LAB EXERCISE

Store

Background:

This lab exercise will build a preliminary program to be used in Lesson 28 to study search algorithms. We will use these classes to load and manipulate data stored in a text file, *file50.txt*.

```
public class Item implements Comparable
{
  private int myId;
  private int myInv;

  public Item(int id, int inv)
  {
    myId = id;
    myInv = inv;
  }

  public int getId(){ )
  public int getInv(){ }
  public int compareTo(Object otherObject){ }
  public boolean equals(Object otherObject){ }
  public String toString(){ }
}

public class Store
{
  private Item[] myStore;

  public Store(String fileName) { }

  public void displayStore()
  public String toString(){ }
  public void doSort() { }
  private void quickSort(Item[] list, int first, int last)
  private void loadFile(String inFileName)
}
```

The first line of *file50.txt* contains the number of id/inventory integer pairs listed on subsequent lines. The idea behind the data type `item` is the simulation of an item in a store, nearly all of which use a bar code for identification purposes. For every item in a store we keep track of an id value and an inventory amount. So *file50.txt* looks like this:

```
50
3679       87
196        60
17914      12
18618      64
2370       65
...        ...
etc. (for 45 more lines)
```

Each `id` value in *file50.txt* will be unique.

Assignment:

1. Write a program that solves the following sequential events:

   – loads the data file, *file50.txt*, as a collection of `Item` types maintained in a `Store` object
   – sorts the data using *quickSort*
   – prints the data , now in increasing order based on the `id` field of an `Item` object

2. The printing should add a blank line after every 10 items in the output. Include a line number in the output. For example:

```
           Id       Inv

 1       184        14
 2       196        60
 3       206        31
 4       584        85
 5       768        85
 6      2370        65
 7      3433         5
 8      3679        87
 9      4329        64
10      5529        11

11      6265        58
12      6835        94
13      6992        76
14      7282        73
15      8303        90
16      9267        68

   etc.
```

3. You will need to complete the `getId()`, `getItem()`, `compareTo()`, `equals()` and `toString()` methods for the `Item` class.

4. You will need to complete the `Store()` constructor, `displayStore()`, `loadFile()`, `quickSort()` and `toString()` methods for the `Store` class.

5. Your program should use appropriate amounts of decomposition, as modeled in the *ages.java* program.

6. Display your source code and a printout of sorted *file50.txt* as described under #2. Call your instructor to your workspace for scoring.