Department of Computer Science and Engineering

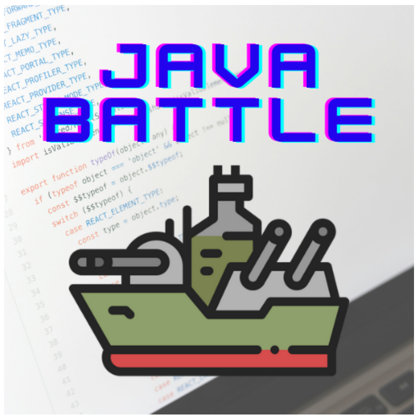
CSE 1325 Project Documentation

PROJECT NAME

Students names, surnames, IDs:
1. Blaise Kruppa, 1001837380
2. Talha Nadeem, 1001992702
3. Jesus Medina, 1001483703
4. Abdul Tahaa, 1002049951

Mentor: Marika Apostolova

CSE 1325



<u>Object oriented programming CSE 1325</u>

Student declaration:

We declare that:

- *We understand what is meant by plagiarism*
- *The implication of plagiarism has been explained to me by our professor*
- *This assignment is all team own work and we have acknowledged any use of the*

1 Student name, surname, ID and **signature:** Blaise Kruppa, 1001837380 [illegible]
2 Student name, surname, ID and **signature:** Talha Nadeem, 1001992702 [illegible]
3 Student name, surname, ID and **signature:** Jesus Medina, 1001483703 [illegible]
4 Student name, surname, ID and **signature:** Abdul Tahaa, 1002049951 [illegible]

Date:……4. 21. 2024..

	Total number of pages including this cover page	5
Class Code / Group	CSE 1325	
Lecturer's Name	MARIKA APOSTOLOVA	

CHAPTER ONE

Project introduction

This is a game where a player tries to guess the location of other players' ships. When the player guesses the location of all the opponents' ships, they win. This was inspired by the game that some of us played in our childhood called "Battleship".

Current systems (Literature review)

We have used java as our programming language and have used eclipse as the IDE. The game should work fine on any IDE.

Proposed model diagram UML

Project Specification/ Function Modules

The methods that we have used are as follows:

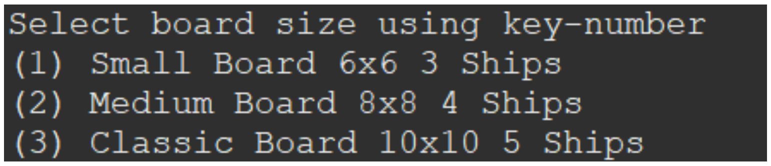
1. FIRE(String location)
2. initaializeBoard()
3. placeShipsRandomly()
4. canPlaceShip(int startRow, int startCol, int shipSize, boolean isHorizontal)
5. checkAdjacent(int row, int column)
6. displayBoard()
7. isSink(int row, int column)

Project (Input/ Output) Specification

This is a very simple game so the user is only allowed to input numbers, or strings like (A1, C3) to locate the ships, etc.

Screen Design

The screen design is pretty simple so that user dosn't get confused and can understand the game with no isuue. This is what the initial output of the code looks like:



```
Select board size using key-number
(1) Small Board 6x6 3 Ships
(2) Medium Board 8x8 4 Ships
(3) Classic Board 10x10 5 Ships
```

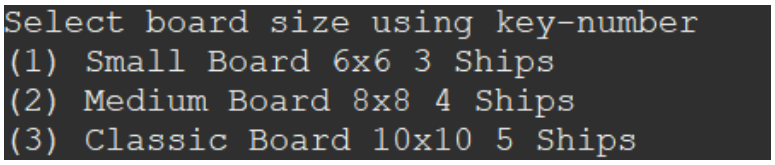
The user is required to enter a number (1-3) to continue.

CHAPTER TWO

DESIGN and CODES

Module Menu Screen

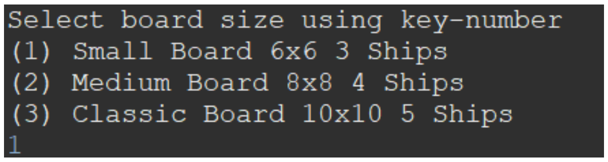
This is the initial message that game will present to the user(s) should see on the console.



```
Select board size using key-number
(1) Small Board 6x6 3 Ships
(2) Medium Board 8x8 4 Ships
(3) Classic Board 10x10 5 Ships
```

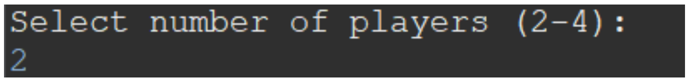
Module Select

They can then choose any of the board dimensions that they would like to play on. For example:



```
Select board size using key-number
(1) Small Board 6x6 3 Ships
(2) Medium Board 8x8 4 Ships
(3) Classic Board 10x10 5 Ships
1
```

Then the console with ask about the number of players for which there is a limit of 2-4 players.



```
Select number of players (2-4):
2
```

CHAPTER THREE

CONCLUSION

Project Weakness

C**onsole-Baseed Interface :** The game interface is text-based and displayed console, may limit the user experience compared to a graphical user interface (GUI). Enhancing the interface with graphics and interactive elements could improve user engagement.

Sinking Ship Detection: the program lacks comprehensive error handling for user inputs and edge cases. For example, it doesn't handle invalid input formats or boundary conditions gracefully, which could lead to unexpected behavior or crashes.

Project Strength

Functional Implementation: The program effectively implements the core functionally of the Battleship game, including board initialization, ship placement, firing shots, and deterring game outcomes.

Modular Design: The code is structured into separate methods within the "Board" class, making it modular and easier to understand, maintain and extend. Each method handles a specific aspect of the game, promoting code reusability and organization.

Visuable Board Size and Ship Count: program allows users to select the board size and number of ships, providing flexibility and customization options of gameplay

Multiplayer Support: supports multiplayer functionality, allowing multiple players to participate and take turns firing shots at each other.

Random Ship Placement : Ships are placed randomly non the board, adding the element of unpredictability and challenge to the game.

Project Enhancement

Graphical Usr Interface: Develop a graphical interface to provide a more immersive and interactive and immersive gaming experience. Include visual representations of the game board, ships, and firing actions for enhanced engagement.

AI Opponent: Develop an AI opponent with varying levels of difficulty to provide single-player gameplay.

Network Multiplayer: Implement network multiplayer functionality to enable players to compete each other over the internet.