# Teacher’s guide

# Sequential and Binary Searches

**OBJECTIVES:** The student will code a solution for a sequential search of an unordered list.

The student will code a solution for a non-recursive and/or recursive binary search of an ordered list.

**ACTIVITIES/TIME:** One and a half Weeks

**MATERIALS:** Student Lesson A19: *Sequential and Binary Searches*

Lab Assignment A19.1, *Store*

Lab Lab Assignment A19.1, Data File, *file50.txt*

Lab Assignment A19.2, *Search*

Lab Assignment A19.3, *CountWords*

Lab Assignment A19.3, Data Files, *dream.txt, Lincoln.txt, test.txt*

Worksheet A19.1, *James Bond Database*

Worksheet A19.1, Data File, *bond.txt*

Worksheet A19.2, *James Bond Search*

Teacher’s Guide A19: *Sequential and Binary Searches*

Lab Assignment A19.1 – Answers, *Store.java, Item.java*

Lab Assignment A19.2 – Answers, *Store.java*

Lab Assignment A19.3 – Answers, *CountWords.java, WordCount.java*

Worksheet A19.1, *Answer Sheet*

Worksheet A19.2, *Answer Sheet*

**REFERENCES:**

**INSTRUCTOR**

**NOTES:** The binary search algorithm can be effectively demonstrated with the yellow pages edition of a large phone book. Have a student pick a topic to search (i.e., Bicycles-Dealers-New) and proceed to binary search through the alphabetized-by-topic yellow pages. You could also try an alternative search for a person's name in the white pages. Apply the alternating 'split-and-check' binary method until the topic is found or it is discovered that the item does not exist. Without getting into the order of the algorithm, discuss the effect of doubling the size of the yellow pages on the number of steps to find an item. You can also comment on the value of having the yellow pages sorted alphabetically by topic.

Prior to assigning Lab Assignment A19.1, *Store*, print out the contents of the data file "*file50.txt*", which is sorted by the *id* field. This assignment calls for the student to search for *id* values as selected by the instructor. To thoroughly test the binary search routine, it is suggested that you choose *id* values in the following locations:

1. The *id* value in position 0 of the array.

2. The *id* value at the end of the array.

3. Several other *id* values in the middle of the list.

4. Several *id* values that are not in the list:

a. An *id* value smaller than the first value in the list.

b. An *id* value that appeared in the middle of the list, but didn't match any of the values. (For example, in the data file, an id of 200 is between the first value 184 and the last value 19967, but does not exist in the data file.)

c. An *id* value larger than the last value in the list.

Students will need to have finished Lab Assignment A19.1, *Store* before they can start on Lab Assignment A19.2, *Search*, where a method that tests the binary search algorithm has been provided to speed up the lab work.

Lab Assignment A19.3, *CountWords,* could be challenging for some students. Here are some suggestions to ease the difficulty.

1. Provide the parseWords and validLetter functions from the answer sheet. Give the students a working program that simply prints out all the words in a text file.

2. Ask students to do a top-down analysis of the problem and determine needed methods including prototypes. This will provide some intermediate feedback about their approach to solving the problem.

A file containing Martin Luther King's famous "I Have a Dream" speech is provided, *dream.txt*. You are free to use a different text file if you choose. (Lincoln’s Gettysburg Address is also provided, *Lincoln.txt*).

**WORKSHEET**

**NOTES:** Worksheet A19.1, *James Bond Database* asks the students to set up a database of James Bond films. The students are asked to write code that does some averaging. Students are also challenged to write code that will search through the Bond film series (using a Sequential Search algorithm) to find a particular movie using three sets of criteria: lead actor, film rating and film length. They are asked to complete two methods that are started for them.