**class GameWorld – contains the actual objects of the game**

levelArray – 3D array describing the layout of the immovable blocks on the level (the “ground”)

levelItems – array of levelItem objects containing the interactive items on the level

robots – array containing the Robot objects in the level

LoadLevel(int LevelID)  
accesses the specified level using a web request and populates the levelArray, levelItems, and robots accordingly.  
  
LoadSolution(int LevelID, int UserID, bool isCorrect)  
accesses the most recent solution to the level (most recent CORRECT solution if the flag is true) and populates the robots’ program accordingly

CheckCompletion( )

Polls the levelItems list to see if each item has its completion conditions satisfied. If so, returns true, if not, returns false.

StepForward( )

If all robots are done, CheckCompletion() and display an appropriate success/fail message. Otherwise,  
Polls the robots, directing each to execute one instruction.  
Polls the items, directing each to move one ‘tick’ if possible

Run( )

Simply calls “StepForward” until done.

**Class Robot – robot character which is directed to move around the GameWorld. May be multiple types of robots in the future which handle different types of CodeElements.**

programArray – array containing the ordered CodeElement objects which make up the program. (this also contains the functions which are defined for this robot)

nextInstruction – int index of the location of the next instruction in the programArray

returnLocation – array of indices to return to – essentially, a Call Stack

ExecuteInstruction( ) – robot executes the next instruction, following the directions, and advances nextInstruction to the proper location. This function should be able to modify the robot’s position on the GameWorld, as well as the positions of other objects on the GameWorld.

HasNextInstruction( ) – returns Boolean value indicating whether this Robot has any more instructions

**Class ProgrammingGUI – allows the player to modify the program of a selected robot. Toolbox only draws CodeElementOrigin for CodeElements that that robot understands**  
toolbox – array containing CodeElementOrigin objects, each of which has an attached CodeElement, clones that element when clicked, and attaches it to the mouse

PickUp()

Called on mouseclick over the program area. Taking into account the current mouse position, either

-attaches the CodeElement in the Robot’s programArray to the mouse, or

-attaches the CodeElement in the parameterArray of the CodeElement to the mouse

Drop()

Called when mouse released over program area. Taking into account the current mouse position and held CodeElement, either

-inserts the CodeElement on the mouse into the Robot’s programArray at the appropriate index, or

-inserts the CodeElement into the parameterArray of the CodeElement at that index

**Class CodeElement – individual “chips” that make up the program**

**(Behavior for each code element is described within Robot)**

type – enum describing where the CodeElement can be placed (for example, a variable can be placed in the parameters of a for loop, but not directly into the program.)

parameterArray - (for functions and loops) array of CodeElement objects which modify the behavior of the CodeElement

scope – pair of integers describing the beginning and ending indices of a structure (a loop’s brackets, or a function’s start and end)

Execute() – creates a “Command” object and sends it to the robot whose program the CodeElement is in