Benjamin Poile/Brevin Simon
11565739/11572415
CS 223 - PA4
PA4 Report

A. Problem Statement
    a. Our goal was to implement a simple board game. The length of the square board must be greater than the amount of players. Players may move horizontally, vertically and diagonally (45˚). If Player A lands in a square where another player, Player B is, the Player B gets overwritten and is removed. The last player remaining is the winner.

B. Experimental Setup
    a. Machine Specs: MacBook (Retina, 12-inch, 2017); 1.4 GHz Intel i7; 16GB RAM
    b. OS and environment used when testing: MacOS Mojave 10.14; C++ 11
    `g++ *.cpp -lstdc++ -std=c++11 -o new; ./new`

C. Algorithm Design
    a. Player objects stored user ID's and coordinate locations as integers. Board objects checked and manipulated Player objects based on game logic/rules. In favor of memory and speed, we decided not to load the entire board into memory. Instead, we only stored occupied locations as Player objects. We needed to implement this project utilizing a form of a Binary Search Tree, either in the form of a STL set, map, multiset, or multimap. Our decision was to use a map, as it's functionality of key/value pairs lends itself nicely to user ID's and Player objects.

Function Name	Runtime Complexity
getX(), getY(), getID(), setID(), setXY() ,Insert()	O(1)
Remove(), Find(), MoveTo(), PrintByID()	O(n)

    The memory complexity of this PA is O(N).

D. Experimental Results and discussion.
    a. See *output.txt*