Project 7 Report

1. Notable obstacles
   1. Previously, I would fail test case 13 of code board because my code’s isGameOver just checked if the number of turns is equal to 0. It did not check if one of the mark methods was called on, so it would not properly handle a case where someone manually sets the number of turns to 0 but no call on who’s the winner or if there was a tie was made. To fix this, I changed my isGameOver to check for one of the three mark methods by checking if mGameOver is true since that is true for all mark methods.

**bool** Board::isGameOver( ) **const**

{

**bool** b = **false**; //initialize

**if** (mGameOver == **true**) //check if mark method went through

b = **true**;

**return**(b);

* 1. Previously, I used mBoard.isGameOver in BeatThat’s determineGameOutcome to first check if the game is over before looking for a winner or tie. However, this did not work because the Board’s isGameOver is updated because of determineGameOutcome because BeatThat’s isGameOver uses determineGameOutcome and changes Board’s mark methods. These mark methods would then change Board’s isGameOver. To fix this, I checked if the turn count reached 5 instead for game over.

BeatThat::GAMEOUTCOME BeatThat::determineGameOutcome( ) **const**

{

GAMEOUTCOME result = GAMEOUTCOME::GAMENOTOVER; //initalize with game not over

**if** (mTurnCount == 5) { //had 5 rounds, so reach game over

**if** (mHuman.getRoundsWon() > mComputer.getRoundsWon()) //human won more rounds

result = GAMEOUTCOME::HUMANWONGAME;

**else** **if** (mHuman.getRoundsWon() < mComputer.getRoundsWon()) //computer won more rounds

result = GAMEOUTCOME::COMPUTERWONGAME;

**else** **if** (mHuman.getRoundsWon() == mComputer.getRoundsWon()) //human and computer tied rounds

result = GAMEOUTCOME::TIEDGAME;

}

**return**( result );

}

1. List of test data

//Player

Player p; //person

Player c; //computer

assert( p.getRoundsWon( ) == 0 ); // nothing won yet...

p.wonARound( );

assert( p.getRoundsWon( ) == 1 ); // won 1!

p.roll( );

Die d1;

d1.setValue(1);

Die d6;

d6.setValue(6);

p.roll(d1,d6);

assert( p.largestDie( ).getValue( ) == 6 );

assert( p.smallestDie( ).getValue( ) == 1 );

p.roll();

string s = p.whatWasRolled(); //check for random

cout << s << endl;

p.roll(d1,d1); //roll 1 1 same number

assert( p.largestDie( ).getValue( ) == 1 );

assert( p.smallestDie( ).getValue( ) == 1 );

//Board

Board b;

assert( b.getHumanRoundsWon( ) == 0 );

assert( b.getComputerRoundsWon( ) == 0 );

assert( b.getTurnsLeft( ) == 0 );

assert( ! b.didHumanWin( ) );

assert( ! b.didComputerWin( ) );

assert( ! b.isGameOver() );

b.setHumanRoundsWon( 1 );

b.setComputerRoundsWon( 2 );

b.setTurnsLeft( 3 );

assert( b.getHumanRoundsWon( ) == 1 );

assert( b.getComputerRoundsWon( ) == 2 );

assert( b.getTurnsLeft( ) == 3 );

b.markComputerAsWinner( );

// b.markTied( );

// b.markHumanAsWinner( );

assert( ! b.didHumanWin( ) );

assert( b.didComputerWin( ) );

assert( b.isGameOver( ) );

//BeatThat

Die d2; d2.setValue( 2 );

Die d3; d3.setValue( 3 );

Die d4; d4.setValue( 4 );

Die d5; d5.setValue( 5 ); //Note: have die 1 and 6 defined above

//computer won

BeatThat game;

assert(game.turnsLeft()==5);

assert(game.isGameOver( ) == **false**);

game.humanPlay( d6, d2 );

game.computerPlay( d3, d4 );

game.endTurn( ); // human won this round...

game.humanPlay( d5, d1 );

game.computerPlay( d3, d4 );

game.endTurn( ); // human won this round...

game.humanPlay( d5, d2 );

game.computerPlay( d6, d4 );

game.endTurn( ); // computer won this round....

assert( game.isGameOver() == **false**); //game not over at this point

assert( game.determineGameOutcome( ) == BeatThat::GAMENOTOVER);

game.humanPlay( d1, d1 );

game.computerPlay( d3, d4 );

game.endTurn( ); // computer won this round...

game.humanPlay( d5, d4 );

game.computerPlay( d3, d6 );

game.endTurn( ); // computer won this round...

assert( game.isGameOver() == **true** );

assert( game.determineGameOutcome( ) != cs31::BeatThat::GAMENOTOVER ); //game over

assert( game.determineGameOutcome( ) == BeatThat::COMPUTERWONGAME ); //computer won

assert(game.getBoard().getHumanRoundsWon()==2);

assert(game.getBoard().getComputerRoundsWon()==3);

assert(game.getBoard().getTurnsLeft()==0); //no turns left

assert(game.getBoard().didComputerWin() == **true**);

//human won

BeatThat game2;

game2.humanPlay( d3, d4 );

game2.computerPlay( d6, d2 );

game2.endTurn( ); // computer won this round...

game2.humanPlay( d3, d4 );

game2.computerPlay( d5, d1 );

game2.endTurn( ); // computer won this round...

game2.humanPlay( d6, d4 );

game2.computerPlay( d5, d2 );

game2.endTurn( ); // human won this round....

game2.humanPlay( d3, d4 );

game2.computerPlay( d1, d1 );

game2.endTurn( ); // human won this round...

assert( game2.isGameOver() == **false**); //game not over at this point

assert( game2.determineGameOutcome( ) == BeatThat::GAMENOTOVER);

game2.humanPlay( d3, d6 );

game2.computerPlay( d5, d4 );

game2.endTurn( ); // human won this round...

assert( game2.isGameOver() == **true** );

assert( game2.determineGameOutcome( ) != cs31::BeatThat::GAMENOTOVER );

assert( game2.determineGameOutcome( ) == BeatThat::HUMANWONGAME ); //human won

assert(game2.getBoard().getHumanRoundsWon()==3);

assert(game2.getBoard().getComputerRoundsWon()==2);

assert(game2.getBoard().getTurnsLeft()==0);

assert(game2.getBoard().didHumanWin());

//tied game

BeatThat game3;

game3.humanPlay( d3, d4 );

game3.computerPlay( d6, d2 );

game3.endTurn( ); // computer won this round...

assert( game3.isGameOver() == **false**); //game not over at this point

assert( game3.determineGameOutcome( ) == BeatThat::GAMENOTOVER);

game3.humanPlay( d3, d3 );

game3.computerPlay( d3, d3 );

game3.endTurn( ); // tied this round...

game3.humanPlay( d6, d4 );

game3.computerPlay( d5, d2 );

game3.endTurn( ); // human won this round....

game3.humanPlay( d3, d4 );

game3.computerPlay( d3, d4 );

game3.endTurn( ); // tied this round...

game3.humanPlay( d3, d6 );

game3.computerPlay( d3, d6 );

game3.endTurn( ); // tied this round...

assert( game3.isGameOver() == **true** ); //game over

assert( game3.determineGameOutcome( ) != cs31::BeatThat::GAMENOTOVER );

assert( game3.determineGameOutcome( ) == BeatThat::TIEDGAME );

assert(game3.getBoard().getHumanRoundsWon()==1);

assert(game3.getBoard().getComputerRoundsWon()==1);

assert(game3.getBoard().getTurnsLeft()==0);