Spike10 Report on data structures

## Structures

The 4 structures I picked were:

* List
* Vector
* Map
* Multimap

## List

List is a dynamic container that supports constant time insertion and removal of elements. It is simple to use and can be inserted to from both the front and back. Its biggest downside is the lack of fast random access (using [1] or at()) meaning it is hard to get to a specific element without searching the entire list manually.

## Vector

Vector is similar to list but has the fast random access. However it has the issue of having no key value pairing (which list has as well) that means say you wanted to have a class as the vector and you needed to find one by its internal class id or type (public) you would need to access it internally. In the case of Zorkish, you would need to search inside of each item for the type, say rock, instead of search the inventory. This means that the search functions would need more complexity.

## Map

Is a sorted associative container that contains key-value pairs with unique keys. This enables you to search quickly for items. However the unique keys is an issue, as say the key was a string, you would need to have rock1, rock2 and rock3 to differentiate them. It also has fast random access.

## Multimap

Multimap is map but allowing non-unique keys. However it does not have fast random access either, meaning lots of uses of iterators in order to access items.

## Choice

Its between vector and map, because of the lack of fast random access.

I am going with Vector as the main difference between map and vector is the keys, and without non-unique keys, there is effectively no difference between the two, meaning that I can use the less complex one in terms of setup, with not having to worry about the keys.