Project 2

Battleship Game
(Updated with Classes and Templates)

CIS-17A

Professor:
Dr. Mark E. Lehr

Name:
Jyro Jimenez
Date:
July 30 2021

Introduction

Title: Battleship game (Updated)

This game is played with two people, however this specific code focuses on one player and a computer as an opponent. The objective of the game is to be the first to sink all five boats of your opponent. Whoever sinks the other's boats first wins the game. The rules are simple, each player has their own field, with different orientations of boats. Both players do not know the pattern of each other's boats, as they have to guess where to shoot in order to sink a ship. Each player calls on a coordinate to initialize an attack, and will be prompted whether they hit a ship or not. With this prompt, the code will illustrate with an "X" if a player misses, or an "O" if a player hits a ship. The game will end if all 9 hittable slots are hit by a player or computer.

Summary

Project size: ~900 lines with comments

Number of Variables: 29

Within this project, I recycled an old game I created for a project from csc-5, where I utilized what I had already built. In this old project, it is a version of the game which utilized while loops, functions, linear search, and 2-dimensional arrays to mimic the game "Battleship." In this game, it mimics the basic rules of battleship, where the pattern of both the player and computer's boards are randomized. Through this, it allows the player and computer to guess where the player's ships are and are informed whether they hit an enemy ship or not.

Through this barebones project that I had already built, I updated the game to illustrate the different concepts we have learned so far in CIS-17A. With each concept containing information from the first half of the class to the second. This Project 2 is the bridge between these ideas, and has been utilized to follow the guidelines set by the rubric.

In addition to the progress I've had during project 1 of CIS-17A, this project has expanded in length and concepts. For the past 2 weeks, I have tweaked the project to include classes, and its functions to better accommodate the new concepts learned within the class. These concepts include the use of inheritance and aggregation to solidify the connection between classes. Additionally, the use of vectors and templates have allowed me to connect each class together with the use of each other's variables.

Project Description

The purpose of this project is as follows:

1. Ask the player If they would like to play Battleship.
2. Play Battleship with a player and the computer.
3. State who won depending on who destroyed all the other player's ships.

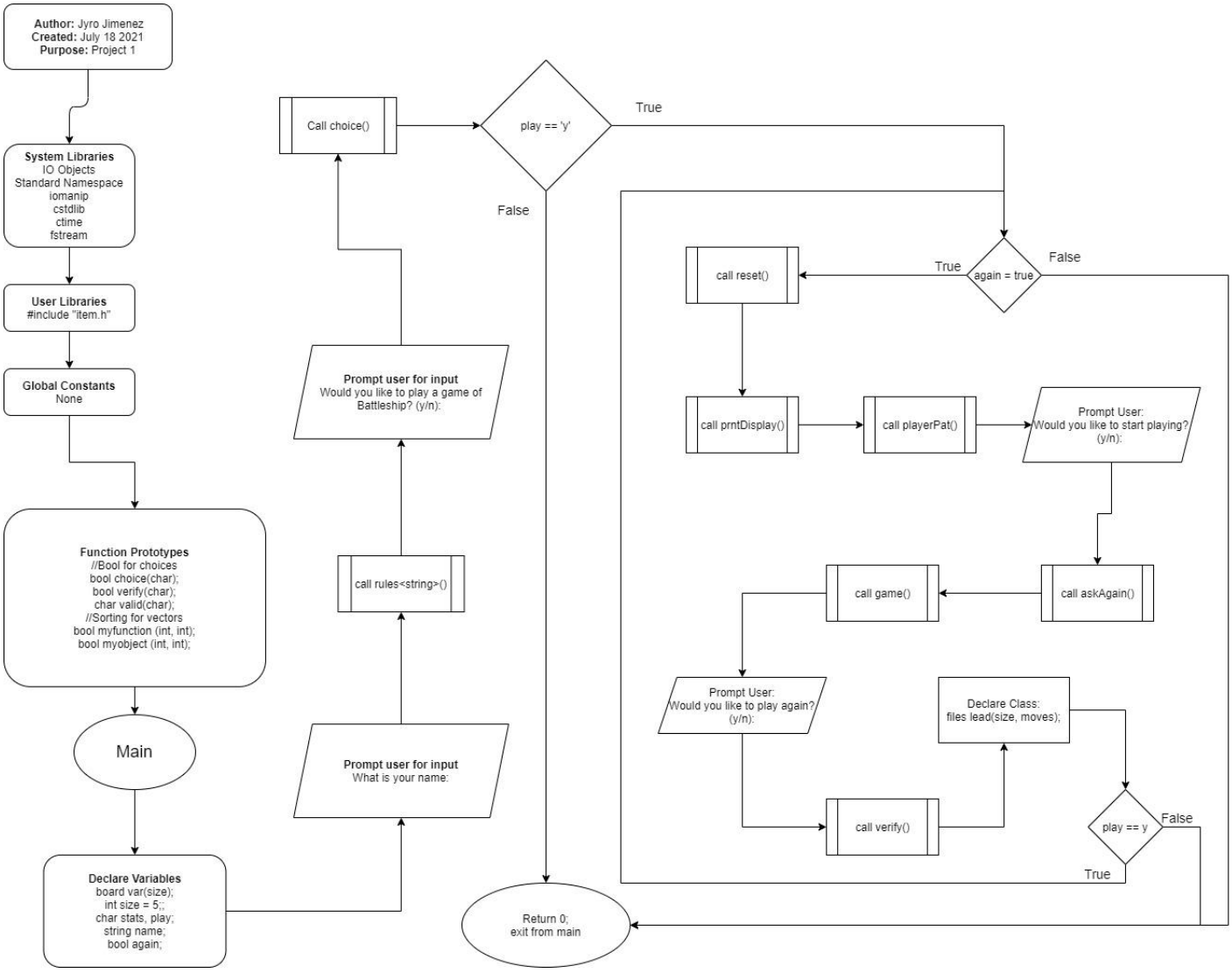
Pseudo Code

```
// System Libraries Here
// User Libraries Here
// Global Constants Only, No Global Variables
// Columns in the 2-D Arrays
// Like PI, e, Gravity, or conversions
// Program Execution Begins Here
// Set Random Number seed
// Declare Variables
// Local Constants
// Initialize structured data
// Initialize other variables
// Initialize Fstream operators
// Ask if player wants to play
// Initialize bool operator
// Start game in loop with bool operator
// Reset player, display, and computer table
// Initialize each table with a different pattern
```

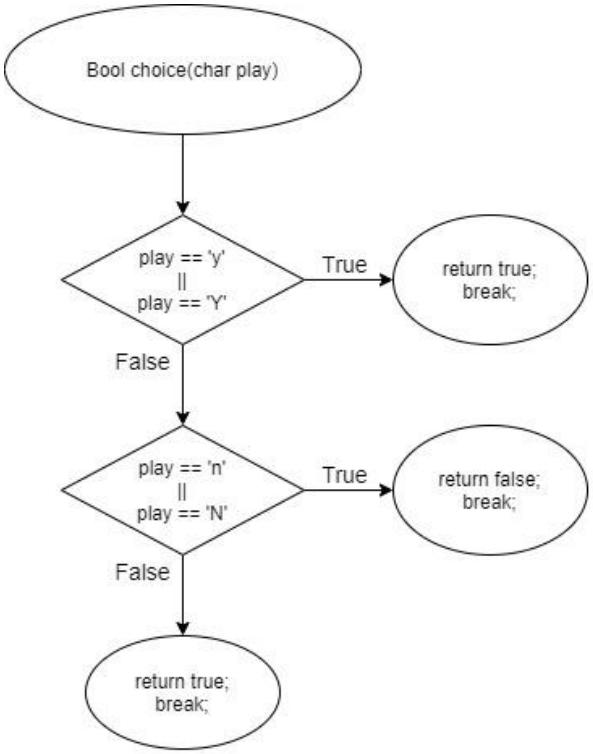
// Process/Map inputs to outputs
// Initialize count
// Pattern for player
// Assign player's pattern depending on random
// Pattern for Computer
// Pass By Reference
// Assign computer's pattern depending on random
// Output data
// Output player's display
// Output player's table
// Ask User if they want to start playing
// Loop just in case they say no
// Ask User where they want hit
// Display player's choices
// Ask player where they would like to hit
// See if player hit a ship or not
// Output that computer is choosing
// Use random and linear search to see if computer chose that number already;
// See if Computer hit a ship or not
// Output player's table
// Output Results
// Use linear search to determine how much you missed and how much you hit.
// Calculate player and computer's accuracy
// Ask if they would like to know their accuracy
// Ask if they would like to know computer's accuracy
// Give player choice to play again or not
// Verify player choice
// Exit

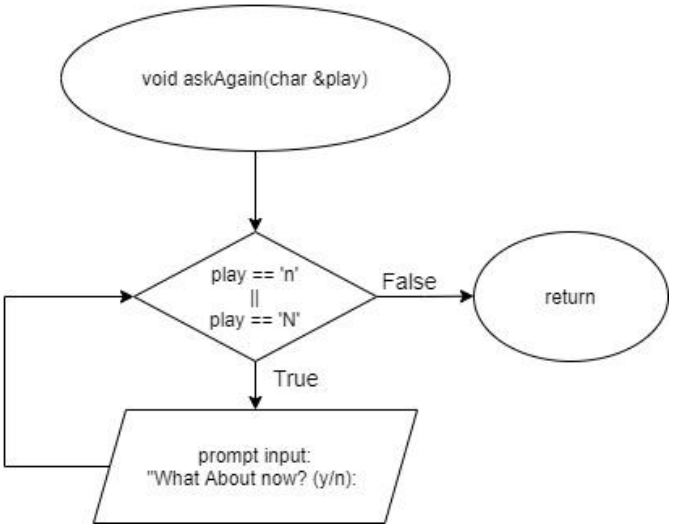
Flow Chart

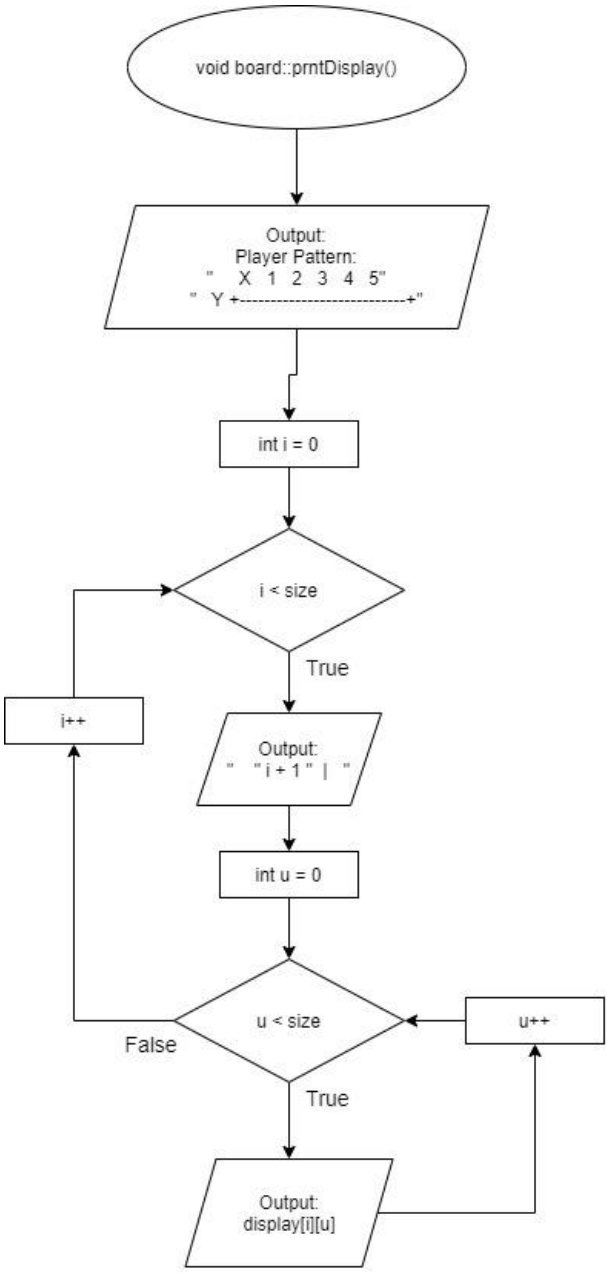
Full flow chart:

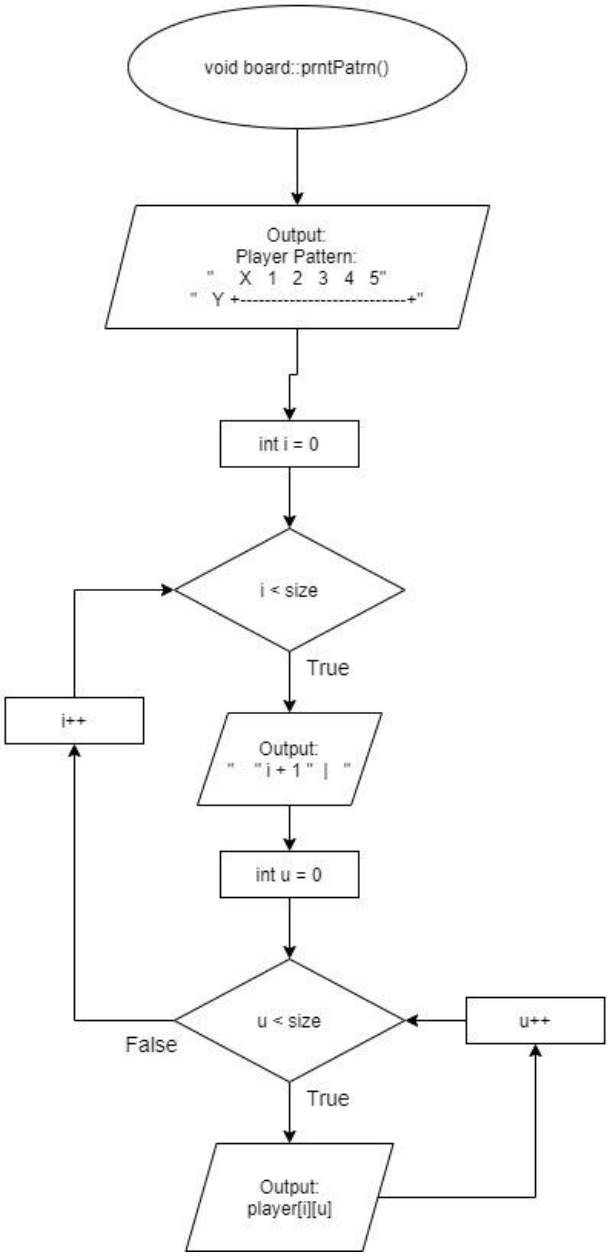


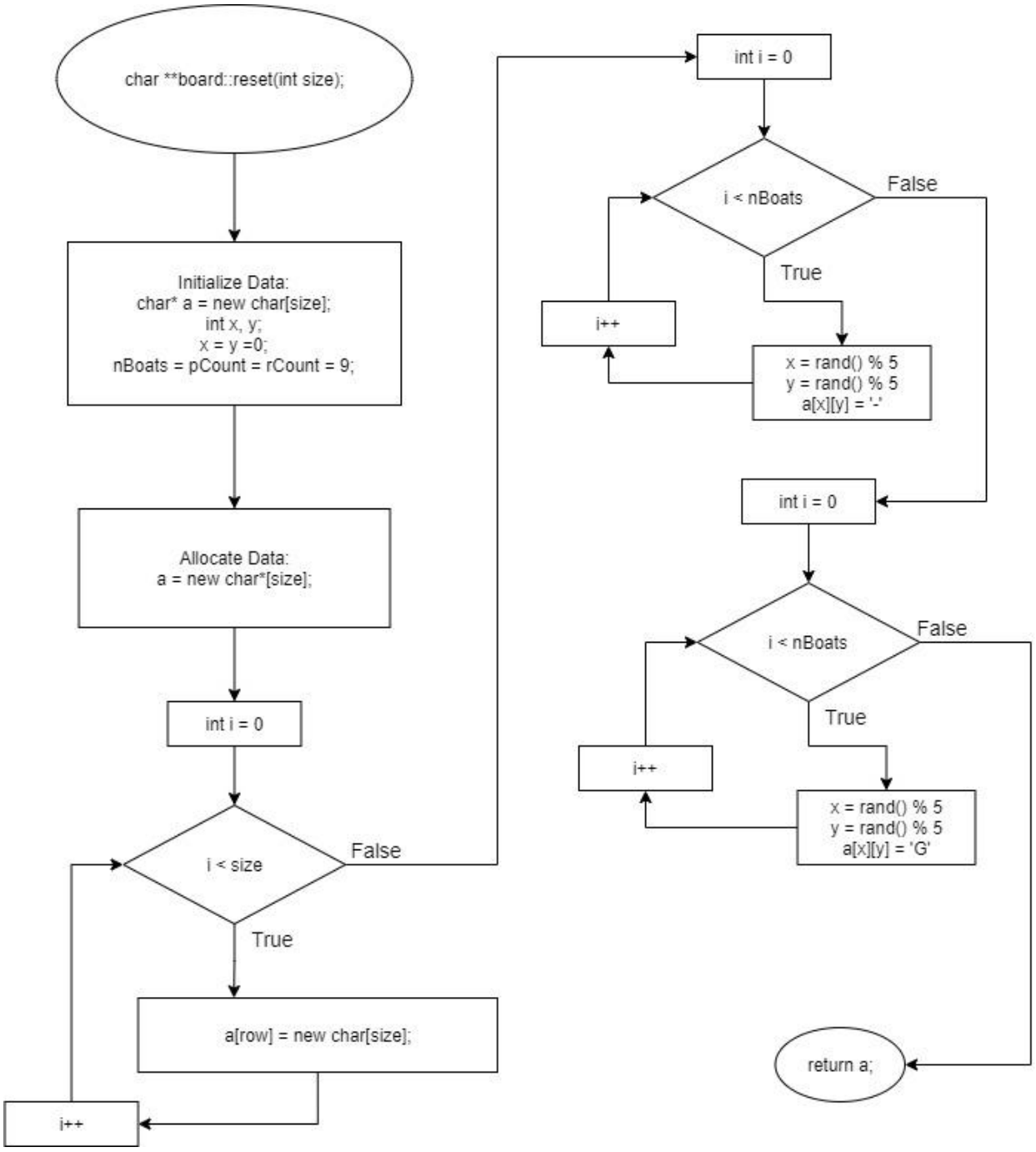
Flow chart functions:

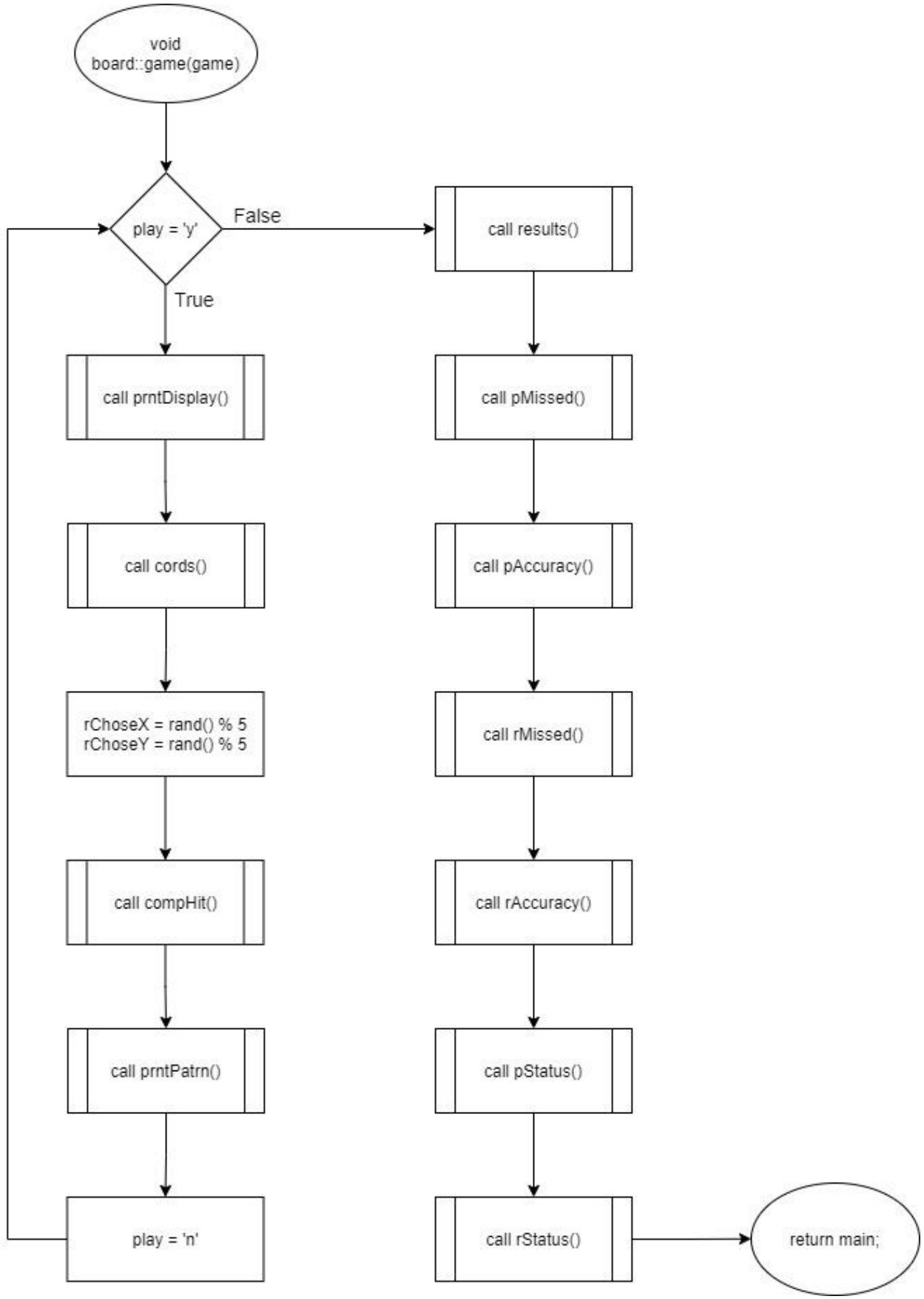




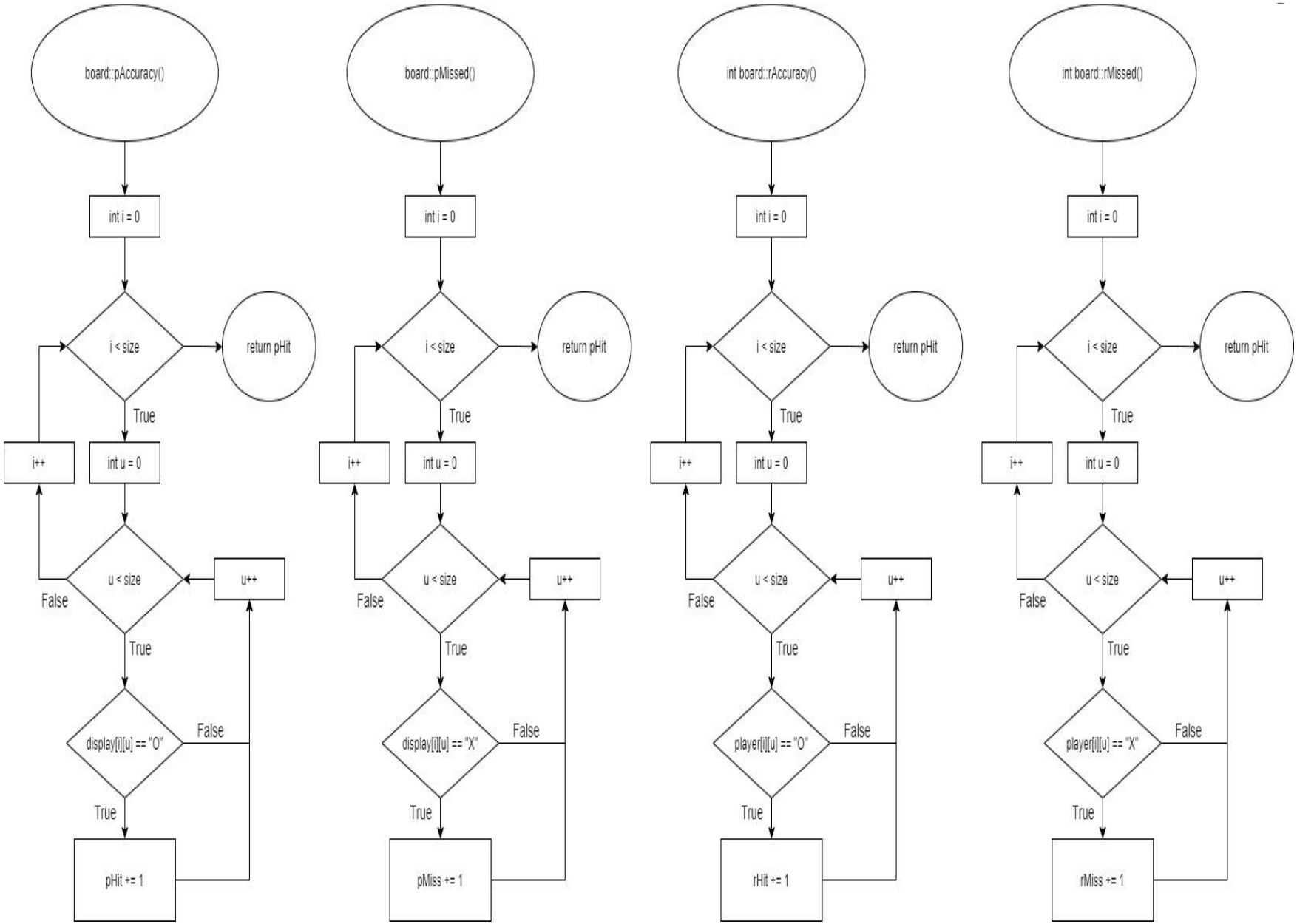

void board::prntDisplay()
Output:
Player Pattern:
" X 1 2 3 4 5"
" Y +----------+"
int i = 0
i < size
True
i++
Output:
" " i + 1 " | "
int u = 0
u < size
u++
False
True
Output:
display[i][u]

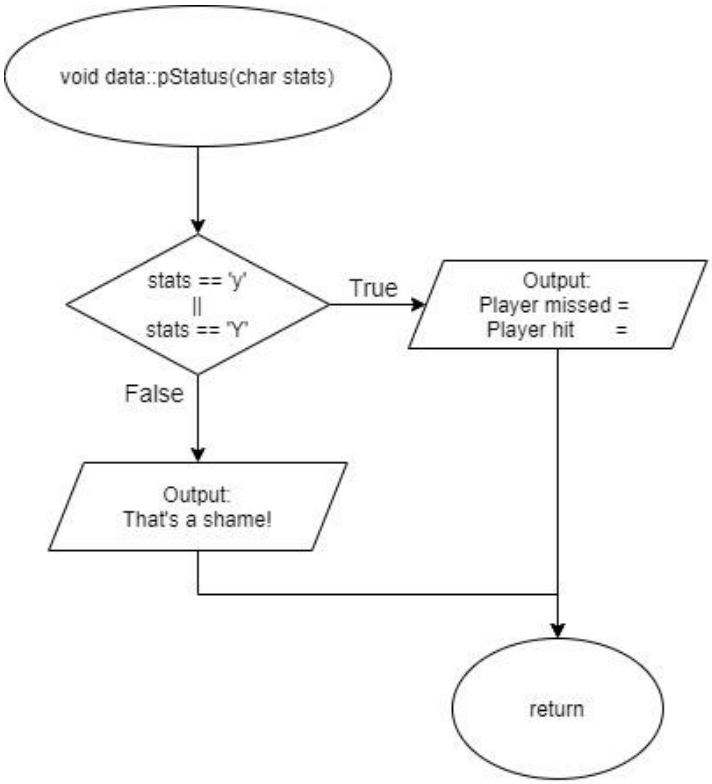

void board::prntPatrn()
Output:
Player Pattern:
" X 1 2 3 4 5"
" Y +----------+"
int i = 0
i < size
True
i++
Output:
" " i + 1 " | "
int u = 0
u < size
u++
False
True
Output:
player[i][u]

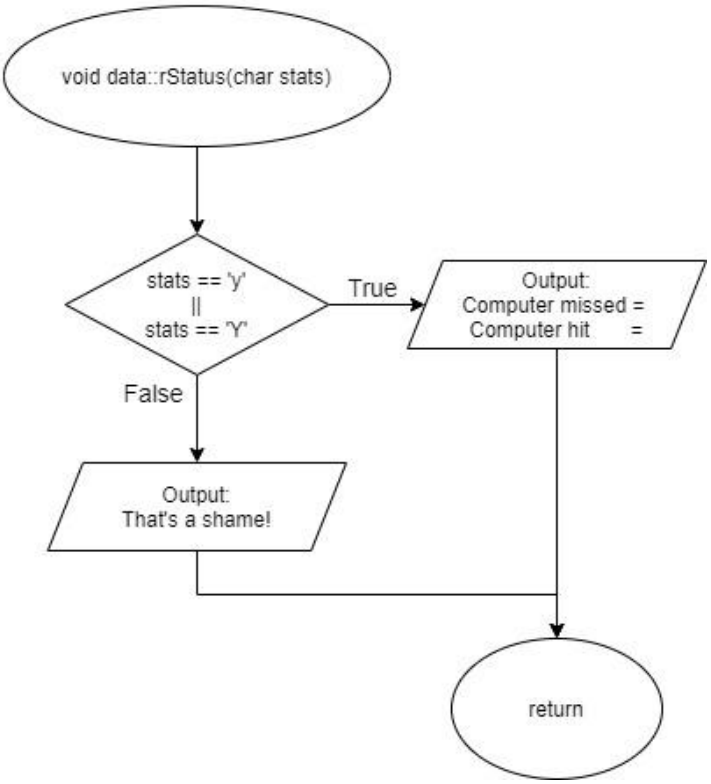

char **board::reset(int size);
Initialize Data:
char* a = new char[size];
int x, y;
x = y =0;
nBoats = pCount = rCount = 9;
Allocate Data:
a = new char*[size];
int i = 0
i < size
False
True
a[row] = new char[size];
i++
int i = 0
i < nBoats
False
True
i++
x = rand() % 5
y = rand() % 5
a[x][y] = '-'
int i = 0
i < nBoats
False
True
i++
x = rand() % 5
y = rand() % 5
a[x][y] = 'G'
return a;

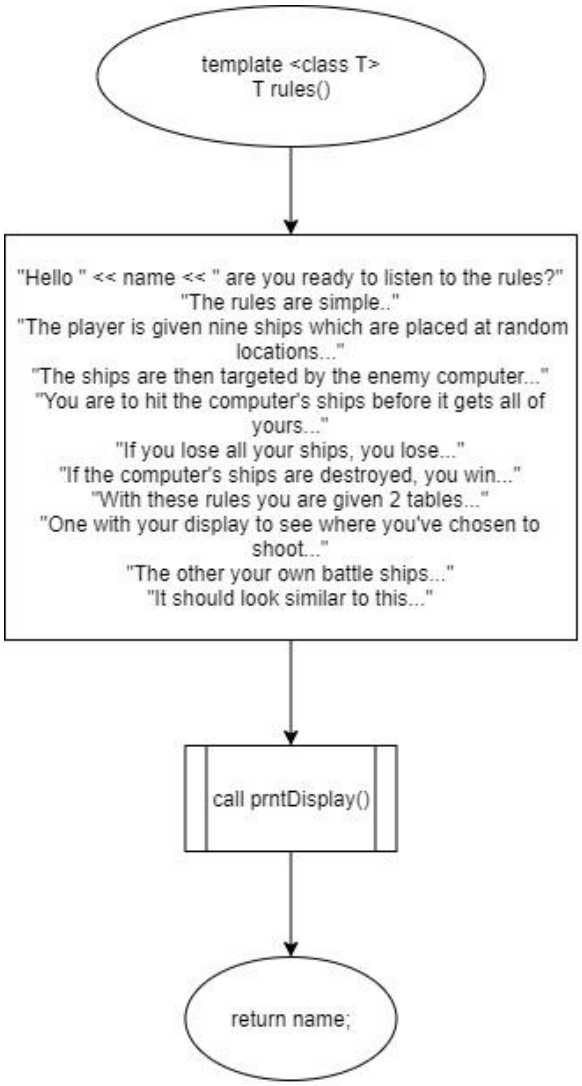

void
board::game(game)
play = 'y'
False
True
call results()
call prntDisplay()
call pMissed()
call cords()
call pAccuracy()
rChoseX = rand() % 5
rChoseY = rand() % 5
call rMissed()
call compHit()
call rAccuracy()
call prntPatrn()
call pStatus()
play = 'n'
call rStatus()
return main;

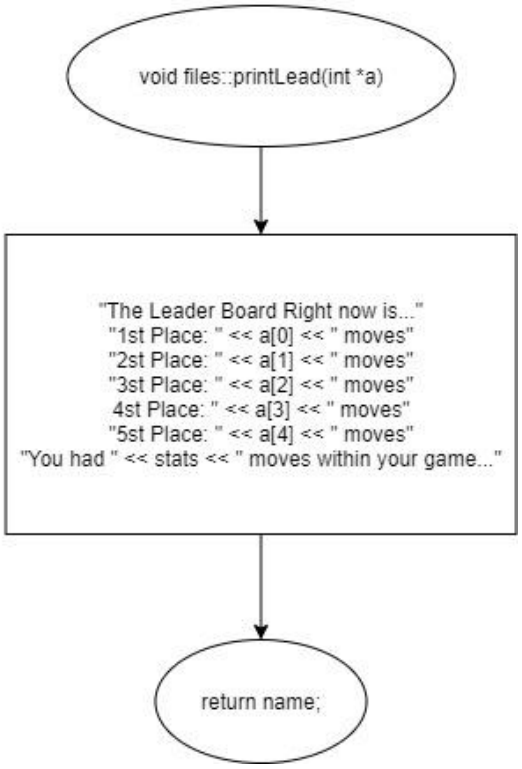
int boards::cords()
prompt input:
What coordinates would you like to shoot in?
X-Coordinate
Y-Coordinate
pChoseY > 9 || pChoseY > 8 ||
pChoseX <= 0 || pChoseY <= 0
True
False
prompt input:
These numbers are invalid try again...
X-Coordinate
Y-Coordinate
return
void board::display(HM)
compChoseY - 1]
[pChoseX - 1] != ' '
True
False
Output:
You have hit a ship
display[pChoseY - 1][pChoseX - 1] = 'O';
rCount -= 1
compChoseY - 1]
[pChoseX - 1] == ' '
True
False
Output:
You missed...
display[pChoseY - 1][pChoseX - 1] = 'X'
return
void board::compHit()
playerI[ChoseY][ChoseX]
!= ' '
True
False
Output:
Computer has hit a ship
playerI[ChoseY][ChoseX] = 'O';
pCount -= 1;
playerI[ChoseY][ChoseX]
== ' '
True
False
Output:
Computer has missed...
playerI[ChoseY][ChoseX] = 'X'
return
Bool board::choice()
rCount == 0
True
False
Output:
You have won! Congratulations
Output:
You have lost! Better luck next time!
return true;
break;


board::pAccuracy()
int i = 0
i < size
return pHit
True
i++
int u = 0
u < size
False
u++
True
display[i][u] == "O"
False
True
pHit += 1
board::pMissed()
int i = 0
i < size
return pHit
True
i++
int u = 0
u < size
False
u++
True
display[i][u] == "X"
False
True
pMiss += 1
int board::rAccuracy()
int i = 0
i < size
return pHit
True
i++
int u = 0
u < size
False
u++
True
player[i][u] == "O"
False
True
rHit += 1
int board::rMissed()
int i = 0
i < size
return pHit
True
i++
int u = 0
u < size
False
u++
True
player[i][u] == "X"
False
True
rMiss += 1


void data::pStatus(char stats)
stats == 'y'
||
stats == 'Y'
True
Output:
Player missed =
Player hit =
False
Output:
That's a shame!
return


void data::rStatus(char stats)
stats == 'y'
||
stats == 'Y'
True
Output:
Computer missed =
Computer hit =
False
Output:
That's a shame!
return


template <class T>
T rules()
"Hello " << name << " are you ready to listen to the rules?"
"The rules are simple.."
"The player is given nine ships which are placed at random locations..."
"The ships are then targeted by the enemy computer..."
"You are to hit the computer's ships before it gets all of yours..."
"If you lose all your ships, you lose..."
"If the computer's ships are destroyed, you win..."
"With these rules you are given 2 tables..."
"One with your display to see where you've chosen to shoot..."
"The other your own battle ships..."
"It should look similar to this..."
call prntDisplay()
return name;


void files::printLead(int *a)
"The Leader Board Right now is..."
"1st Place: " << a[0] << " moves"
"2st Place: " << a[1] << " moves"
"3st Place: " << a[2] << " moves"
4st Place: " << a[3] << " moves"
"5st Place: " << a[4] << " moves"
"You had " << stats << " moves within your game..."
return name;

Check-Off Sheet for Project 1

Chapter 13:

Topic	Frequency	Location
Instance of a Class	3	Inside items.h
Private Members	2	Inside items.h
Specification and Implementation	6	Inside items.h and items.cpp
Inline	1	Inside items.h
Constructors	3	Inside items.h
Destructors	2	Inside items.h

Chapter 14:

Topic	Frequency	Location
Static	1	Inside items.h
Friends	1	Inside items.h
Aggregation	1	Inside items.h

Chapter 15:

Topic	Frequency	Location
Protected members	12	Inside items.h
Base Class to Derived	2	Inside items.h
Polymorphic Associations	1	Inside items.h

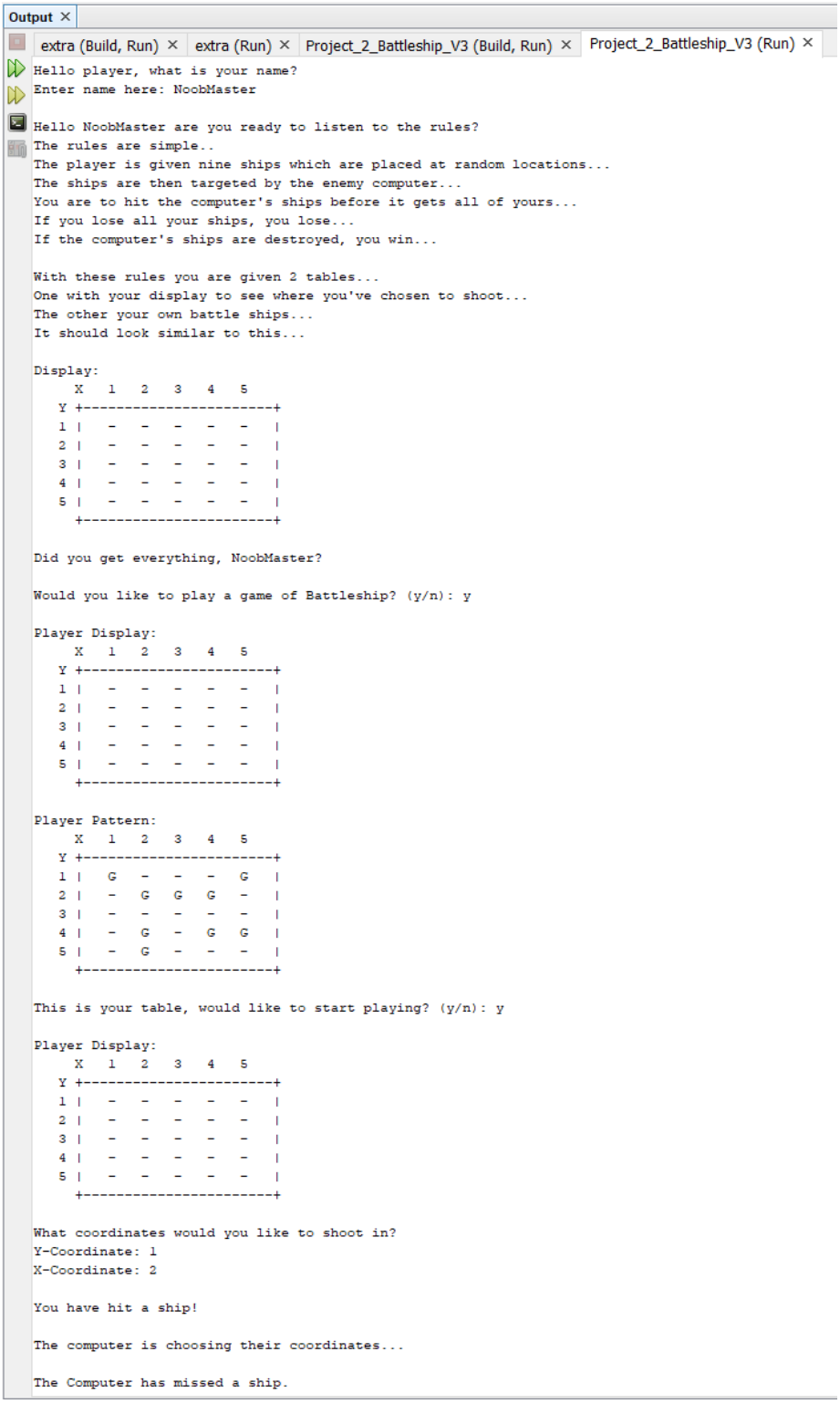
Chapter 16:

Topic	Frequency	Location
Exceptions	6	Lines 117, 141, 161, 174, 190, 232
Templates	1	Line 46
STL	3	Lines 208-221

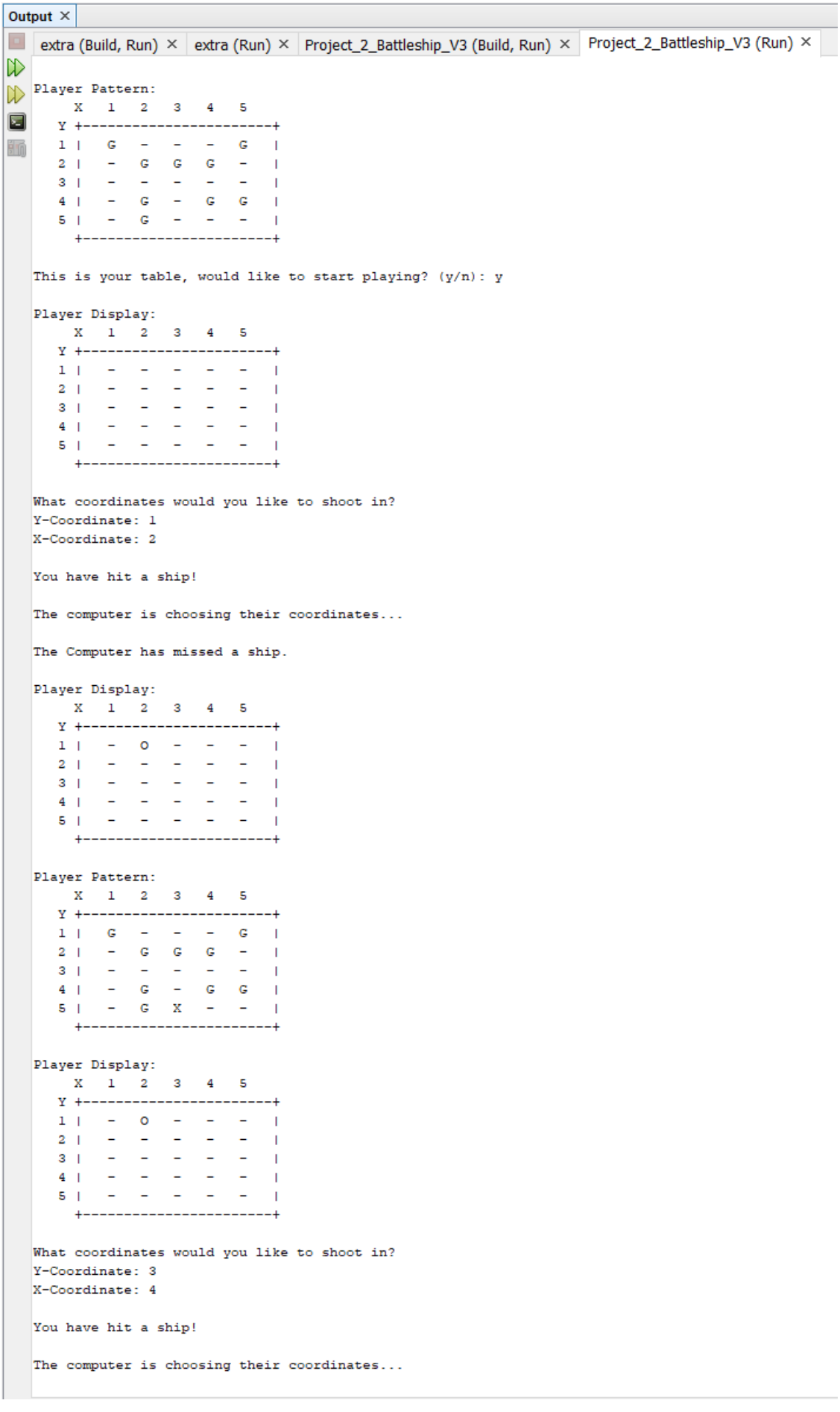
References

1. Dr. Lehr's Lectures
2. Class Homeworks
3. Class Github
4. Class books (Gaddis and Savitch)
5. https://www.w3schools.com/cpp/cpp_break.asp (Clarify the use of operators)

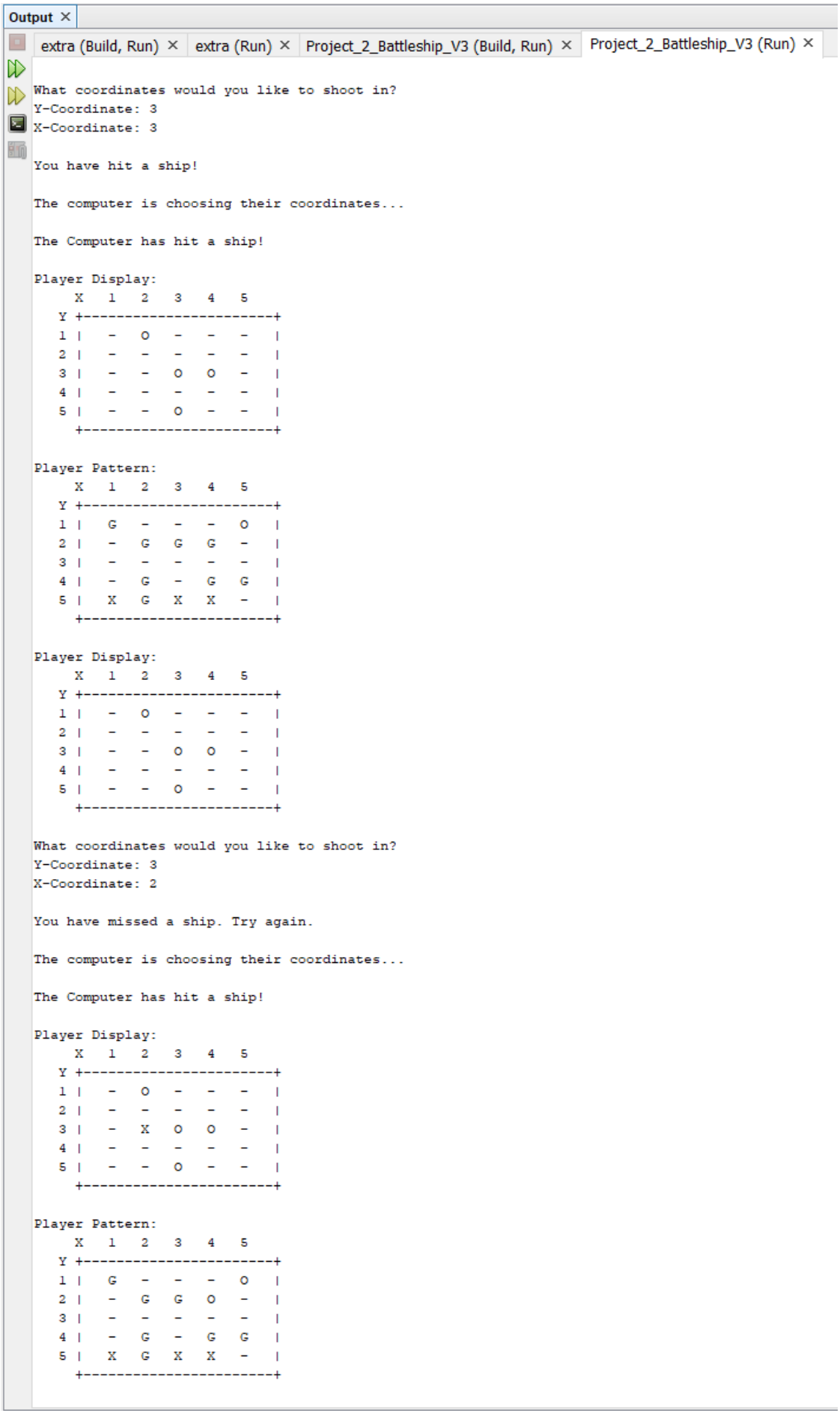
Example Output

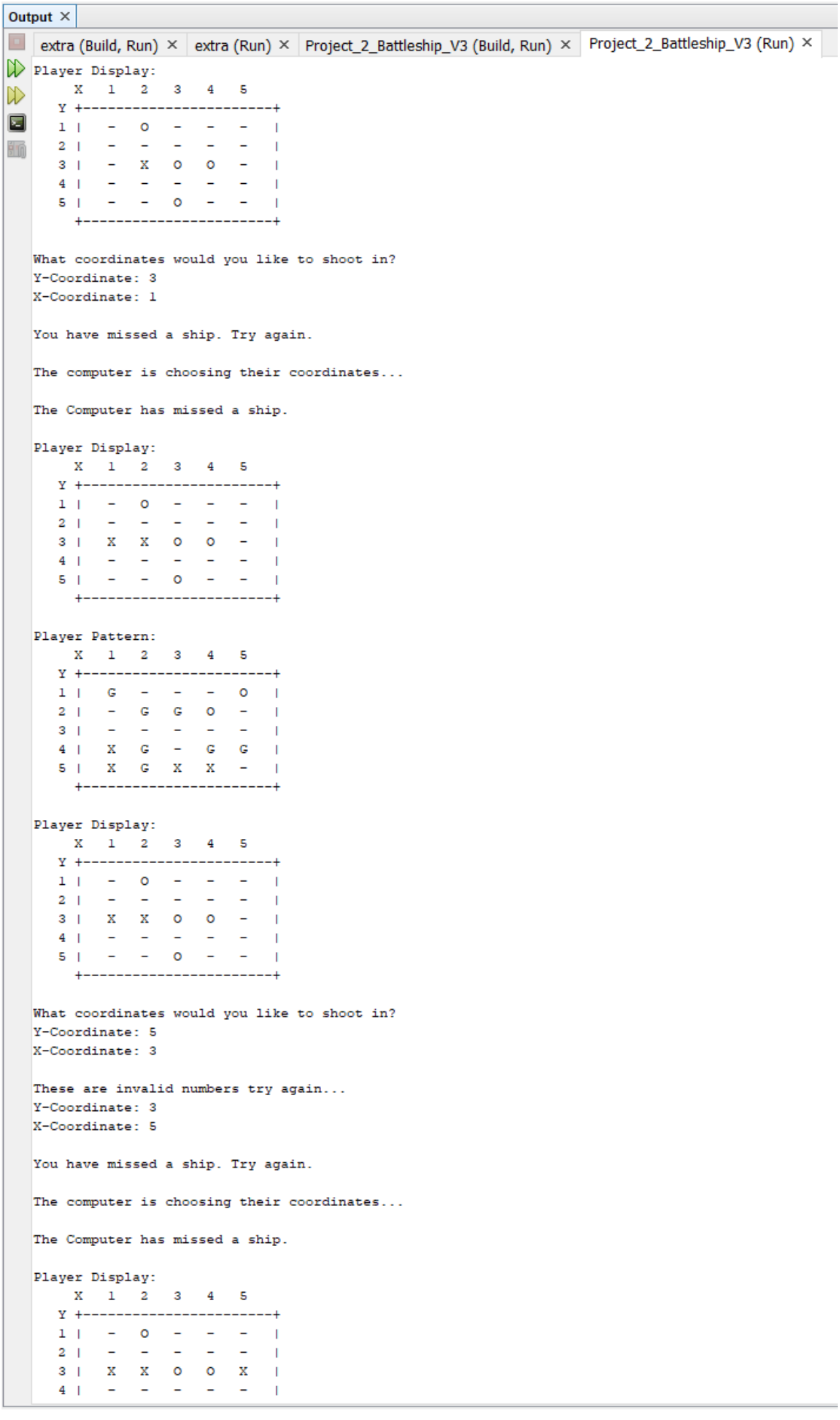


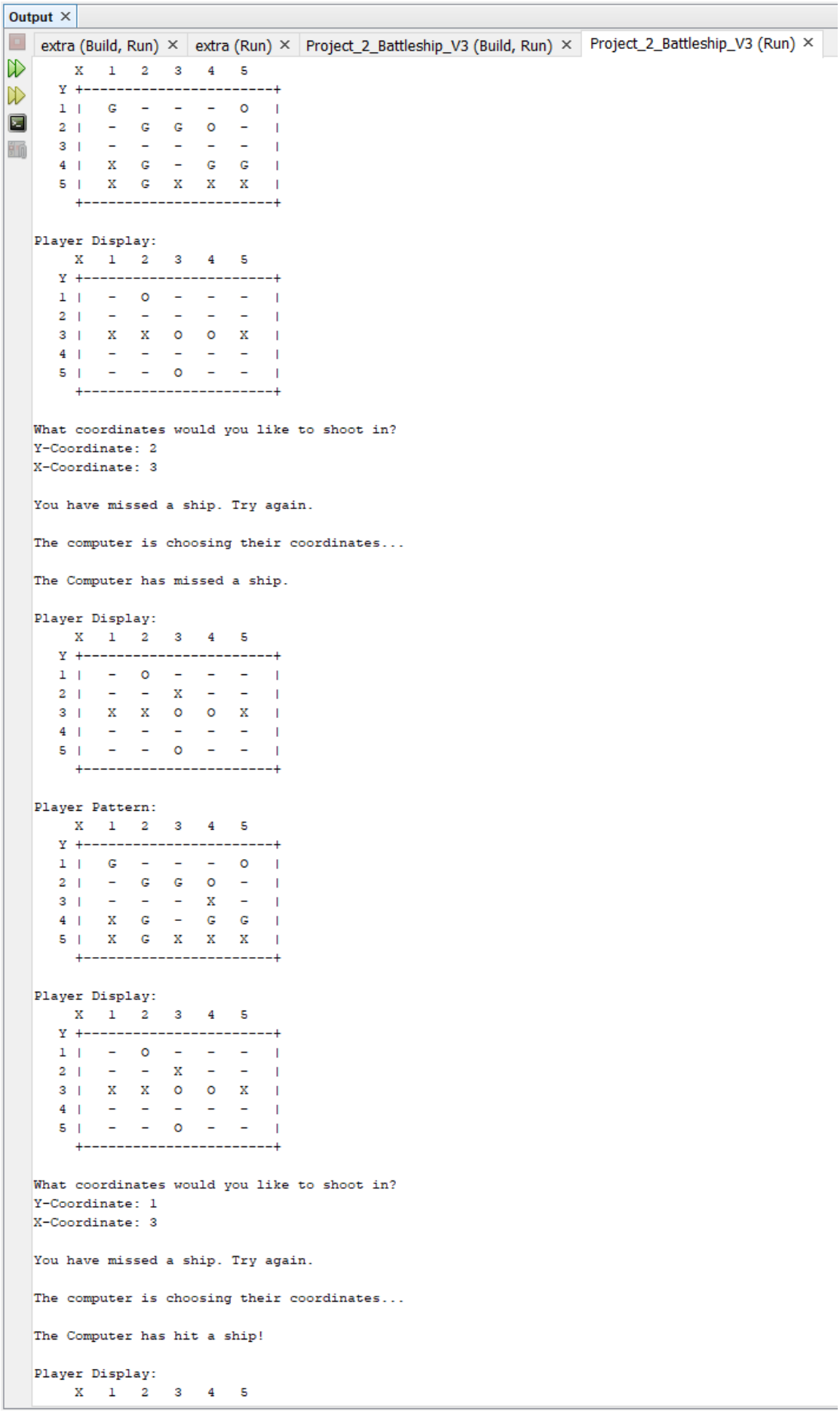
```
Output ×
extra (Build, Run) × | extra (Run) × | Project_2_Battleship_V3 (Build, Run) × | Project_2_Battleship_V3 (Run) ×
Hello player, what is your name?
Enter name here: NoobMaster

Hello NoobMaster are you ready to listen to the rules?
The rules are simple..
The player is given nine ships which are placed at random locations...
The ships are then targeted by the enemy computer...
You are to hit the computer's ships before it gets all of yours...
If you lose all your ships, you lose...
If the computer's ships are destroyed, you win...

With these rules you are given 2 tables...
One with your display to see where you've chosen to shoot...
The other your own battle ships...
It should look similar to this...

Display:
     X   1   2   3   4   5
   Y +-----------------------+
   1 |   -   -   -   -   -   |
   2 |   -   -   -   -   -   |
   3 |   -   -   -   -   -   |
   4 |   -   -   -   -   -   |
   5 |   -   -   -   -   -   |
     +-----------------------+

Did you get everything, NoobMaster?

Would you like to play a game of Battleship? (y/n): y

Player Display:
     X   1   2   3   4   5
   Y +-----------------------+
   1 |   -   -   -   -   -   |
   2 |   -   -   -   -   -   |
   3 |   -   -   -   -   -   |
   4 |   -   -   -   -   -   |
   5 |   -   -   -   -   -   |
     +-----------------------+

Player Pattern:
     X   1   2   3   4   5
   Y +-----------------------+
   1 |   G   -   -   -   G   |
   2 |   -   G   G   G   -   |
   3 |   -   -   -   -   -   |
   4 |   -   G   -   G   G   |
   5 |   -   G   -   -   -   |
     +-----------------------+

This is your table, would like to start playing? (y/n): y

Player Display:
     X   1   2   3   4   5
   Y +-----------------------+
   1 |   -   -   -   -   -   |
   2 |   -   -   -   -   -   |
   3 |   -   -   -   -   -   |
   4 |   -   -   -   -   -   |
   5 |   -   -   -   -   -   |
     +-----------------------+

What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 2

You have hit a ship!

The computer is choosing their coordinates...

The Computer has missed a ship.
```

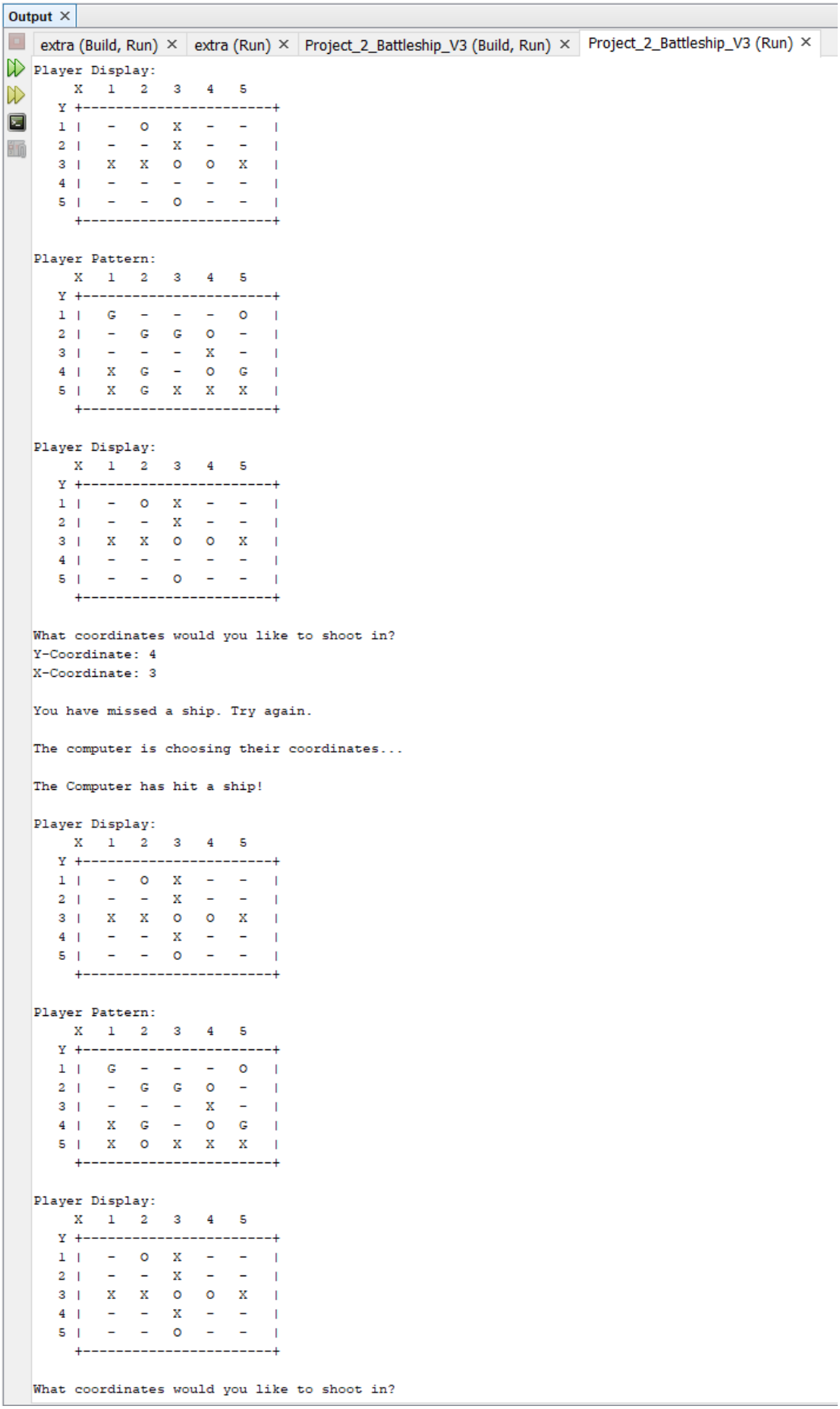

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - G |
2 | - G G G - |
3 | - - - - - |
4 | - G - G G |
5 | - G - - - |
+----------+
This is your table, would like to start playing? (y/n): y
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - - - - - |
2 | - - - - - |
3 | - - - - - |
4 | - - - - - |
5 | - - - - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 2
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - - - - |
4 | - - - - - |
5 | - - - - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - G |
2 | - G G G - |
3 | - - - - - |
4 | - G - G G |
5 | - G X - - |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - - - - |
4 | - - - - - |
5 | - - - - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 3
X-Coordinate: 4
You have hit a ship!
The computer is choosing their coordinates...

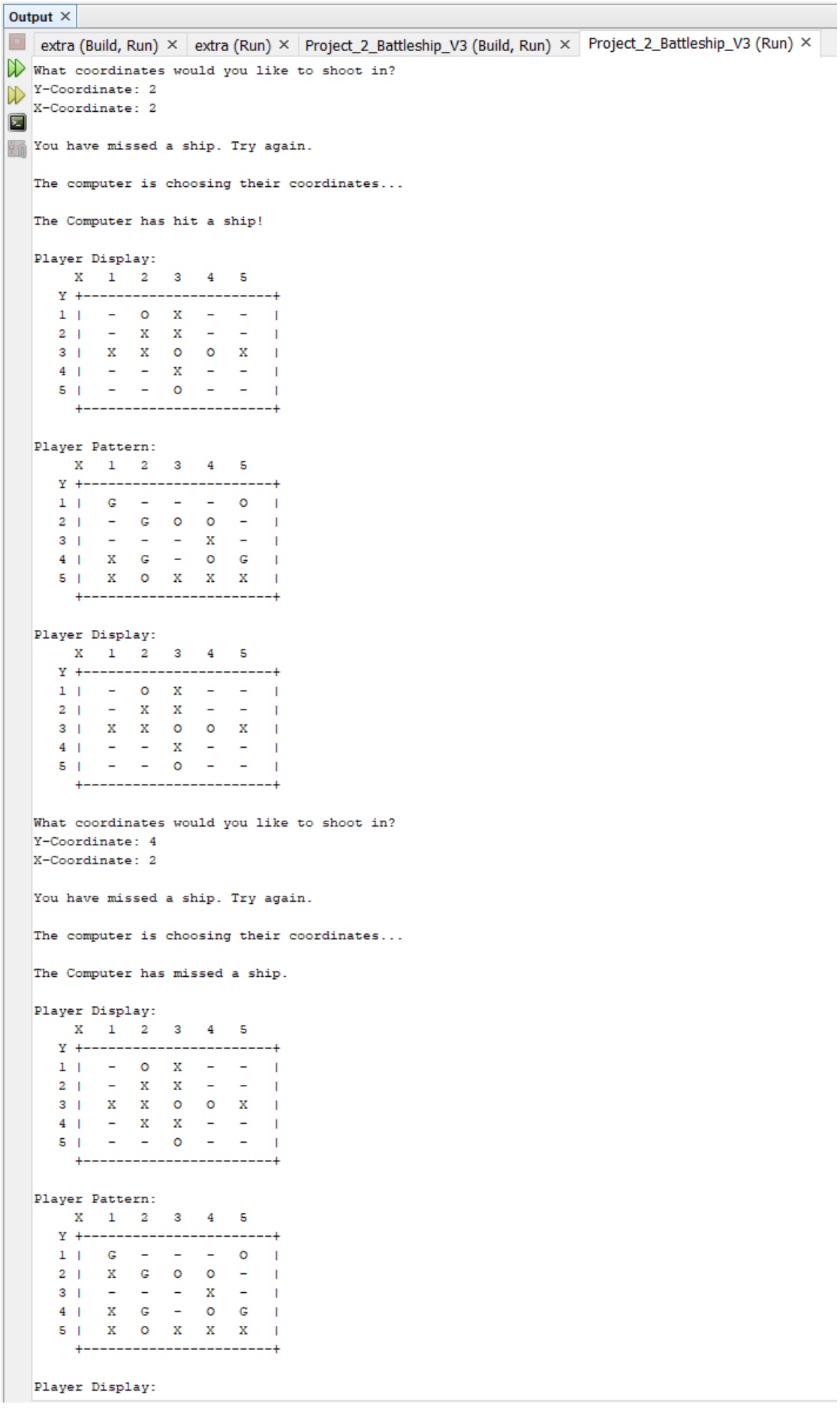

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - - O - |
4 | - - - - - |
5 | - - - - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - G |
2 | - G G G - |
3 | - - - - - |
4 | - G - G G |
5 | - G X X - |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - - O - |
4 | - - - - - |
5 | - - - - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 5
X-Coordinate: 3
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - - O - |
4 | - - - - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - G |
2 | - G G G - |
3 | - - - - - |
4 | - G - G G |
5 | X G X X - |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |


Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
What coordinates would you like to shoot in?
Y-Coordinate: 3
X-Coordinate: 3
You have hit a ship!
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - O O - |
4 | - - - - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G G - |
3 | - - - - - |
4 | - G - G G |
5 | X G X X - |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - - O O - |
4 | - - - - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 3
X-Coordinate: 2
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - X O O - |
4 | - - - - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G O - |
3 | - - - - - |
4 | - G - G G |
5 | X G X X - |
+----------+

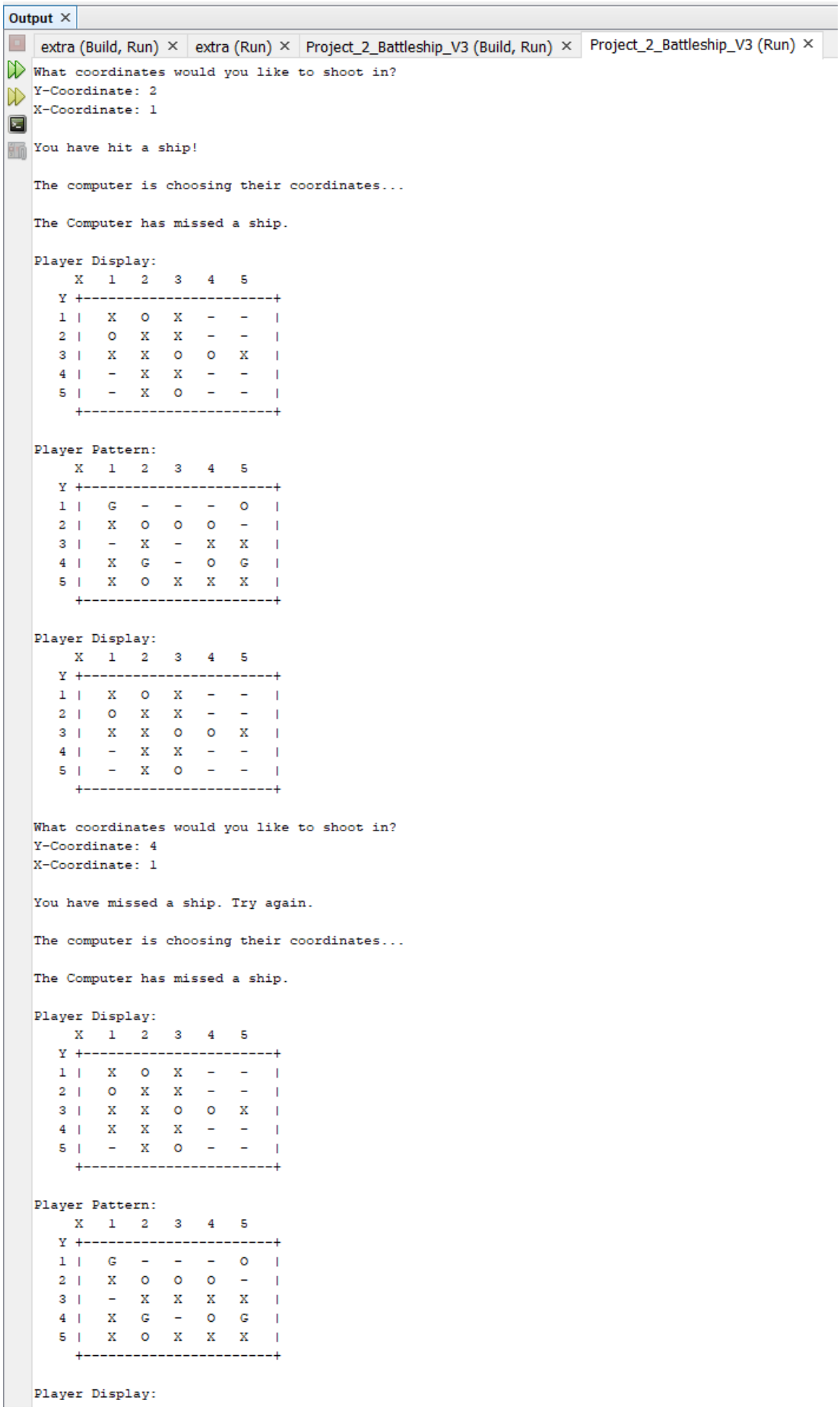

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | - X O O - |
4 | - - - - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 3
X-Coordinate: 1
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | X X O O - |
4 | - - - - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G O - |
3 | - - - - - |
4 | X G - G G |
5 | X G X X - |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | X X O O - |
4 | - - - - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 5
X-Coordinate: 3
These are invalid numbers try again...
Y-Coordinate: 3
X-Coordinate: 5
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | X X O O X |
4 | - - - - - |

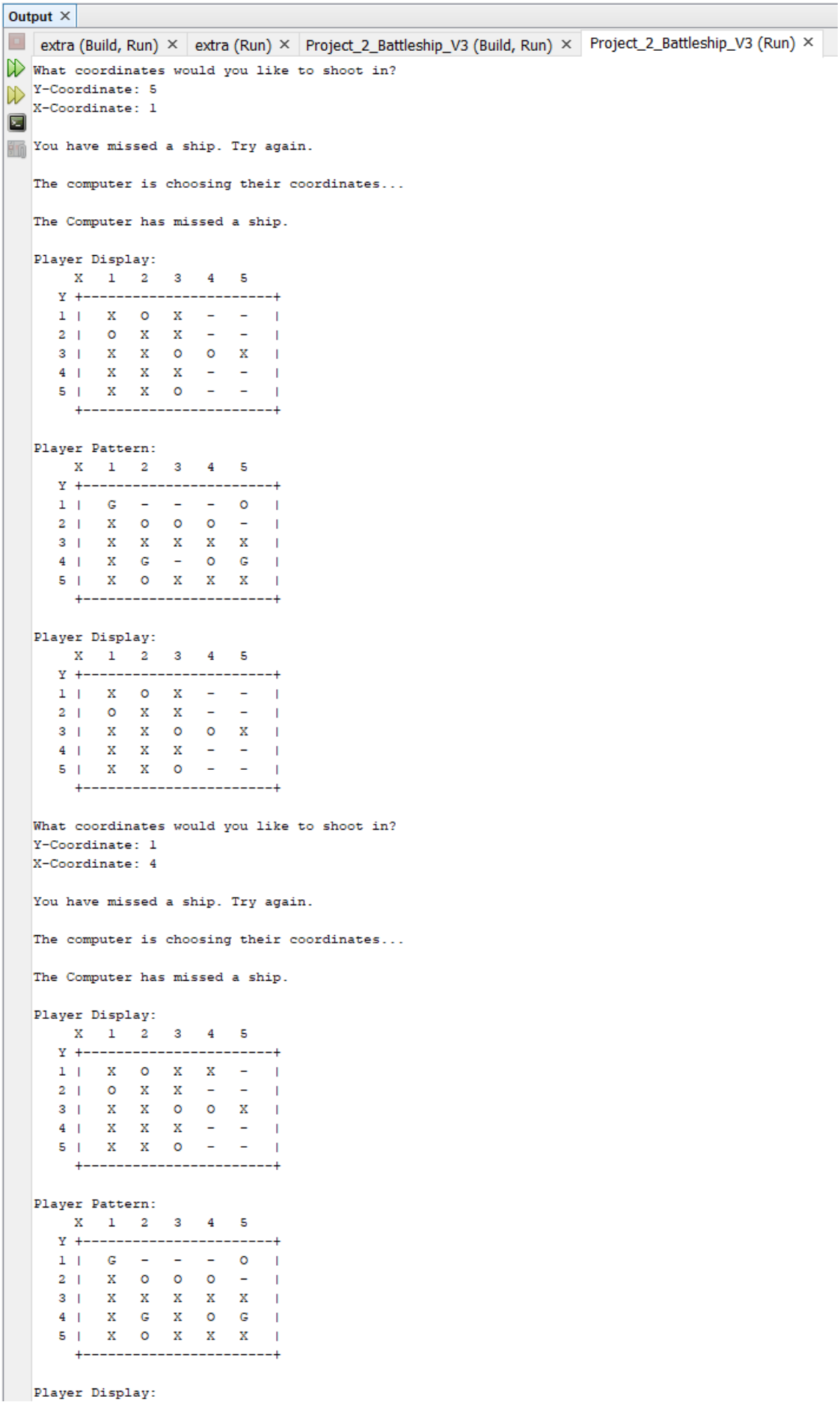

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G O - |
3 | - - - - - |
4 | X G - G G |
5 | X G X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - - - - |
3 | X X O O X |
4 | - - - - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 2
X-Coordinate: 3
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - X - - |
3 | X X O O X |
4 | - - - - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G O - |
3 | - - - X - |
4 | X G - G G |
5 | X G X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O - - - |
2 | - - X - - |
3 | X X O O X |
4 | - - - - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 3
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5


Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - - X - - |
3 | X X O O X |
4 | - - - - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G O - |
3 | - - - X - |
4 | X G - O G |
5 | X G X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - - X - - |
3 | X X O O X |
4 | - - - - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 4
X-Coordinate: 3
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - - X - - |
3 | X X O O X |
4 | - - X - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G G O - |
3 | - - - X - |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - - X - - |
3 | X X O O X |
4 | - - X - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?

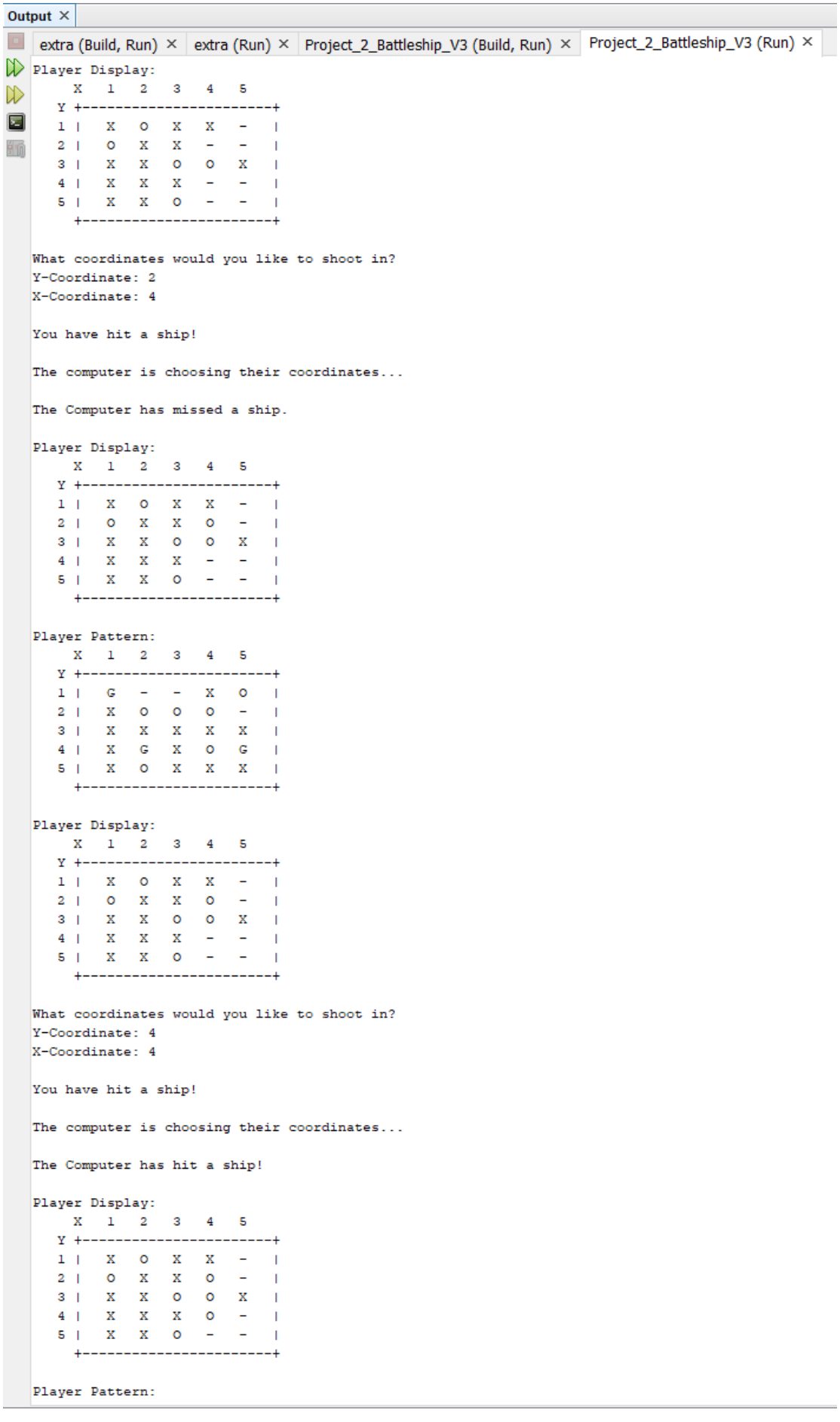

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
What coordinates would you like to shoot in?
Y-Coordinate: 2
X-Coordinate: 2
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - X X - - |
3 | X X O O X |
4 | - - X - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | - G O O - |
3 | - - - X - |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - X X - - |
3 | X X O O X |
4 | - - X - - |
5 | - - O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 4
X-Coordinate: 2
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - X X - - |
3 | X X O O X |
4 | - X X - - |
5 | - - O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | X G O O - |
3 | - - - X - |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:

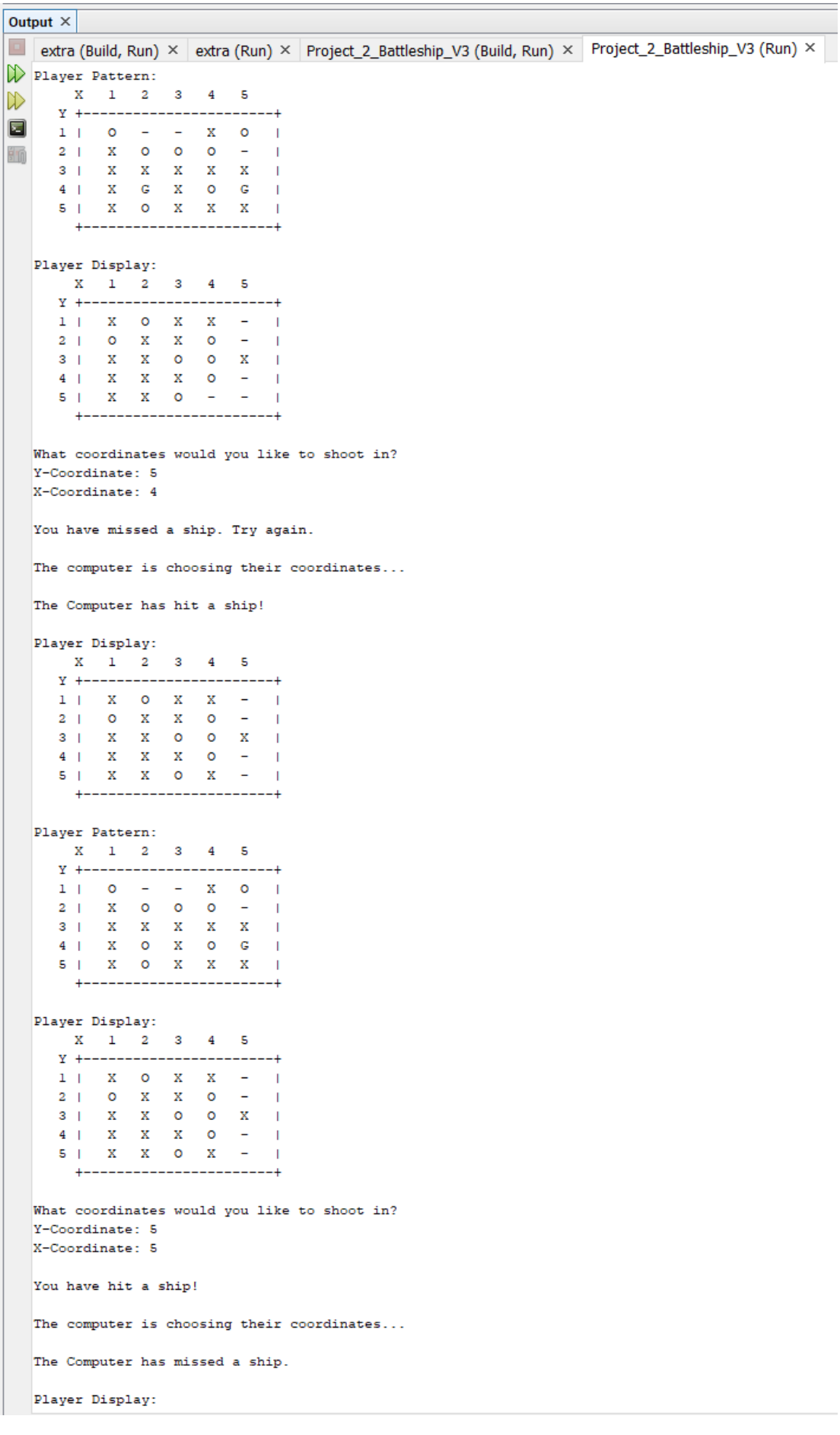

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
What coordinates would you like to shoot in?
Y-Coordinate: 5
X-Coordinate: 2
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - X X - - |
3 | X X O O X |
4 | - X X - - |
5 | - X O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | X O O O - |
3 | - - - X - |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | - O X - - |
2 | - X X - - |
3 | X X O O X |
4 | - X X - - |
5 | - X O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 1
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X - - |
2 | - X X - - |
3 | X X O O X |
4 | - X X - - |
5 | - X O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | X O O O - |
3 | - X - X - |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:

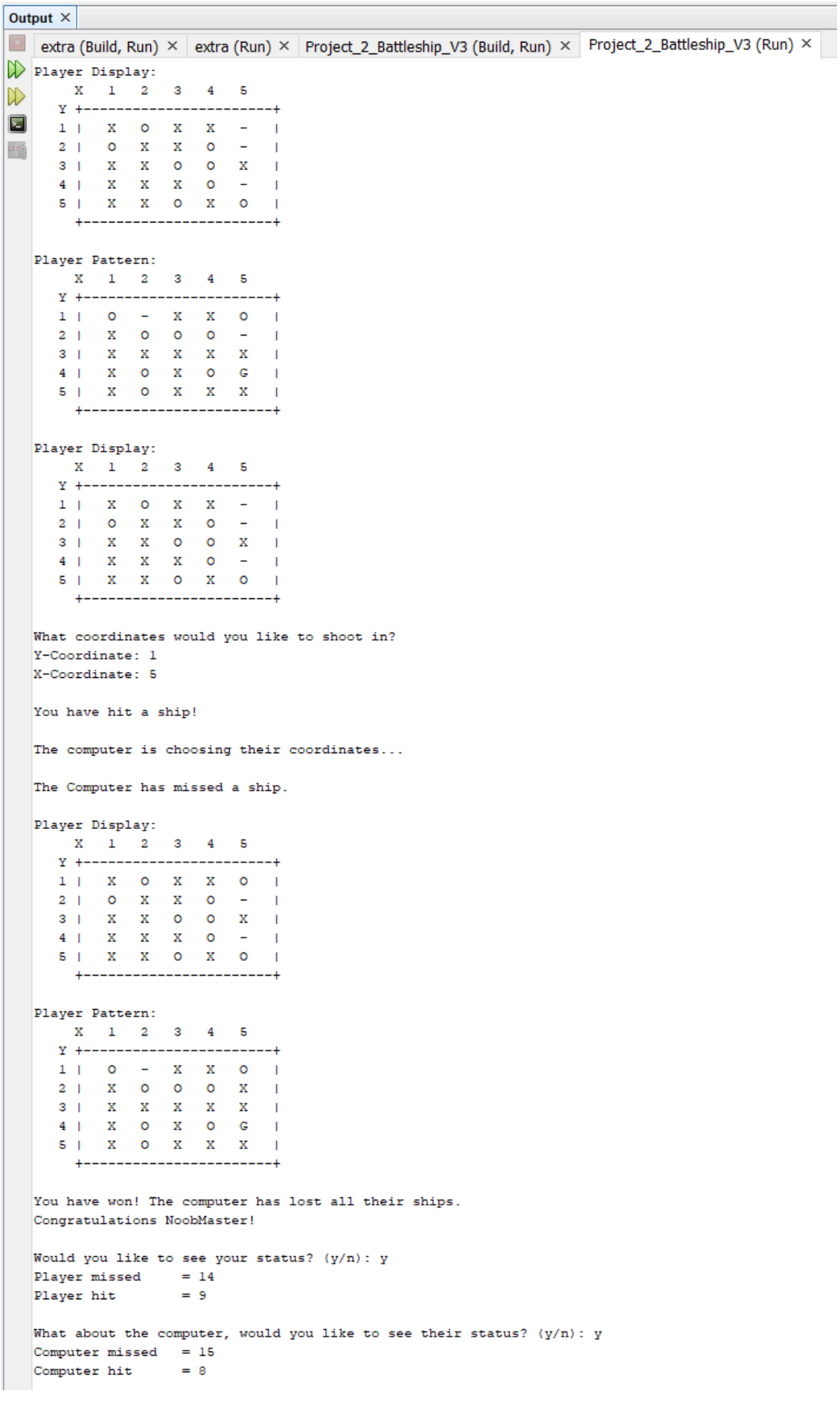

Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
What coordinates would you like to shoot in?
Y-Coordinate: 2
X-Coordinate: 1
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X - - |
2 | O X X - - |
3 | X X O O X |
4 | - X X - - |
5 | - X O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | X O O O - |
3 | - X - X X |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X - - |
2 | O X X - - |
3 | X X O O X |
4 | - X X - - |
5 | - X O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 4
X-Coordinate: 1
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X - - |
2 | O X X - - |
3 | X X O O X |
4 | X X X - - |
5 | - X O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - - O |
2 | X O O O - |
3 | - X X X X |
4 | X G - O G |
5 | X O X X X |
+----------+
Player Display:

