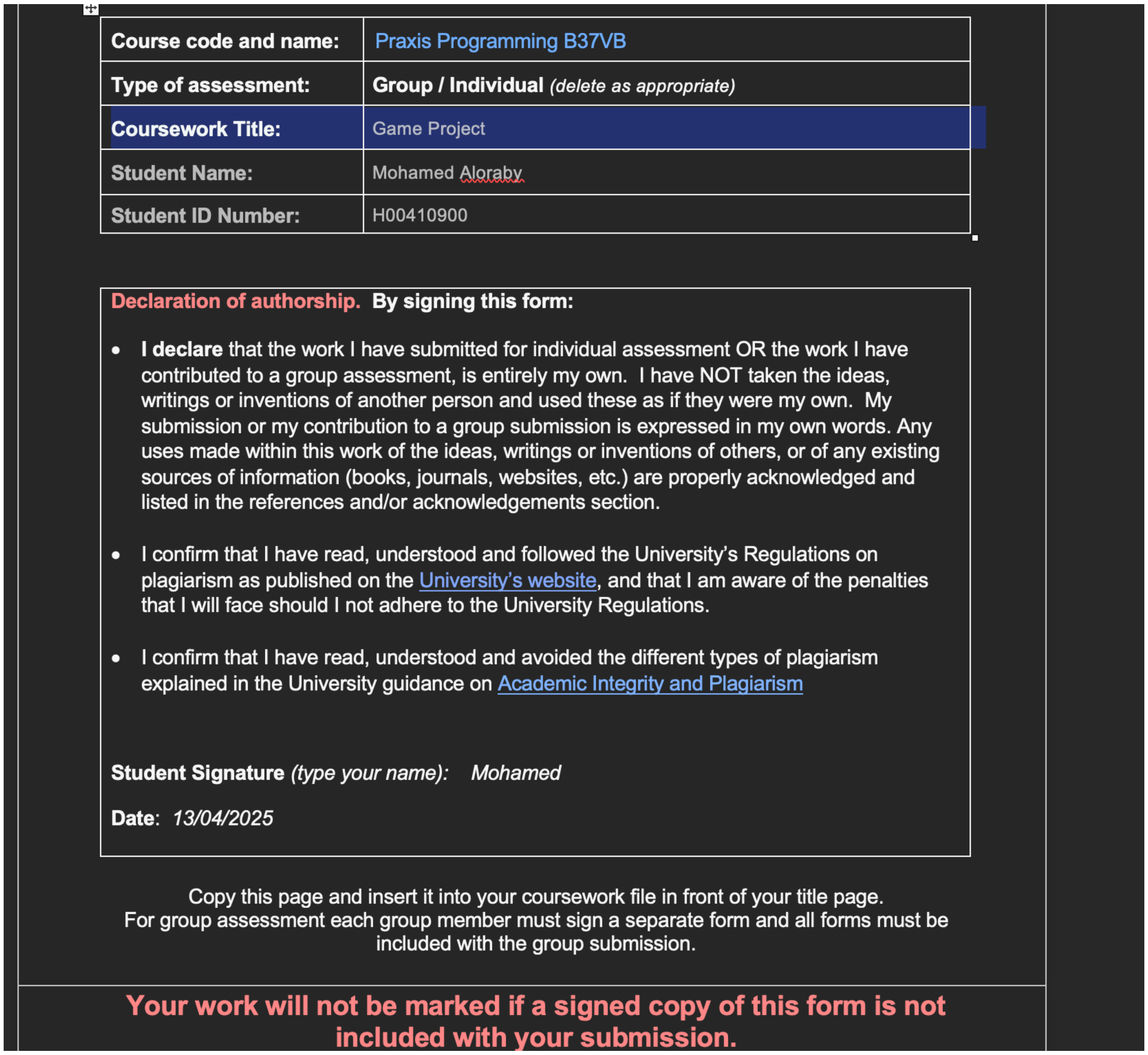
A direction: 1 for horizontal, 0 for vertical

Take turns attacking AI by entering a coordinate.

Hits and misses will be shown.

First to destroy all ships wins.

Winning: The first player to sink all ships wins the game.



Course code and name:	Praxis Programming B37VB
Type of assessment:	**Group / Individual** *(delete as appropriate)*
Coursework Title:	Game Project
Student Name:	Mohamed Alorabv
Student ID Number:	H00410900

Declaration of authorship. **By signing this form:**

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature *(type your name):* *Mohamed*

Date: *13/04/2025*

Copy this page and insert it into your coursework file in front of your title page.
For group assessment each group member must sign a separate form and all forms must be included with the group submission.

Your work will not be marked if a signed copy of this form is not included with your submission.