CSCI 3010 SP ‘23
Leif Anders
HW1 - Part 1 Deliverable

// Player --- --- --- --- --- --- --- ---

Method	Description
Player(const std::string name, const bool is_human)	Construct a new Player:: Player object with the values passed for its members
void ChangePoints(const int x)	Changes the value of player's points to the value passes
void SetPosition(Position pos)	Sets the new position of the player object to the Position passed
void setHasTreasure()	Sets player object's setHasTrasure_ to true
void setIsDead(bool isdead)	Checks to see if an ENEMY player is dead
void setLives()	Update/ set the player objects lives (back to 3)
std::string ToRelativePosition(Position other)	Translates nearby valid positions of player object to directions (Up, down, left, right)
std::string Stringify()	Uses the overloaded '<<' operator to print player objects name and points
int getMoves()	Gets the number of moves made by this player object
int incrementMoves()	increment this player objects moves_ by 1

// SquareType --- --- --- --- --- --- --- ---

Methods	Description
std::string SquareTypeStringify(SquareType sq)	Takes an enumerated SquareType class and returns it as a string

// Board --- --- --- --- --- --- --- ---

Methods	Description
Board()	Construct a new Board:: Board object
void SetSquareValue(Position pos, SquareType value)	Set the SquareType value at a Position with the given values
std::vector<Position> GetMoves(Player *p)	Return a vector of the possible moves that a player object can make
bool MovePlayer(Player *p, Position pos, std::vector<Player*> enemylist)	Move a player across the board. Return with the status of the movement's success.
bool MoveEnemy(Player *p, Position pos)	Move an enemy across the board. Return with the status of the movement's success.

// Game --- --- --- --- --- --- --- —

Methods	Description
Game()	Construct a new Game::Game object
void NewGame(Player *human, std::vector<Player*> enemylist, const int enemies)	Initialize a new game with a human player and generate a number of enemies to oppose
void TakeTurn(Player *p, std::vector<Player*> enemylist)	Sequence for human player’s turn
void TakeTurnEnemy(Player *p)	Sequence for computer player’s turn
bool IsGameOver(Player *p)	Checks human players isDead_ status to see if game is over
bool CheckifdotsOver()	Checks if all pellets are gone from board
std::string GenerateReport(Player *p)	Generates a report of game statistics