Jaryd Meek
CSCI 3010 – HW4 Pt. 1

Class Outline:

mainwindow

Public:	
MainWindow(QWidget *parent = nullptr);	Constructor for main window, creates object and sets it up.
~MainWindow();	Deconstructor For Main Window
LoadLeaderboard(string filename);	Loads leaderboard from csv
SaveLeaderboard(string filename, vector<*Players>);	Updates leaderboard with information from new players
Private Slots:	
void NewGame();	Default slot that reacts to New Game button being clicked
void StartGame();	Default slot that reacts to Start Game button being clicked
void EndGame();	Default slot that reacts to End Game button being clicked
void QuitGame();	Default slot that reacts to Quit Game button being clicked
void UndoMove();	Default slot that reacts to Undo button being clicked. Emits signal to Player::Undo();
void ReRoll();	Default slot that reacts to Re-Roll button being clicked
void Roll();	Default slot that reacts to Roll button being clicked
void Move();	Default slot that reacts to Move button being clicked. Emits signal to Player::MovePlayer();
Private:	
vector<*Player> still_playing_players_;	Storage for players who have not quit
vector<*Player> all_players_;	Storage for all players who started (only used for leaderboard)
vector<*Tile> tiles_;	Storage for Tiles

int current_turn_;	Current Player's Turn

Player

Public:	
Player(QColor color, string name)	Constructor for player, creates object and sets it up.
Signals:	
void MovePlayer(int roll);	moves player's current location. Receives signal from Move() slot.
int Undo();	returns the location of the player previously. Receives signal from UndoMove() slot.
Private:	
string name_;	storage for name
QColor color_;	Storage for color
vector<*int> previous_moves_;	storage for previous moves
int current_location_;	Player's current location
int undos_remaining;	tracks number of undos remaining
int rerolls_remaining;	tracks number of rerolls remaining

Tile

Public:	
Tile(int number);	Constructor for tile, creates object and sets it up.
AddPlayer(Player* add_player);	adds the player to the tile

RemovePlayer(Player* remove_player);	removes the player from the tile
Private:	
int number;	storage for number of tile
int * chute;	storage for chute (if applicable for this tile)
int * ladder;	storage for ladder (if applicable for this tile)
vector<*Player>	storage for players currently on this tile