1. Design Documentation:
   1. A complete UML document of the game package is located in the doc subdirectory of the packaged source code.
   2. The title of the UML document is: EricSabelhausSorcerersCaveUML.png
2. User's Guide: description of how to set up and run your application
   1. Setup of this application requires at least Java version 1.7
   2. In order to run the application, the following command must be run within the bin directory of the compiled application code:
      1. java -classpath bin game.GameGUI
   3. Once the application is running, select one of the three available caves to explore. Listed below:
      1. SmallSimpleCave.txt
      2. A\_BigData.txt
      3. LargeCave.txt
   4. After you’ve selected your cave, you will be presented the sorcerers cave GUI
   5. Going from top to bottom, the first element you see is the main set of navigational tabs, which is followed by the content presented on the selected tab. The following are the tabs functionally from left to right:
      1. Search Tab
         1. Top bar
            1. To the left is the submit button
            2. To the right of the submit button is the search selector, where you may choose to search by Name, Type, or by Index.
            3. To the right of the tab is the input field
         2. Search results
            1. When a search is executed, the content of the item searched for will be displayed in this text field. If an illegal search parameter is entered an appropriate popup will notify of the change necessary to perform the search.
      2. Party Tab 1…n
         1. To the left of the party tab is a vertical tab selection of a sortable text view, followed by the individual creature tabs of that party sorted alphabetically
         2. Sortable Party View Tab:
            1. In this tab, you will be presented two dropdowns in a top bar, as well as a submit button.
            2. The first dropdown allows sorting on Creatures or Treasures
            3. The second dropdown allows sorting on the selected type in the first dropdown by the available search parameters on that type
            4. How to Sort:

Creature – Sorts entire party

Age – sorted numerically

Empathy – sorted numerically

Fear – sorted numerically

Height – sorted numerically

Name – sorted alphabetically

Weight – sorted numerically

Treasure – Sorts on each creature

Value – sorted numerically

Weight – sorted numerically

Once a set of selections has been picked, pressing the Sort button will reorder the elements of the text accordingly

1. Test Plan: sample input and ***expected*** results, and including test data and results, with screen snapshots of some of your test cases
   1. Sample Input
      1. The following is the sample input provided during testing:

// p:<index>:<name>

p : 10001 : Unity

// c:<index>:<type>:<name>:<party>:<empathy>:<fear>:<carrying capacity>

c : 20001 : Woman : Lucy :10001 : 17 : 22 : 20

c : 20002 : Woman : Jane :10001 : 22 : 15 : 25

c : 20003 : Worg : Brandon : 0 : 30 : 21 : 0

// t:<index>:<type>:<creature>:<weight>:<value>

t : 30001 : Gold : 20001 : 50 : 2000

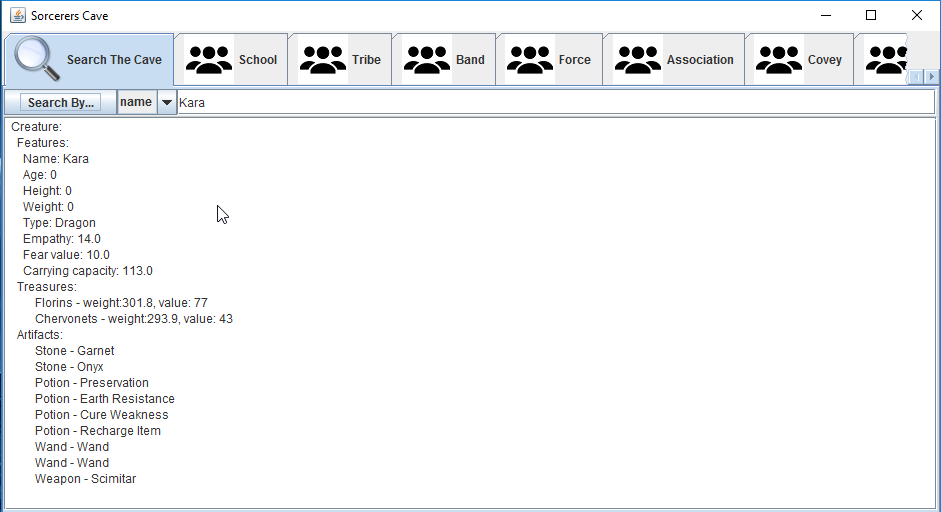
t : 30002 : Gold : 0 : 75 : 5000

t : 30003 : Gems : 20002 : 10 : 10000

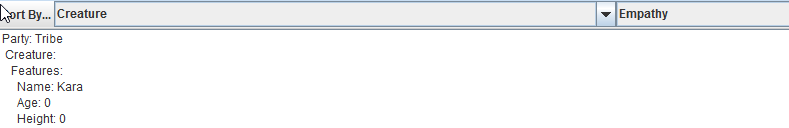
// a:<index>:<type>:<creature>[:<name>]

