Readme File for CS 342 Project 5 by Conway Dang, Michelle Nguyen, Jack O’Donnell
The program can successfully take in a GDF 3.1 file, GDF 4.0 file, GDF 5.0 file, and a new file type GDF 5.1. This program sets up the places, directions, artifacts, and characters using any of the above files.

<u>Differences in Files</u>
3.1 - No Pre-Created Characters
4.0 - Has Pre-Created Characters
5.0 - Shifted Starting Location to a new line
5.1 - Includes Health for characters and damage for certain artifacts

The user can type in a set of commands that will help him navigate the map, pick up artifacts, use artifacts, exit the game, drop the artifacts, search through the room, and print information. The GDF file must be in the same directory as the code AND the file must be labeled “MystiCity 31.gdf” or “MystiCity 40.gdf” or “MystiCity 50.gdf” or “MystiCity 51.gdf”.

There is a UI for every character that is a “Player” and a AI interface for every NPC. NPCs are only allowed to move and pass their turn. They are limited to this because we did not want the NPCs to sabotage the player’s progress as he/she navigates through the map by taking an artifact that the user needs. Additional functionality can be implemented later.

<u>Compiling Program</u>
The user must type make and then to run the program the user must type java GameTester. To clean, the user must type make clean.

<u>Inputs Allowed</u>:
- 16 Different Cardinal Directions (N,S,E,W, NW, SW, etc.) allows the user to move in that direction.
  - Case insensitive and typing the name will do the same thing as well.
- **Quit** or **Exit** for Ending the program (Case insensitive)
- **Print** for printing information about the ENTIRE game. Places, available directions, what characters are in the room, their inventory, etc. (Case insensitive)
- **Look** for telling the user what room he is in and what other characters are in the room as well.
- **Get [Artifact name]** for grabbing the artifact assuming it is in the room and the user can carry it.
- **Drop [Artifact name]** for dropping the artifact in the current room the user is in
- **Use [Artifact name]** for using the artifact in the room the user is in.
  - If the item is a key it will unlock any valid door. Otherwise, it does nothing :(
- **Go**/**Head**/**Move** [Cardinal Direction] allows the user to move in the direction the user specifies.
- **Inventory** or **Inve** for checking what artifacts are in the user’s bag and how much the capacity is filled.

- **Pass** to pass your turn to next person/AI.
- **Help** prints out some useful UI information that the user can read to help navigate through the game.
- **Fight**/**Attack**/**Challenge** causes the user to do some damage to another character. If the character's health goes at or below 0 then the character "dies" and does not make any turns anymore.
- **GUI #** switches interfaces, where # is replaced with an integer starting at 0, and "GUI 0" switches to the text interface.

<u>Individual Contributions</u>
Conway - Players able to fight each other with the "Fight" command. Attacks will occur 100% of the time, but they can be easily modified for Random generated purposes. Wrote the Readme as well.
Michelle - Implemented the damage characteristic to Artifact. Modified the GDF to include health and damage. Updated the UML.
Jack - Implemented health to characters. Implemented characters dying. Modified the parsing of the GDF to include health and damage.

<u>GUI Implementation</u>
<u>Side Note:</u> Make sure you type in no for the first two prompts before trying to switch between GUIs'.
<u>Second Side Note:</u> You may have to resize the window to see all the text in all of our GUI's.
Conway - My GUI can display output and has a submit button to send in what the user types in back into the code, but it is not implemented well because I was unsure how the GUI will be implemented into the code itself. However, most of the code's print statements and gaining input was replaced with appropriate display and getline methods.
Michelle - I designed my GUI to display the output using JTextArea. I attempted to use JTextField to submit the user input, but that functionality lacks. Also, I designed the UML diagram.
Jack - Basically designed my GUI with JFrame. Got all the text to show whenever anything is displayed. Then I have a button and a text field so the user is supposed to basically type in their command and when they are done click the button to send it to getLine(). Which it does but even though getLine() properly sends the command, the command doesn't get executed and I don't know why.