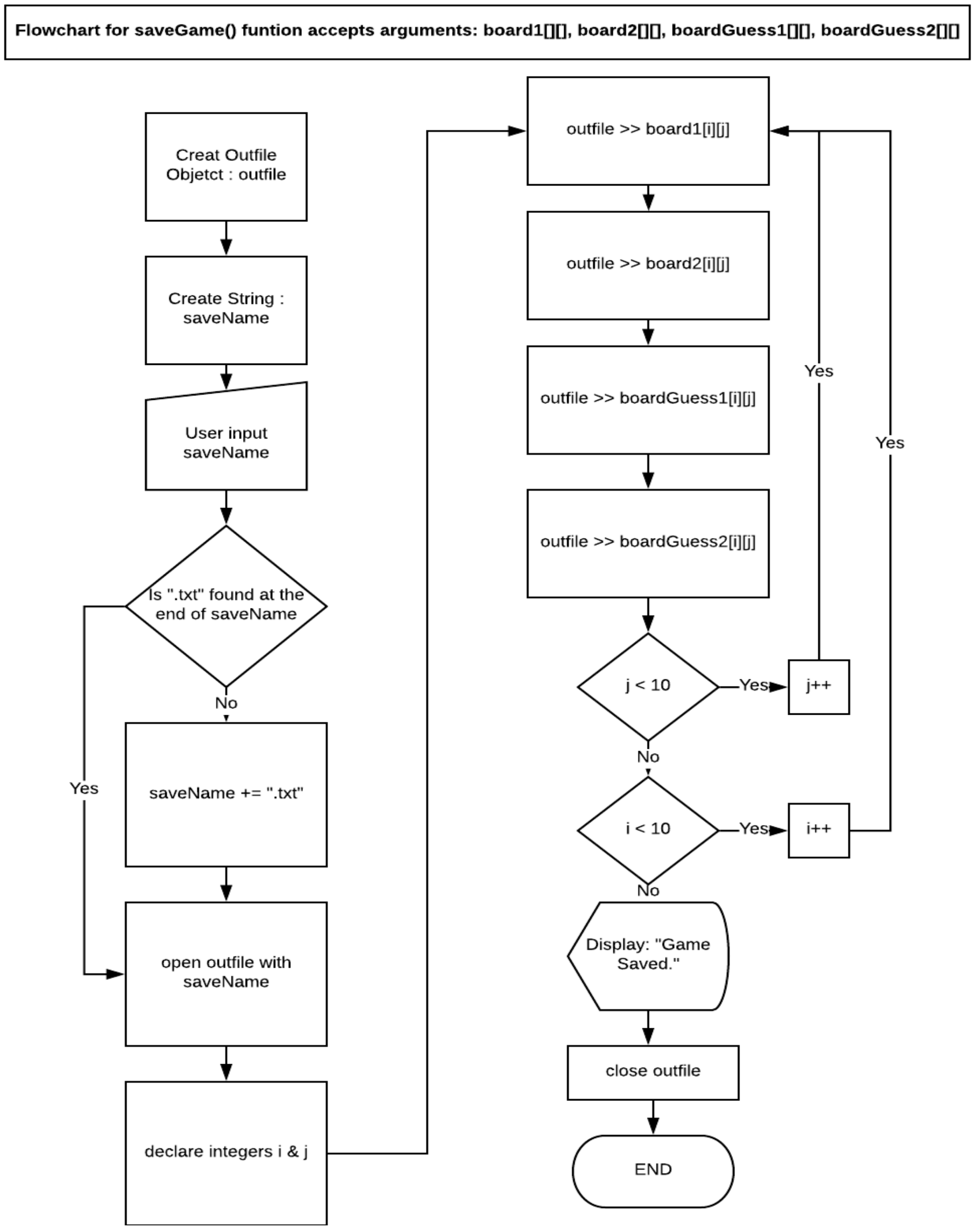
Battleship: Final Project

Cole Nordmann, Zach Johnson, Trevon Prude

CSI 230

5/6/18

Use of Flow Chart



Flowchart for saveGame() funtion accepts arguments: board1[][], board2[][], boardGuess1[][], boardGuess2[][]
Creat Outfile Objetct : outfile
Create String : saveName
User input saveName
Is ".txt" found at the end of saveName
No
Yes
saveName += ".txt"
open outfile with saveName
declare integers i & j
outfile >> board1[i][j]
outfile >> board2[i][j]
outfile >> boardGuess1[i][j]
outfile >> boardGuess2[i][j]
j < 10
Yes
j++
Yes
No
i < 10
Yes
i++
Yes
No
Display: "Game Saved."
close outfile
END

Appropriate Data Types:

Example taken from playGame() function. Char used for clear and easy user input that is later converted to an integer for position1. Position 2 entered by user. Counter used for win condition check.

```
char temp;
int position1, position2, counter;
```

Appropriate Data Structures:

Example taken from main() function. 2 dimensional arrays used for play boards.

```
const int SIZE = 10; // constant size for boards
// --- four 2D arrays for play boards -------
char BoardP1[SIZE][SIZE];
char BoardP2[SIZE][SIZE];
char BoardGuessP1[SIZE][SIZE];
char BoardGuessP2[SIZE][SIZE];
```

Appropriate Control Structures

Example taken from Battleshipai::generateDirection() function from the Battleshipai class. Switches a random number between 1 and 4 to allow the computer to slect a random direction for its ship placement.

```
	switch (rand() % 4)
	{
	case 0:
		return "North";
		break;
	case 1:
		return "East";
		break;
	case 2:
		return "South";
		break;
	case 3:
		return "West";
		break;
	}
```

Use of OO programming

```
// -- Pure Abstract Class Declaration ------------------------------------------------
class GeneralShipPlacement
{
	// protected member variables
protected:
	int xposition1;
	int xposition2;
	string yposition;
public:
	// pure virtual member funtions
	virtual void SetxpositionAircraft(char Board[][10], int SIZE) = 0;
	virtual void SetypositionAircraft(char Board[][10], int SIZE) = 0;
	virtual void SetxpositionBattleship(char Board[][10], int SIZE) = 0;
	virtual void SetypositionBattleship(char Board[][10], int SIZE) = 0;
	virtual void SetxpositionCruiser(char Board[][10]) = 0;
	virtual void setypositionCruiser(char Board[][10], int SIZE) = 0;
	virtual void SetxpositionSubmarine(char Board[][10]) = 0;
	virtual void SetypositionSubmarine(char Board[][10], int SIZE) = 0;
	virtual void SetxpositionDestroyer(char Board[][10]) = 0;
	virtual void SetypositionDestroyer(char Board[][10], int SIZE) = 0;
};
// -- Ship Placement Class : inherits General Ship Placement -----------------------
class ShipPlacement : public GeneralShipPlacement
{
public:
	// virtual funtion overidders
	virtual void SetxpositionAircraft(char Board[][10], int SIZE);
	virtual void SetypositionAircraft(char Board[][10], int SIZE);
	virtual void SetxpositionBattleship(char Board[][10], int SIZE);
	virtual void SetypositionBattleship(char Board[][10], int SIZE);
	virtual void SetxpositionCruiser(char Board[][10]);
	virtual void setypositionCruiser(char Board[][10], int SIZE);
	virtual void SetxpositionSubmarine(char Board[][10]);
	virtual void SetypositionSubmarine(char Board[][10], int SIZE);
	virtual void SetxpositionDestroyer(char Board[][10]);
	virtual void SetypositionDestroyer(char Board[][10], int SIZE);
};
// -- Battleship AI class : inherits General ship Placement Class -----------------------
---
class Battleshipai : public GeneralShipPlacement
{
public:
	string generateDirection();

	virtual void SetxpositionAircraft(char Board[][10], int SIZE);
	virtual void SetypositionAircraft(char Board[][10], int SIZE);
	virtual void SetxpositionBattleship(char Board[][10], int SIZE);
	virtual void SetypositionBattleship(char Board[][10], int SIZE);
	virtual void SetxpositionCruiser(char Board[][10]);
	virtual void setypositionCruiser(char Board[][10], int SIZE);
	virtual void SetxpositionSubmarine(char Board[][10]);
	virtual void SetypositionSubmarine(char Board[][10], int SIZE);
	virtual void SetxpositionDestroyer(char Board[][10]);
	virtual void SetypositionDestroyer(char Board[][10], int SIZE);
};
```

