# Assignment 4: Alice in Wonderland Game Design notes.

Jackson Wiebe

3519635

# Main game architecture:

Alice in Wonderland, the Text based adventure is written in Java 11. The code can be compiled with the standard command *JAVAC \*.java* and run the main loop with the command *JAVA Wonderland*. Instructions for the user are held in the Help.txt file.

Wonderland is the main static class involved by the user. This creates a Game object which contains the main loop of our game. The game class then prints out the relevant user information, Welcome file and Help file. Some ASCII art was included to increase the aesthetics of the game. Then this class loads the game resources by creating an instance of each room child class, which in turn loads all the relevant Characters, Items, Exits, and conversation nodes. The game loops through the target location executing the Enter() command until a new location is determined and ultimately the END location is given to end the game or the user quits or fails.

Each room is built as a child class of a parent Location class. This Location class contains most of our code and the main loop while the user is contained in the room. The enter() function is the most critical is this loop will gather user commands and execute them accordingly. Subsequent Speak, Take, Help etc, are executed from there.

# Item architecture:

Items are built of a parent Item class. Since so many items contain unique functions or variables each unique item is given a child class which overrides some portion of the parent class. As the objects are created at the beginning of the game, the constructor grabs descriptive text from a related TXT file of the same name. Each item is given a unique ID through the use of a ENUM as defined in ItemList.java. Items can be created and destroyed. Some are given as disposable, one time use, and some are reusable through the game.

# Room architecture:

Rooms are built of a parent Item class. Each room will have a unique set of items, characters, exits, and such are added at the child class level through the use of functions in the super class. Many rooms have unique functions that will also override the parent class. Each room is given a unique ID through the use of a ENUM as defined in LocationList.java.

The parent class Location.java contains the majority of our loop and decision making code. New user input is gathered here and decisions are made to decide the course of action for the game as per the user choice. Speak, Take, Quit, Inventory are all handled at the base level here. Further code with proceed from these base points.

Garden class also its own instance of inventory to act as a safe point for Alice to store items of importance. This class contains several overrides to handle the safe, Get, Take, and Store methods as requested to handle those cases. Items stored in this location persist through the duration of the game.

# Character architecture:

Non-playable characters are built from the Character.java class. At creation each character will load resources from a txt as defined by their name through the constructor. Character class also contains Conversation nodes that relate to the target conversation. Future expansion was considered to chain conversation nodes. Each character is given a unique ID through the use of a ENUM as defined in CharacterList.java.

Alice, as the main character, was given her own class. As she contains special variables and commands it made more sense to keep this class separate. Alice also contains an instance of inventory class to handle the items she gathers along the journey.

# Character List:

* Alice – The main character, Contains an instance of inventory and has a unique property of changing size
* Mad hatter - A real Riddler with a puzzle to solve
* The White Rabbit- only seen in passing
* Queen of Hearts – A very rude women who tries to take your head!
* Cheshire Cat – A curious cat who wants to know about the mad hatter.
* The Caterpillar – Has an unsolvable riddle.
* The Dormouse – A small mouse with an easy test question to get the user familiar with the conversation system.

# Item List:

* Golden Key – used to open the door from the hall of many doors. Only works in one room.
* Leaf – Demo object found in first room. Has no effect.
* Cake – Makes Alice large
* Bottle – makes Alice smaller
* Mushroom – starts an instance of the mushroom minigame
* Mallet – Used to play Croquet of course
* Ball – a ball to play Croquet, only works with the mallet.
* Gloves - a pair of white kid gloves, has no game changing effects.
* Fan - A Large handheld fan, has no game changing effects.
* Tea – A scalding hot drink which ends the dream for Alice. Game over.
* Watch – The rabbits pocket watch, Has some unique properties of telling time.
* Slippers – Ruby slippers. Wait? Who’s are these? , has no game changing effects.

# Room List:

* Riverbank: The first scene for alice where she is given the chance to wake from her dream early. This room contains a one way exit to the subsequent Hall of tears.
* Hall of tears: Based on the chapter of the same name this location contains 2 exits leading the user to the Mad Hatter in the Tea Chapter or the Rabbits garden.
* Garden: The safe is located here for users to store items. Only the cat is given for conversation with a un solvable puzzle. This room contains 2 exits, back to the Hall of tears or through the mushroom forest.
* Mushrooms: In the forest of mushrooms the user can talk with the caterpillar and engage in his mini game. 2 exits are given; back to the Garden or forward to the Tea party with the Mad Hatter.
* Tea in the garden with the Mad Hatter. Alice meets the famous mad hatter and has a hard riddle to solve. 3 exits are given here. Back to the Hall of Tears, Back to the Mushroom Forrest. Or forward to the Courtyard where she can play croquet with the queen.
* Courtyard Croquet. Here the user has a few items to interact with and sets a one way path to the finish. 2 exits are given; back to the Tea party or forward to the courtroom.
* Courtroom. A final room where Alice is given a hard puzzle again. Success here wins the game.

# Room Layout: