Further Property Testing

Hugh Pedley

CS 1632 – FINAL DELIVERABLE

Property Testing Sorted ArrayLists

For this final deliverable, I decided to do more property testing. In a similar vein to a previous deliverable, I did property testing on sorted ArrayLists. I chose to do this kind of property testing because I wanted to further reinforce the idea that ArrayLists have been properly tested in and out, and that they are an extremely viable means of storing objects. In fact, I had to sort an ArrayList in another project earlier today where I had to write my own Comparable function, and was inspired to do this.

Five property tests were completed on one hundred different ArrayLists of varying sizes. These tests checked that every ArrayList remained an ArrayList object after being sorted, every ArrayList retained its original size, duplicate ArrayLists are equivalent after being searched, sorting an ArrayList is idempotent, and that sorting an ArrayList does in fact return a sorted ArrayList. In the future, I think I would continue to do property testing on sorted ArrayLists where they are parameterized differently. Testing on sorted objects with String parameters, generic parameters, or even object parameters may yield different results on these tests.

Fortunately for us all, these written tests all passed. This means that the ArrayList object and specifically the sort functionality used on it should continue to be used. It operates exactly as intended and expected as of this moment (at least, for lists of integers). No tests written failed. And as of this moment, I cannot come up with another property to test on a sorted ArrayList. Because of this, I wholeheartedly recommend using the provided collections.sort() on any ArrayList<Integer>.

GitHub Repository: <https://github.com/hughpedley/cs1632_deliverable6>

