**Documentation:**

**How to Compile Program:**

1. Set current directory to the Question3/src folder using Command Prompt or Terminal
2. Type javac Main.java to compile the program
3. Type java Main.java to run program

**How to Use:**

**RANDOM:** Sets Map with a random positioning of potholes and one Start Tile

**RESET:** Sets all tiles in the Map to a Coordinate Tile

**SOLVE:** Finds all valid paths from current Map

**<:** Display previous Valid Path for Map

**>:** Display next Valid Path for Map

**Design**

The JavaFX application displays a Map of Chicken City and a simple interface to interact

with the Map and Valid Paths. When clicking the Tiles on the Map, the Tile will either

change to a Coordinate, Pothole, or Start Tile. The Start Tiles can only be created on the

left-most column of the Map where Henny Penny will cross. The Map can also be set

with random positioning of potholes and one Start Tile. Once the Valid Paths are found,

the Valid Paths can be displayed through iterating through a Valid Path list.

**Implementation**

**Model:**

The Model for the application contains a Map, Tile, TileType, and a ValidPathVisitor. The Map is a composition of Tiles. A Tile contains a row position, column position, and a TileType. A TileType consists of subtypes which are Coordinate, PotHole, ValidPath, and StartTile to determine if Henny Penny can cross a Tile. The ValidPathVisitor first visits the starting tile and then the visitor executes a depth first search until it is able to get to the last column position. If it encounters a Pothole, the visitor will stop at that branch and go back to the previous Tile. When the ValidPathVisitor completes, the ValidPathVisitor will contain a list of all Valid Paths.

**Controller:**

The MainController sends commands to the Model based on user input. The

MainController also attaches a TileView to a Tile. A TileView details how to display a Tile in the View depending on the TileType.

**View:**

The MapView and TileView display the Map and Tiles in the Model. The TileView is an Observer of a specific Tile. When a Tile’s TileType changes, the TileView will be notified which will change its Color. Once the Map is solved for all Valid Paths. The View can display all Valid Paths.

**Extension**

**Does your solution work with larger grids?**

The solution does work with any size grid. The issue occurs with time complexity. The

time complexity of finding all Valid Paths for an instance of a map is exponential. The

exact complexity is O((NUM\_OF\_ROWS)^(NUM\_OF\_COLUMNS-1)). The worst case

being there are no potholes for an instance of a Map.