a : 40001 : Wand : 20001 : ElderWand

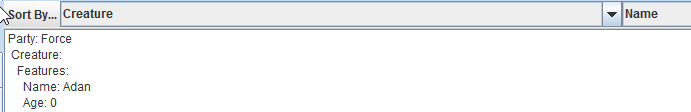
* + 1. The expectations are as follows:
       1. One Cave is created
       2. One Party is created
          1. Must belong to the one cave
       3. Two Creatures are created
          1. Both must belong to the one party
       4. Three Treasures are created
          1. One owned be each creature
          2. One must be undiscovered
       5. One Artifact is created
          1. Must belong to the first Creature
       6. Before sorting Lucy should come first
       7. After sorting Jane should come first
  1. Result
     1. Assertion tests using Java JUnit testing framework succeeded in 0.806 seconds
  2. GUI Test Cases:
     1. Picking LargeCave.txt using JFilePicker. File contains no undiscovered items. After file loaded, all parties are present in tabs with no undiscovered items or creatures present as expected
     2. Search by name for character named Kara returns expected creature.



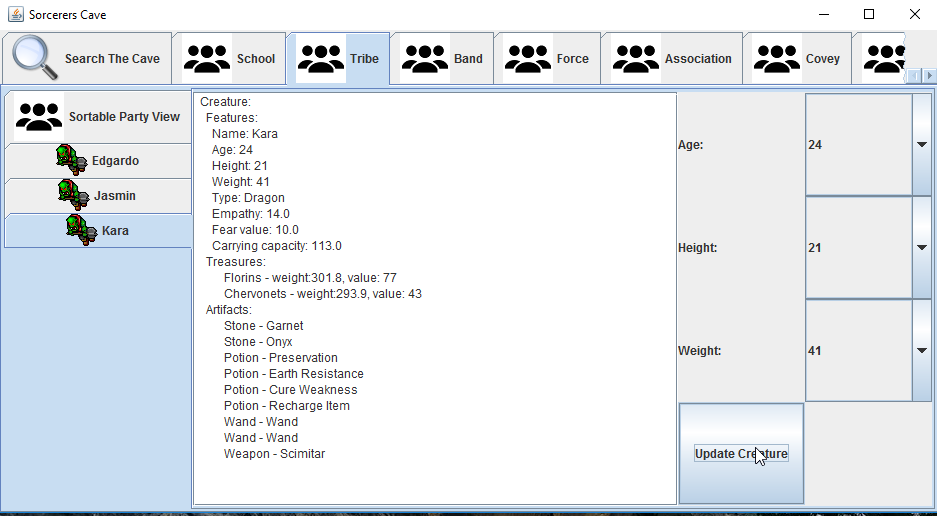
* + 1. Search by Name for foobarbang returns expected standard output. Pressing
    2. Search by type of Stone returns all Stones as expected.
    3. Search by Type for foobarbang returns expected standard response.
    4. Search by index of 20002 returns Creature Kara as expected.
    5. Search of non-existent index returns standard response.
    6. Search of index (below lower bound) returns standardized response.
    7. Search of index (above upper bound) returns standardized response.
    8. Search with empty string returns correct standardized response.
    9. Sorting on creature by empathy within the Tribe group places Kara



* + 1. Sorting on creature by name within the Force party places Adan at top



* + 1. Selecting any character tab within any party reveals the character update form, and all character attributes
    2. Update attributes of a character occurs when button is pressed and numbers are selected



1. Comments:
   1. I think I may need to refactor a little more, there is a tighter coupling than I would prefer
   2. I don’t like that I have iterative search functionality for every item, though, it is very effective for the purposes of this application since it’s not an insane amount of iteration.