5: Column Logic

CSCI 6626 / 4526 February 2018

1 Goals

- To build functions with non-trivial behaviors.
- To model the rules for moving a player in a column.
- To build a test plan that will test all parts of your class.

2 Column Logic

Phase 4 of this program ended with a Game shell that contains dice, players, and columns. None of these are in their final configuration, but they are enough for you to begin building the game logic.

3 Instructions

In the Column class, add these parts:

1. A public function *bool startTower(Player*)* to place a tower in the column, in the appropriate square for the given player. The function should return `true` if it is legal for the current player to place a tower in this column at this time, `false` otherwise.

2. A public function *bool move( )* to advance the tower one square in the column. If the move would capture the column, set the state to pending. Return `false` if a move was illegal at this time, `true` if you made a move.

3. A public function *void stop( )* to be called when a player chooses to end his turn. It must call Player::Color() to access the color of the tiles to use, then replace the tower by the appropriate colored tile. If If the column state is `pending`, set the state to `captured` and call Player::wonColumn() if the player has captured the column. Return the column State.

4. A public function *void bust( )* to remove the tower from the column at the end of a failed turn. Reset the state to available.

4 Testing

In your Game class, you already have two columns. Using those columns, test all possible outcomes from this week's functions. Make sure the state is set to pending or captured when it should be. Make sure the output from print is neat and makes sense.