High Scores

Instructions: Unless it has been done for you, print a copy of this document for your team. As you work through it, write answers to any questions or prompts that are in **boldface**.

Team:

Manager:

Recorder:

Presenter:

Analyst:

Analyst: Pay special attention to your team's performance in the area of *information processing*. How does your team read and interpret text, diagrams, code, etc.?

Learning Objectives

Content Objectives

Parentheses below correspond to part of the knowledge units in the ACM's *Computer Science Curricula 2013*.

After completing this activity, students should be able to:

- Explain and modify the insertion sort algorithm (AL/Fundamental data structures and algorithms).
- Explain and modify the mergesort algorithm (AL/Fundamental data structures and algorithms).

Process Objectives

After completing this activity, students should have improved their ability to:

- Read the code of nontrivial algorithms. [Information processing]
- Modify code that solves one problem to solve a slightly different problem. [Problem solving]

Model 1: Playing the Game

Open the High Scores project in Eclipse.

This project isn't a game *per se*, but could be a part of many game programs.

Run HighScoreList.java and follow the instructions. Note that this program runs in the console; it does not have a graphic user interface.

1. **How many scores does the program let you enter?**

2. **In what order are the scores sorted?**

Model 2: Basic Data Structure

Examine HighScoreList.java and HighScoreListTest.java.

3. **In the HighScoreList constructor, how is the argument `capacity` used?**

4. **Suppose a new HighScoreList is created and the scores 3, 8, and 5 are added. What are the values of the instance variables in the HighScoreList?**

5. **If a HighScoreList contains 5 scores, how many times does the for loop in `toString` run?**

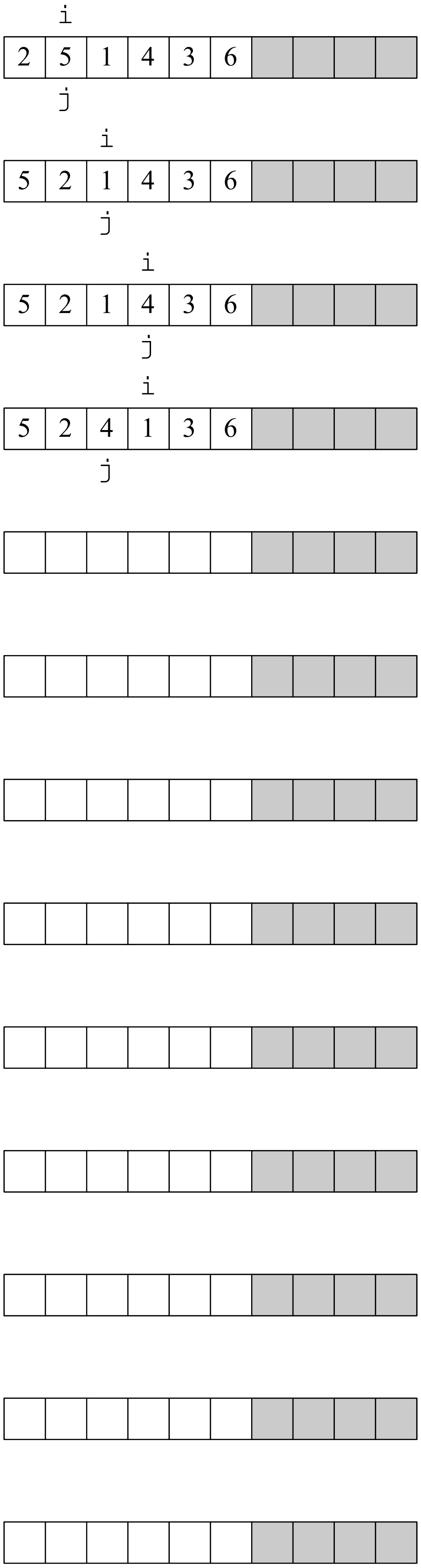
6. **What would happen if `addScore` was called on a “full” HighScoreList?**

7. **What alternate behavior seems natural to you in this situation?**

8. **How would you implement this behavior?**

Model 3: Insertion Sort

Suppose we run `testInsertionSortEasy`. The diagrams at right show the state of the HighScoreList data structure at the beginning of each pass through the inner for loop (line 46). The shaded region at the end of the array is unused because there are only six scores. While `i` and `j` are really just ints, they are shown here above and below the array because they are used as array indices. For example, in the first (top) diagram, `i` and `j` are both 1.



9. **Devise and describe a plan for filling in the remaining diagrams. Hint: if it's tedious, you're doing it wrong!**

10. **Carry out your plan.**

11. **What does each pass through the inner loop accomplish?**

12. **At the beginning of each pass through the outer loop, what is true of the part of the array to the left of element `i`?**

13. **What does each pass through the outer loop accomplish?**

14. **Would the method still work correctly if, in the outer loop, `i` started at 0 instead of 1? If so, is there a reason to prefer 1 over 0? If not, why not?**

15. **How would you modify the method to sort the scores in increasing order instead of decreasing order?**

Model 4: Mergesort

16. **The HighScoreList class has two overloaded versions of `mergeSort`. Which one is called directly by the tests?**

The diagram to the right shows the HighScoreList data structure at the beginning of a call to the recursive `mergeSort` method.

						hi			
2	5	1	4	3	6	▨	▨	▨	▨
lo									

17. **What value is chosen for `mid`?**

18. **Fill in the diagram at right to show the data structure after the *first* recursive call (line 97).**

						hi			
						▨	▨	▨	▨
lo									

19. **How did you determine your answer to the previous question?**

20. **Fill in the diagram at right to show the data structure after the *second* recursive call (line 98).**

						hi			
						▨	▨	▨	▨
lo									

21. **Draw the array returned by the subsequent call to `merge` (line 100).**

22. **Why does merge need a three-way if/else if/else statement? In other words, why can't it simply check if `scores[b] > scores[a]`?**

23. **`merge` creates a separate array for its results; the elements of this array are then copied back into `scores` by `mergeSort`. Why couldn't `merge` simply write each element directly into the proper place in `scores`?**

24. **How would you modify the method to sort the scores in increasing order instead of decreasing order?**

Reflection

Analyst: Reflect on your team's performance in the area of *information processing*.

25. **What were your team's strengths?**

26. **What were potential areas of improvement?**

27. **What insights did you gain?**