Language Overview

- Assists casual users to create their own game levels.
- Focuses on the game platform similar to the Helicopter game.
- Support several useful implementations: function calls, array variables, control flow, recursion
How to use?

■ Syntax is similar to Java and C
  ■ If, else, for, while loops are the same
  ■ Array access is similar
  ■ Uses primitives such as int and string

■ However, it is also very different!
Data Types

- **Brick**
  - RGB values, Pointer to an Array, and X and Y coordinates
  - RGB are three ints
  - The Array contains a list of points that creates the shape
  - X and Y coordinates marks where the Brick is on the display

- **Player**
  - RGB values, Pointer to an Array, and Y coordinate
  - Player only moves vertically (similar to that in the helicopter game)

- **Map**
  - Height, Width, and pointer to a function
  - Size of the map
  - The function generates and returns an array of bricks
Identifiers

- We decided to start all identifiers with “$”. This makes spotting a variable name or function name very simple. And avoids confusion for both the user and the parser.

- The built-in functions also need “$” before the function name in order to remain consistent.
  - E.g. $Run, $printint, $printstring, $GenerateRandomInt, etc.
Demo