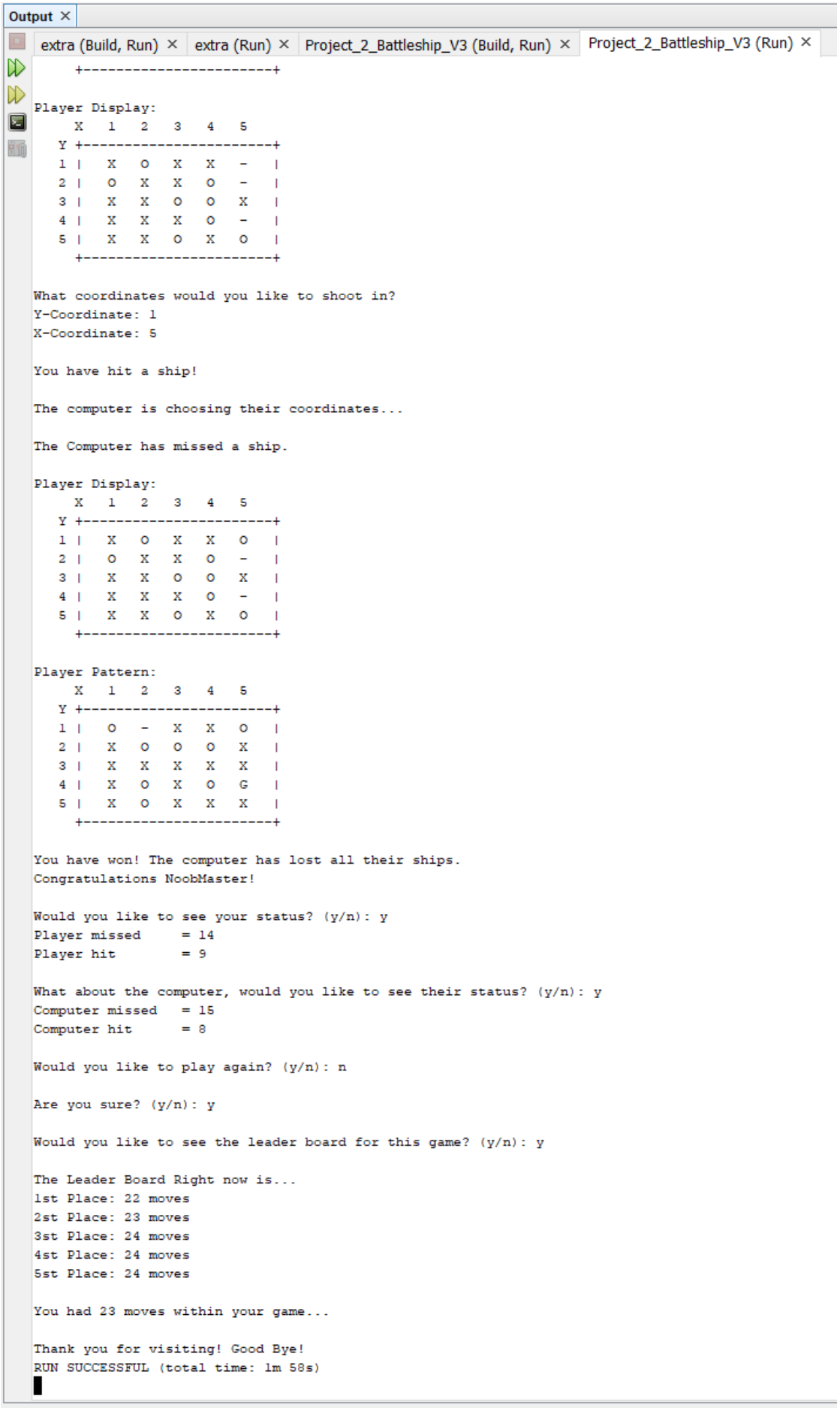


```
Output
extra (Build, Run) | extra (Run) | Project_2_Battleship_V3 (Build, Run) | Project_2_Battleship_V3 (Run)
What coordinates would you like to shoot in?
Y-Coordinate: 5
X-Coordinate: 1

You have missed a ship. Try again.

The computer is choosing their coordinates...

The Computer has missed a ship.

Player Display:
    X   1   2   3   4   5
  Y +-----------------------+
  1 |   X   O   X   -   -   |
  2 |   O   X   X   -   -   |
  3 |   X   X   O   O   X   |
  4 |   X   X   X   -   -   |
  5 |   X   X   O   -   -   |
    +-----------------------+

Player Pattern:
    X   1   2   3   4   5
  Y +-----------------------+
  1 |   G   -   -   -   O   |
  2 |   X   O   O   O   -   |
  3 |   X   X   X   X   X   |
  4 |   X   G   -   O   G   |
  5 |   X   O   X   X   X   |
    +-----------------------+

Player Display:
    X   1   2   3   4   5
  Y +-----------------------+
  1 |   X   O   X   -   -   |
  2 |   O   X   X   -   -   |
  3 |   X   X   O   O   X   |
  4 |   X   X   X   -   -   |
  5 |   X   X   O   -   -   |
    +-----------------------+

What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 4

You have missed a ship. Try again.

The computer is choosing their coordinates...

The Computer has missed a ship.

Player Display:
    X   1   2   3   4   5
  Y +-----------------------+
  1 |   X   O   X   X   -   |
  2 |   O   X   X   -   -   |
  3 |   X   X   O   O   X   |
  4 |   X   X   X   -   -   |
  5 |   X   X   O   -   -   |
    +-----------------------+

Player Pattern:
    X   1   2   3   4   5
  Y +-----------------------+
  1 |   G   -   -   -   O   |
  2 |   X   O   O   O   -   |
  3 |   X   X   X   X   X   |
  4 |   X   G   X   O   G   |
  5 |   X   O   X   X   X   |
    +-----------------------+

Player Display:
```


Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X - - |
3 | X X O O X |
4 | X X X - - |
5 | X X O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 2
X-Coordinate: 4
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X - - |
5 | X X O - - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | G - - X O |
2 | X O O O - |
3 | X X X X X |
4 | X G X O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X - - |
5 | X X O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 4
X-Coordinate: 4
You have hit a ship!
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O - - |
+----------+
Player Pattern:


Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | O - - X O |
2 | X O O O - |
3 | X X X X X |
4 | X G X O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O - - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 5
X-Coordinate: 4
You have missed a ship. Try again.
The computer is choosing their coordinates...
The Computer has hit a ship!
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X - |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | O - - X O |
2 | X O O O - |
3 | X X X X X |
4 | X O X O G |
5 | X O X X X |
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X - |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 5
X-Coordinate: 5
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:


Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
Player Display:
X 1 2 3 4 5
Y
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X O |
Player Pattern:
X 1 2 3 4 5
Y
1 | O - X X O |
2 | X O O O - |
3 | X X X X X |
4 | X O X O G |
5 | X O X X X |
Player Display:
X 1 2 3 4 5
Y
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X O |
What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 5
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y
1 | X O X X O |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X O |
Player Pattern:
X 1 2 3 4 5
Y
1 | O - X X O |
2 | X O O O X |
3 | X X X X X |
4 | X O X O G |
5 | X O X X X |
You have won! The computer has lost all their ships.
Congratulations NoobMaster!
Would you like to see your status? (y/n): y
Player missed = 14
Player hit = 9
What about the computer, would you like to see their status? (y/n): y
Computer missed = 15
Computer hit = 8


Output
extra (Build, Run)
extra (Run)
Project_2_Battleship_V3 (Build, Run)
Project_2_Battleship_V3 (Run)
+----------+
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X - |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X O |
+----------+
What coordinates would you like to shoot in?
Y-Coordinate: 1
X-Coordinate: 5
You have hit a ship!
The computer is choosing their coordinates...
The Computer has missed a ship.
Player Display:
X 1 2 3 4 5
Y +----------+
1 | X O X X O |
2 | O X X O - |
3 | X X O O X |
4 | X X X O - |
5 | X X O X O |
+----------+
Player Pattern:
X 1 2 3 4 5
Y +----------+
1 | O - X X O |
2 | X O O O X |
3 | X X X X X |
4 | X O X O G |
5 | X O X X X |
+----------+
You have won! The computer has lost all their ships.
Congratulations NoobMaster!
Would you like to see your status? (y/n): y
Player missed = 14
Player hit = 9
What about the computer, would you like to see their status? (y/n): y
Computer missed = 15
Computer hit = 8
Would you like to play again? (y/n): n
Are you sure? (y/n): y
Would you like to see the leader board for this game? (y/n): y
The Leader Board Right now is...
1st Place: 22 moves
2st Place: 23 moves
3st Place: 24 moves
4st Place: 24 moves
5st Place: 24 moves
You had 23 moves within your game...
Thank you for visiting! Good Bye!
RUN SUCCESSFUL (total time: 1m 58s)

Program

Main.cpp:

```
//System Libraries Here
#include <iostream>
#include <iomanip>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <fstream>
#include <algorithm>
#include <vector>
using namespace std;

//User Libraries Here
#include "items.h"

//Global Constants Only, No Global Variables
// Columns in the 2-D Arrays

//Like PI, e, Gravity, or conversions

//Function Prototypes Here
//Bool for choices
bool choice(char);
bool verify(char);
char valid(char);
//Sorting for vectors
bool myfunction (int, int);
bool myobject (int, int);

//**************************************************************************
// Template function For Game Rules...
//**************************************************************************

template <class T>
```

```
T rules() {
  T name;
  // Ask if player wants to play
  cout << "Hello player, what is your name?" << endl
     << "Enter name here: ";
  //Input Name
  cin.clear();
  getline(cin, name);
  //Output Rules of the game
  cout << endl;
  cout << "Hello " << name << " are you ready to listen to the rules?" << endl;
  cout << "The rules are simple.." << endl;
  cout << "The player is given nine ships which are placed at random locations..." << endl;
  cout << "The ships are then targeted by the enemy computer..." << endl;
  cout << "You are to hit the computer's ships before it gets all of yours..." << endl;
  cout << "If you lose all your ships, you lose..." << endl;
  cout << "If the computer's ships are destroyed, you win..." << endl;
  cout << endl;

  //Output the Table
  cout << "With these rules you are given 2 tables..." << endl;
  cout << "One with your display to see where you've chosen to shoot..." << endl;
  cout << "The other your own battle ships..." << endl;
  cout << "It should look similar to this..." << endl;

  //Display
  cout << endl << "Display:" << endl;
  cout << "    X  1  2  3  4  5" << endl;
  cout << "  Y +----------------------+" << endl;
  for(int rows = 0; rows < 5; rows++) {
    cout << "  " << rows + 1 << " |  ";
    for(int cols = 0; cols < 5; cols++) {
      cout << "-  ";
    }
    cout << "|" << endl;
    if(rows == 4) {
      cout << "    +----------------------+" << endl;
    }
  }
```

```
  //End the rules...
  cout << endl << "Did you get everything, " << name << "?" << endl << endl;
  //Return Value
  return name;
}

//**************************************************************************
// Main function...
//**************************************************************************

//Program Execution Begins Here
int main(int argc, char** argv) {
  //Set Random Number seed
  srand(static_cast<unsigned int>(time(0)));

  //Declare Variables
  int size = 5;     //Size of table
  int moves;        //Count for number of moves
  char stats, play;   //Characters for choices
  bool again;       //Boolean for choices
  string name;      //PLayer Name

  // Ask if player wants to play and Output game Rules
  name = rules<string>();

  //Ask If they want to play Battleship
  cout << "Would you like to play a game of Battleship? (y/n): ";
  cin >> play;

  // initialize bool operator
  try {
    again = choice(play);
  }
  catch(char error) {
    cout << "Invalid choice inputed! " << error << " is not an option!" << endl;
    cout << "Input your answer (y/n): ";
    cin >> play;
    again = choice(play);
```

```
}

// start game in loop with bool operator
while(again == true) {
  //Initialize class
  board var(size);

  //Display player's board
  var.printDisplay();
  var.printPlayer();

  // Ask User if they want to start playing
  cout << endl << "This is your table, would like to start playing? (y/n): ";
  cin >> play;

  //Exception catch for player
  try {
    play = valid(play);
  }
  catch(char error) {
    cout << "Invalid choice inputed! " << error << " is not an option!" << endl;
    cout << "Input your answer (y/n): ";
    cin >> play;
    play = valid(play);
  }

  //Loop Game
  var.game(play);

  // Output Results
  var.results(name);

  //Initialize number of moves
  moves = var.retrnMoves();

  // Ask if they would like to know their accuracy
  try {
    cout << "Would you like to see your status? (y/n): ";
    cin >> stats;
```

```
        var.pStatus(stats);
    }
    catch(char error) {
        cout << "Invalid choice inputed! " << error << " is not an option!" << endl;
        cout << "Input your answer (y/n): ";
        cin >> stats;
        var.pStatus(stats);
    }

    // Ask if they would like to know computer's accuracy
    try {
        cout << endl << "What about the computer, would you like to see their status? (y/n):
";
        cin >> stats;
        var.rStatus(stats);
    }
    catch(char error) {
        cout << "Invalid choice inputed! " << error << " is not an option!" << endl;
        cout << "Input your answer (y/n): ";
        cin >> stats;
        var.rStatus(stats);
    }

    //Ask The player if they would like to play again
    cout << endl << "Would you like to play again? (y/n): ";
    cin >> play;
    //Exception for the choice
    try {
        again = verify(play);
    }
    catch(char error) {
        cout << "Invalid choice inputed! " << error << " is not an option!" << endl;
        cout << "Input your answer (y/n): ";
        cin >> play;
        again = verify(play);
    }

    //Call class for leaderboard
    files lead(size, moves);
```

```
//Allocate memory for leaderboard
int *leaderBoard = new int[size];
leaderBoard = lead.returnLead();

//Use vector to utilize algorithm
vector<int> leadVect (leaderBoard, leaderBoard+6);
leadVect[5] = moves;

// using default comparison (operator <):
sort (leadVect.begin(), leadVect.begin()+4);

// using function as comp
sort (leadVect.begin()+4, leadVect.end(), myfunction);

// using object as comp
sort (leadVect.begin(), leadVect.end(), myobject);

//Remove last number in the vector
leadVect.pop_back();

//Assign sorted vector to leaderboard array
for(int i = 0; i < size; i++) {
  leaderBoard[i] = leadVect[i];
}

//Output array to binary file
lead.arrayToFileBin(leaderBoard);

//Print the leaderboard
try {
  //Ask If they want to see it
  cout << endl << "Would you like to see the leader board for this game? (y/n): ";
  cin >> stats;
  cout << endl;
  lead.printLead(leaderBoard, stats);
}
catch(char error) {
  cout << "Invalid choice inputed! " << error << " is not an option!" << endl;
```

```
        cout << "Input your answer (y/n): ";
        cin >> stats;
        lead.printLead(leaderBoard, stats);
      }
    }

    //Output an ending statement
    cout << endl << "Thank you for visiting! Good Bye!";

    //Exit
    return 0;
}
```

items.h:

```
#include <string>
using namespace std;

//*****************************************************************************
// Classes...
//*****************************************************************************

class data {
protected:
    //Player and Computer's Choices
    int pChoseX, pChoseY, rChoseX, rChoseY;
    //Player and Computer's hits and miss
    int pCount, rCount, pMiss, pHit, rMiss, rHit;
    //Static Int for number of boats
    static int nBoats;
    //Int for number of moves to put in leader board
    int nMoves;
    friend class files;
public:
    data();
    void results(string);
```

```
   void pStatus(char);
   void rStatus(char);
   int retrnMoves();
};

class board: public data {
private:
   char **player,
        **comp,
        **display;
   int  size;
public:
   board(int);
   ~board();
   char **resetDisplay();
   char **resetPattern();
   void printDisplay();
   void printPlayer();
   void game(char);
   int pMissed();
   int pAccuracy();
   int rMissed();
   int rAccuracy();
   void wrtTxt();
};

class files {
private:
   int stats;
   int size;
   int *leader;
public:
   files(int, int);
   ~files();
```

```
  int arrayToFileBin(int *);
  int *fileToArray();
  int *returnLead();
  void printLead(int *, char);
};
```

