Samuel Razumovskiy

CSCI 3010

HW 1: Part 1

Player:

Player(const std::string name, const bool is_human)	initialize a Player with the given name and whether or not it is human
void ChangePoints(const int x)	update the points_ value according to the int passed in
void SetPosition(Position pos)	update the Player's pos_ to the new position
std::string ToRelativePosition(Position other)	translate the other position into a direction relative to the Player by comparing other with pos_

Board:

int get_rows() const int get_cols() const	Get rows or columns
SquareType get_square_value(Position pos) const	Returns what kind of SquareType the position is on the Board
void SetSquareValue(Position pos, SquareType value)	set the value of a square to the given SquareType
std::vector<Position> GetMoves(Player *p)	get the possible Positions that a Player could move to, likely uses get_square_value
bool MovePlayer(Player *p, Position pos)	Move a player to a new position on the board. Return true if they moved successfully, false otherwise. Uses get_square_value, and ToRelativePosition()
SquareType GetExitOccupant()	Get the square type of the exit square using get_square_value()
friend std::ostream& operator<<(std::ostream& os, const Board &b)	Allows to print board positions, uses get_square_value()

Maze:

void NewGame(Player *human, const int enemies)	initialize a new game, given one human player and a number of enemies to generate, initializes using player class.

void TakeTurn(Player *p)	have the given Player take their turn, prints out the possible moves using GetMoves() and moves the player using MovePlayer.
Player * GetNextPlayer()	Get the next player in turn order
bool IsGameOver()	return true if the human made it to the exit or the enemies ate all the humans. Checks GetExitOccupant()
std::string GenerateReport()	Prints out the results of the game
friend std::ostream& operator<<(std::ostream& os, const Maze &m)	Print out Maze values