Code Documentation Requirements
CSc 345 – Summer 2010
Kevin Coogan

1 Project Documentation

The file that contains your main method will begin with a block comment that describes the problem to be solved, and gives a brief description of your solution. See Figure 1.

```
/*
 * Program 1 -- sorting
 * Kevin Coogan
 * created: June 16, 2010
 *
 * Problem Description -- implement and compare all major sorting routines
 * based on runtime, number of element comparisons, and total number of
 * operations.
 *
 * Solution -- bubbleSort, insertSort, quickSort, mergeSort, and crazySort
 * implemented in sorting.java.
 *
 * sortDriver.java (this file) calls each sort method on the same data set,
 * recording total execution time, number of element comparisons, and total
 * number of operations for each.  A table is printed showing all results,
 * and the best algorithm for each category is identified.
 *
 */
```

Figure 1: Sample Project Block Comment

2 File Documentation

Each file will contain a block comment as the first text in the file (except the file that contains main, see above) that gives the name of the file, the authors name, the date the file was created, and the purpose of the code contained in the file. See Figure 2.

```
/*
 * sort.java
 * Kevin Coogan
 * created: June 16, 2010
 *
 * sort.java implements several different sorting routines that typically
 * takes an arrayList of values, and returns a reference to a new arrayList
 * of sorted values (exceptions where noted).
 *
 */
```

Figure 2: Sample File Block Comment

3 Method Documentation

Each method will contain a block comment above the method that describes the algorithm used (briefly), gives the in, out, and in/out parameters, and describes the return value of the method. See Figure 3.

```
/*
 * crazySort -- uses standard quickSort with the following modifications:
 *      selection of pivot is random from last three elements of the
 *          arrayList.
 *      when arrayList size is less than 35 elements, switches to standard
 *          insertionSort to improve speed.
 *  Parameters:
 *      in:     arrayList of elements
 *      out:    none
 *      in/out: none
 *
 *  Return Value:
 *      reference to new arrayList of sorted elements
 */
```

Figure 3: Sample Method Block Comment

4 Other Documentation

Documentation should be added within the code whenever the technique used may not be obvious, or when the reader may be confused. Typically, we use line comments (not block comments) within methods. See Figure 4.

```
//find value in the array
while((index >= 0) && !found) {
    if(array[index] == val) {
        found = true;
    } else {
        index--;
    }
}

//if found, index holds array index of val, if not, it holds -1 which
//will serve as our error code to return to calling method
return index;
```

Figure 4: Sample Other Comment