Items.cpp:

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;

#include "items.h"

//*************************************************************************
// Functions for boolean operators and try/catch functions...
//*************************************************************************

bool choice(char play) {
  if(play == 'y' || play == 'Y' || play == 'n' || play == 'N') {
    switch(play) {
      case 'y':
        return true;
        break;
      case 'n':
        return false;
        break;
      case 'Y':
        return true;
        break;
      case 'N':
        return false;
        break;
```

```
    }
  }
  else {
    throw play;
  }
  return false;
}

bool verify(char play) {
  if(play == 'n' || play == 'N') {
    cout << endl << "Are you sure? (y/n): ";
    cin >> play;

    if(play == 'y' || play == 'Y') {
      return false;
    }
    else {
      return true;
    }
  }
  else if(play == 'y' || play == 'Y') {
    return true;
  }
  else {
    throw play;
  }
  return false;
}

char valid(char play) {
  if(play == 'y' || play == 'Y') {
    return play;
  }
  else if(play == 'n' || play == 'N') {
```

```
    while(play == 'n' || play == 'N') {
      cout << endl << "What about now? (y/n): ";
      cin >> play;
      cout << endl;
    }
  }
  else {
    throw play;
  }
  return play;
}

//*******************************************************************************
// Functions for Vector sorting...
//*******************************************************************************

bool myfunction (int i, int j) {
  return (i < j);
}

bool myobject (int i, int j) {
  return (i < j);
}

//*******************************************************************************
// Constructors...
//*******************************************************************************

data::data() {
  //Initialize
  pChoseX = 0;
  pChoseY = 0;
  rChoseX = 0;
  rChoseY = 0;
```

```
  //Player and Computer's hits and miss
  pCount = 9;
  rCount = 9;
  //initialize number of moves
  nMoves = 0;
}

board::board(int s) {
  size = s;
  display = resetDisplay();
  player = resetPattern();
  comp = resetPattern();
}

//*************************************************************************
// Static Member for Number of Boats...
//*************************************************************************

int data::nBoats = 9;

//*************************************************************************
// Destructor...
//*************************************************************************

board::~board() {
  //Deallocate Display
  for(int i = 0; i < size; i++){
    delete [] display[i];
  }
  delete [] display;
  //Deallocate player
  for(int i = 0; i < size; i++){
    delete [] player[i];
  }
```

```
  delete [] player;
  //Deallocate Comp
  for(int i = 0; i < size; i++){
    delete [] comp[i];
  }
  delete [] comp;
}

//*****************************************************************************
// Initialize values of the board using another function...
//*****************************************************************************

char **board::resetDisplay() {
  //Initialize dynamic array
  char **a = new char*[size];
  //Allocate the memory for each row
  for(int row = 0; row < size; row++) {
    a[row] = new char[size];
  }
  //Initialize Arrays with "-" to fill with ships
  for(int rows = 0; rows < size; rows++) {
    for(int cols = 0; cols < size; cols++) {
      a[rows][cols] = '-';
    }
  }
  //Return values
  return a;
  //Deallocate Values
  for(int i = 0; i < size; i++){
    delete [] a[i];
  }
  delete [] a;
}
```

```
char **board::resetPattern() {
  //Initialize dynamic array
  char **a = new char*[size];
  int x, y;
  //Allocate the memory for each row
  for(int row = 0; row < size; row++) {
    a[row] = new char[size];
  }
  //Initialize Arrays with "-" to fill with ships
  for(int rows = 0; rows < size; rows++) {
    for(int cols = 0; cols < size; cols++) {
      a[rows][cols] = '-';
    }
  }
  //Initialize the patterns for the player
  for(int boats = 0; boats < nBoats; boats++) {
    x = rand() % 5;
    y = rand() % 5;
    while(a[x][y] == 'G') {
      x = rand() % 5;
      y = rand() % 5;
    }
    a[x][y] = 'G';
  }
  //Return values
  return a;
  //Deallocate Values
  for(int i = 0; i < size; i++){
    delete [] a[i];
  }
  delete [] a;
}

//****************************************************************************
```

```
// Print Player's pattern and Player's Display
//*****************************************************************************

void board::printDisplay() {
   cout << endl << "Player Display:" << endl;
   cout << "    X  1  2  3  4  5" << endl;
   cout << "  Y +----------------------+" << endl;
   for(int rows = 0; rows < size; rows++) {
      cout << "  " << rows + 1 << " |  ";
      for(int cols = 0; cols < size; cols++) {
         cout << display[rows][cols] << "  ";
      }
      cout << "|" << endl;
      if(rows == size - 1) {
         cout << "    +----------------------+" << endl;
      }
   }
}

void board::printPlayer() {
   cout << endl << "Player Pattern:" << endl;
   cout << "    X  1  2  3  4  5" << endl;
   cout << "  Y +----------------------+" << endl;
   for(int rows = 0; rows < size; rows++) {
      cout << "  " << rows + 1 << " |  ";
      for(int cols = 0; cols < size; cols++) {
         cout << player[rows][cols] << "  ";
      }
      cout << "|" << endl;
      if(rows == size - 1) {
         cout << "    +----------------------+" << endl;
      }
   }
}
```

```
//*******************************************************************************
// Function for the entire game
//*******************************************************************************

void board::game(char play) {
  while(play == 'y' || play == 'Y') {
    // Display player's choices
    printDisplay();

    // Ask player where they would like to hit and validate
    cout << endl << "What coordinates would you like to shoot in?" << endl;

    //Exception check for Y-Coordinate
    try {
      //Input Y
      cout << "Y-Coordinate: ";
      cin >> pChoseY;
      // Validate User input if choice is invalid
      if(pChoseY > 5 || pChoseY <= 0) {
        throw pChoseY;
      }
    }
    catch(int error) {
      cout << endl << "The Y-Coordinate " << error << " is invalid! Try again..." << endl;
      //Do-While loop for input
      do {
      cout << "Y-Coordinate: ";
      cin >> pChoseY;
      } while (pChoseY > 5 || pChoseY <= 0);
    }

    //Exception Check for X-Coordinate
    try {
```

```
        //Input X
        cout << "X-Coordinate: ";
        cin >> pChoseX;
        // Validate User input if choice is invalid
        if(pChoseX > 5 || pChoseX <= 0) {
          throw pChoseX;
        }
      }
      catch(int error) {
        cout << endl << "The X-Coordinate " << error << " is invalid! Try again..." << endl;
        //Do-While loop for input
        do {
        cout << "X-Coordinate: ";
        cin >> pChoseX;
        } while (pChoseX > 5 || pChoseX <= 0);
      }

      //Loop If try The choice is already taken...
      if(display[pChoseY - 1][pChoseX - 1] == 'O' || display[pChoseY - 1][pChoseX - 1] == 'X')
{
        do {
          cout << endl << "These are invalid numbers try again..." << endl;
          cout << "Y-Coordinate: ";
          cin >> pChoseY;
          cout << "X-Coordinate: ";
          cin >> pChoseX;
        } while (pChoseX > 5 || pChoseY > 5 || pChoseX <= 0 || pChoseY <= 0 ||
comp[pChoseY - 1][pChoseX - 1] == 'O' || comp[pChoseY - 1][pChoseX - 1] == 'X');
      }

      // See if player hit a ship or not
      if(comp[pChoseY - 1][pChoseX - 1] != '-') {
        cout << endl << "You have hit a ship!" << endl;
        display[pChoseY - 1][pChoseX - 1] = 'O';
```

```
  rCount -= 1;
  nMoves++;
}
else if(comp[pChoseY - 1][pChoseX - 1] == '-') {
  cout << endl << "You have missed a ship. Try again." << endl;
  display[pChoseY - 1][pChoseX - 1] = 'X';
  nMoves++;
}

// Output that computer is choosing
cout << endl << "The computer is choosing their coordinates..." << endl;

// use random and linear search to see if computer chose that number already;
rChoseX = rand() % 5;
rChoseY = rand() % 5;
//Loop so it doesn't choose the same coordinate again
while(player[rChoseY][rChoseX] == 'X' || player[rChoseY][rChoseX] == 'O') {
  rChoseX = rand() % 5;
  rChoseY = rand() % 5;
}

// See if Computer hit a ship or not
if(player[rChoseY][rChoseX] != '-') {
  cout << endl << "The Computer has hit a ship!" << endl;
  player[rChoseY][rChoseX] = 'O';
  pCount -= 1;
}
else if(player[rChoseY][rChoseX] == '-') {
  cout << endl << "The Computer has missed a ship." << endl;
  player[rChoseY][rChoseX] = 'X';
}

// Output player's table
printDisplay();
```

```
      printPlayer();


      //If statement to end game if there are no more boats left for one or both sides
      if(rCount == 0 || pCount == 0) {
         play = 'n';
      }

   }

   // Use linear search to determine how much you missed and how much you hit.
   // Calculate player and computer's accuracy
   pMiss = pMissed();
   pHit = pAccuracy();
   rMiss = rMissed();
   rHit = rAccuracy();

   //Write Both results and txt
   wrtTxt();
}

//*********************************************************************************
// Function for results...
//*********************************************************************************

int data::retrnMoves() {
   //return Number of moves to main
   return nMoves;
}

//*********************************************************************************
// Function for results...
//*********************************************************************************
```

```
void data::results(string name) {
  if(rCount == 0) {
    cout << endl << "You have won! The computer has lost all their ships."
      << endl << "Congratulations " << name << "!" << endl << endl;
  }
  else if(pCount == 0){
    cout << endl << "You have lost! The computer has won and destroyed all your ships."
      << endl << "Better luck next time " << name << "!" << endl << endl;
  }
  else if(pCount == rCount){
    cout << endl << "You have tied! your ships and the computer's ships are all destroyed."
      << endl << "Better luck next time " << name << "!" << endl << endl;
  }
}

//********************************************************************************
// Function for Player and Computer Stats...
//********************************************************************************

int board::pMissed() {
  //Initialize to 0
  pMiss = 0;
  //Count how much is missed
  for(int i = 0; i < size; i++) {
    for(int u = 0; u < size; u++) {
      if(display[i][u] == 'X') {
        pMiss += 1;
      }
    }
  }
  //return results
  return pMiss;
}
```

```
int board::pAccuracy() {
  //Initialize to 0
  pHit = 0;
  //Count how much is hit
  for(int i = 0; i < size; i++) {
    for(int u = 0; u < size; u++) {
      if(display[i][u] == 'O') {
        pHit += 1;
      }
    }
  }
  //return results
  return pHit;
}

int board::rMissed() {
  //Initialize to 0
  rMiss = 0;
  //Count how much is missed
  for(int i = 0; i < size; i++) {
    for(int u = 0; u < size; u++) {
      if(player[i][u] == 'X') {
        rMiss += 1;
      }
    }
  }
  //return results
  return rMiss;
}

int board::rAccuracy() {
  //Initialize to 0
  rHit = 0;
  //Count how much is hit
```

```
  for(int i = 0; i < size; i++) {
    for(int u = 0; u < size; u++) {
      if(player[i][u] == 'O') {
        rHit += 1;
      }
    }
  }
  //return results
  return rHit;
}

void data::pStatus(char stats) {
  if(stats == 'y' || stats == 'Y') {
    cout << "Player missed    = " << pMiss << endl;
    cout << "Player hit       = " << pHit << endl;
  }
  else if(stats == 'n' || stats == 'N'){
    cout << "That's a shame!" << endl;
  }
  else {
    throw stats;
  }
}

void data::rStatus(char stats) {
  if(stats == 'y' || stats == 'Y') {
    cout << "Computer missed   = " << rMiss << endl;
    cout << "Computer hit     = " << rHit << endl;
  }
  else if(stats == 'n' || stats == 'N'){
    cout << "That's a shame!" << endl;
  }
  else {
    throw stats;
```

```
  }
}

//*******************************************************************************
// Write The match results to a binary file...
//*******************************************************************************

void board::wrtTxt(){
  //Declare fstream variables
  fstream out;
  out.open("data.txt",ios::in|ios::out);

  //output Diagrams
  out << endl << "Player Display:" << endl;
  out << "    X  1  2  3  4  5" << endl;
  out << "  Y +----------------------+" << endl;
  for(int rows = 0; rows < size; rows++) {
    out << "   " << rows + 1 << " |  ";
    for(int cols = 0; cols < size; cols++) {
      out << display[rows][cols] << "  ";
    }
    out << "|" << endl;
    if(rows == size - 1) {
      out << "    +----------------------+" << endl;
    }
  }

  out << endl << "Player Pattern:" << endl;
  out << "    X  1  2  3  4  5" << endl;
  out << "  Y +----------------------+" << endl;
  for(int rows = 0; rows < size; rows++) {
    out << "   " << rows + 1 << " |  ";
    for(int cols = 0; cols < size; cols++) {
      out << player[rows][cols] << "  ";
```

```
      }
      out << "|" << endl;
      if(rows == size - 1) {
        out << "    +----------------------+" << endl;
      }
    }
    //Output Results
    out << "Player missed    = " << pMiss << endl;
    out << "Player hit       = " << pHit << endl;
    out << "Computer missed   = " << rMiss << endl;
    out << "Computer hit     = " << rHit << endl;
}

//*************************************************************************
// Functions for files class...
//*************************************************************************

files::files(int s, int c) {
    //size of the leaderboard
    size = s;
    stats = c;
    leader = fileToArray();
}

files::~files() {
    delete [] leader;
}

int files::arrayToFileBin(int *a) {
    //Place Holder for return value
    int placeHolder = 0;
    //Open Binary File
    fstream out;
    out.open("data.bin",ios::in|ios::out|ios::binary);
```

```
   //Output Array contents into binary file
   out.write(reinterpret_cast<char *>(a),size*sizeof(int));
   //Close Binary File
   out.close();
   //Return something so there isn't any warnings
   return placeHolder;
}

int *files::fileToArray() {
   //Open Binary File
   fstream in;
   in.open("data.bin",ios::in|ios::out|ios::binary);
   //Initialize cursor and array
   long cursor = 0;
   int *array = new int[size];
   //Read file
   in.seekg(cursor,ios::beg);
   in.read(reinterpret_cast<char *>(array),size*sizeof(int));
   //Close Files
   in.close();
   //Return array
   return array;
}

int *files::returnLead() {
   return leader;
}

void files::printLead(int *a, char stat) {
   if(stat == 'y' || stat == 'Y') {
      cout << "The Leader Board Right now is..." << endl;
      cout << "1st Place: " << a[0] << " moves" << endl;
      cout << "2st Place: " << a[1] << " moves" << endl;
      cout << "3st Place: " << a[2] << " moves" << endl;
```

```
      cout << "4st Place: " << a[3] << " moves" << endl;
      cout << "5st Place: " << a[4] << " moves" << endl;
      cout << endl << "You had " << stats << " moves within your game..." << endl;
    }
    else if(stat == 'n' || stat == 'N'){
      cout << "Well That's a shame!" << endl;
    }
    else {
      throw stat;
    }
}
```