An Interactive Parts Inventory

CSCI 2270 Project

Carter Mak and William Walker

Notes on compiling and running:

- The script `compileAll.sh` will produce a `project.jar` file to be run (the file is already compiled and included here).
- The program is built for Unix systems, specifically Linux. It may work on Apple Mac systems, however is untested.
- Running the program will require the Java JRE and JDK be installed. We used Oracle Java 11 for this project.
- To run, use the bash command:

```
java -jar project.jar
```

Data Structures Used

Hash Table:

We use a table indexed by hashed part numbers to store the parts in inventory/on machines. Collisions are managed with chaining, using vectors in lieu of a manually implemented linked list because vectors are more conducive to serialization for saving.

Binary Search Tree:

We use a binary search tree to store part name words and facilitate part search by name. The associated search algorithm returns all parts which contain all of the words input by the user (case-insensitive).

Future Developments

Because this project is designed to be implemented for use after its completion with regards to the requirements of the CSCI 2270 project assignment, there are various developments which we already have planned/are implementing as improvements over this version.

Saving

We have already begun to implement backend code to periodically serialize and save data, then recall that saved data on startup (i.e. *autosave*). This also opens to the possibility of using an open source linux utility such as RClone to facilitate periodic cloud backup. The latter functionality could be paired with separate file access and writing to facilitate basic remote access, e.g. through Google Drive.

Some code has been established to facilitate saving. The code is declared in `project.hpp` and defined in `saveMethod.cpp`, however it is not currently being compiled into the final program.

User Interface

The Java GUI we've implemented is a functional demonstration of our data structures and of our general ability to create a graphical interface, however the final implementation of this program will likely involve a rearrangement of our UI to better cater to the needs of end users. Such a redesign might follow a plan akin to this: