```
#ifndef _PLAYER_H_
#define _PLAYER_H_

#include <string>
using std::string;

// struct Position {
//      int row;
//      int col;

//      // // already implemented for you!
//      // bool operator==(const Position &other) {
//      //      return row == other.row && col == other.col;
//      // }
// };



// I think that the above position struct is not needed, as position is just 2 ints
and can be passed as such which i think is a lot more simplistic




class Player {
public:
        // TODO: implement
        // Player(const std::string name, const bool is_human);  // constructor

        // // These are already implemented for you
        // std::string get_name() const {return name_; }  // inline member function
        // int get_points() const {return points_; }  // inline member function
        // Position get_position() const {return pos_; }  // inline member function
        // bool is_human() const {return is_human_; }  // inline member function
        // bool hasTreasure() const {return has_Treasure_; }  // inline member funct
ion
        // bool isDead() const {return isDead_; }  // inline member function
        // int getLives() const {return lives_; }
        // these are just a bunch of getters for the player class

        // // TODO: implement the following functions
        // // You MUST implement the following functions
        // void ChangePoints(const int x);
        // // this will change the points of the player by the value of x
        // to me this seems like an odd way to do it since it has less control than
just setting the points to a value

        // // set a new position for this player
        // void SetPosition(Position pos);
        // this will set the position of the player to the given position
        // I might chagne it to like the others to just take 2 ints
        // that new function would be:
        // void SetPosition(int row, int col);

        // // checks if the player owns a treasure
        // void setHasTreasure();
        // this will set the has_Treasure_ to true
        // this function does not check if it owns the treasure, it just sets it to
true


        // to check if the player has a treasure i would use a getter
        // the getter is above



        // //checks if the enemy is dead
        // void setIsDead(bool isdead);
        // this will set the isDead_ to the given value
        // this function does not check if the enemy is dead, it just sets it to the
 given value
        // see above for the getter
```

```
        // //updates the lives for a player
        // void setLives();
        // this will update the lives of the player
        // it does not take a value, so i would change it to take a value
        // the new function would be:
        // void setLives(int lives);

        // // You may want to implement these functions as well
        // // ToRelativePosition is a function we used to translate positions
        // // into direction s relative to the player (up, down, etc)
        // std::string ToRelativePosition(Position other);
        // this will take a position and return a string of the relative position of
 the other position to the player
        // this will be based on the current position of the player and the other po
sition
        // this might not take into account walls



        // // Convert this player to a string representation of their name and point
s
        // std::string Stringify();
        // this will return a string representation of the player's name and points
        // this will be in the form of "name: points"
        // will also show the position of the player, and potentially the other fiel
ds


        // // this method should return the number of moves made by the player in th
is game
        // int getMoves();
        // this will return the number of moves made by the player in this game
        // i might not implement this because it will require storing what happened
in the game
        // i could store how many times the player moved, but that sounds less usefu
l than what happened
        // and i think that is not needed for this game


        // // update the number of moves in each human player's turn
        // void incrementMoves();
        // this goes along with the above function

        // You may add other functions as needed/wanted

private:
        string name_;
        int points_;
        // Position pos_;
        // instead of the position struct i will use 2 ints to represent the positio
n
        int row;
        int col;
        bool is_human_;
        bool has_Treasure_;
        bool isDead_;
        int lives_ = 3;
        int moves_;

        // You may add other fields as needed

}; // class Player

#endif  // _PLAYER_H_
```