**Report**

CPTS\_233 PA #5

Xuanfu Huang & Refo Yudhanto

***A: Problem statement.***

In this project, we need to implementing a simple board game. For this project, we are going to use the STL data structures for balanced binary search trees.

***B: Experimental setup*.**

Windows, Microsoft Visual Studio 2015, Linux Ubuntu

***C:Algorithm design.***

*Class Board:*

1. container:

Memory Complexity O(N)

map <ID, player> Player O(N)

map< “(X,Y)”, ID> Positions O(N)

1. function:

Time Complexity

Insert (ID, X, Y) O(logN)

Check is board full, out of board and others O(1)

Player.find (ID) O(logN)

Positions.find(“ (X,Y) ”) O(logN)

if (all satisfied)

Player.insert(<ID, player>) O(logN)

Positions.insert(<“(X,Y)” , ID>) O(logN)

N++

return true

return false

Remove(ID) O(logN)

iter=Player.find (ID) O(logN)

if (iter!= Player.end )

positions<- iter.player.positions O(1)

Player.erase (iter) O(1)

Positions.erase(positions) O(logN)

N--;

return true

return false

MoveTo (ID, X2, Y2) O(logN)

within board check O(1)

iter=Player.find (ID) O(logN)

if (iter!= Player.end )

Positions<- iter.player.positions O(1)

check is movement legal O(1)

if (all satisfied)

iter2=Positions.find(“(X2,Y2)”) O(logN)

ID2=iter2.ID;

Remove(ID2); O(logN)

Remove(ID) O(logN)

Insert(ID) O(logN)

return true

return false

PrintByID: O(N)

in-order traversal of Player