Read write to file

Example taken from loadGame() function. Reads info from a file to load a game that was previously saved.

```
ifstream infile; // creates infile object

string loadName; // variable for load file name

                              // user enters name of file they wish to load
cout << "Enter the name of the game file you wish to load: ";
cin.clear();
cin >> loadName;

// checks if user specified .txt
if (loadName.find(".txt") > loadName.length())
{
       loadName += ".txt"; // adds .txt if not
}

infile.open(loadName); // opens load file

if (!infile) // checks for file open error
{
       cout << "Save file does not exist." << endl;
       infile.close();
       return false; // returns false to indcate the load failed
}

// loops though all elements saved in load file
for (int i = 0; i < size; i++)
{
       // fills all boards with those respective elements
       for (int j = 0; j < size; j++)
       {
              infile >> board1[i][j];
              if (board1[i][j] == '\b')
                     board1[i][j] = ' ';

              infile >> board2[i][j];
              if (board2[i][j] == '\b')
                     board2[i][j] = ' ';

              infile >> boardGuess1[i][j];
              if (boardGuess1[i][j] == '\b')
                     boardGuess1[i][j] = ' ';

              infile >> boardGuess2[i][j];
              if (boardGuess2[i][j] == '\b')
                     boardGuess2[i][j] = ' ';
       }
}

infile >> playerTurn;

infile.close(); // closes file
return true; // return true when load is completed
```

Input Validation

Example taken from ShipPlacement::SetxpositionAircraft() function in the ship placement class.

```
char temp;
	cout << "Where would you like to place your Aircraft Carrier (Takes 5 space)?
	enter Ex. A 5" << endl;
	cin >> temp >> xposition2;
	xposition1 = charToInt(temp);
	while (xposition1 < 0 || xposition1 > 10)
	{
		cout << "Invalid grid for the placed ship, the aircraft carrier is 5 pegs
		long try again" << endl;
		cin >> temp >> xposition2;
		xposition1 = charToInt(temp);
	}
```

Indentation and comments

For indentation and comments please see all other examples in this report as well as the project as a whole.

Modularization

The following functions were all created for the purpose of modularization. Display() prints the board that is passed to it. Create board fills the board passed to it with spaces for the start of the game. DisplayMenu() prints the menu to the screen. DisplayBattleship() prints the ascii battleship title card. And both playGame() and playGameVai() run through the actual playing of the game in the main.

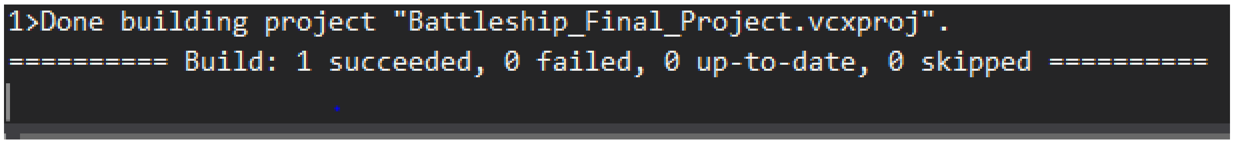
```
void Display(char Board[][10], int SIZE);
void CreateBoard(char Board[][10], int SIZE);
void displayMenu();
void DisplayBattleship();

void playGame(char BoardP1[][10], char BoardP2[][10], char BoardGuessP1[][10], char
BoardGuessP2[][10], int SIZE, int);

void playGameVai(char BoardP1[][10], char BoardP2[][10], char BoardGuessP1[][10], char
BoardGuessP2[][10], int SIZE);
```

Syntax and logical error free execution.

The program runs without any fatal errors.



```
1>Done building project "Battleship_Final_Project.vcxproj".
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```