**LESSON SET 8**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Searching and Sorting Arrays**

**OBJECTIVES FOR STUDENT**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lesson 8A:**

1. To introduce the concept of a search routine

2. To introduce linear and binary searches

**Lesson 8B:**

3. To introduce the concept of a sorting algorithm

4. To introduce the bubble and selection sorts

**ASSUMPTIONS**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Lesson 8A:**

1. Since this lesson focuses on searching and sorting arrays, it is essential that students

have a solid understanding of arrays before attempting these labs.

Completion of Lesson Set 7 is a sufficient pre-requisite

2. A basic understanding of search routines

**Lesson 8B:**

1.A basic understanding of sort routines

**PRE-LAB WRITING ASSIGNMENT SOLUTIONS**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. simple (easy)

2. time-consuming or inefficient

3. more efficient

4. ordered

5. 1/8 (approximately)

6. bubble

7. selection

8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 19 | -4 | 91 | 0 | -17 |

Element 0 Element 1 Element 2 Element 3 Element 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -4 | 19 | 91 | 0 | -17 |

Element 0 Element 1 Element 2 Element 3 Element 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -4 | 19 | 0 | 91 | -17 |

Element 0 Element 1 Element 2 Element 3 Element 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -4 | 19 | 0 | -17 | 91 |

Element 0 Element 1 Element 2 Element 3 Element 4

1. Use the selection sort on the array below and construct the first 3 steps that actually make changes. (Assume the sort if from smallest to largest).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 19 | -4 | 91 | 0 | -17 |

Element 0 Element 1 Element 2 Element 3 Element 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -17 | -4 | 91 | 0 | 19 |

Element 0 Element 1 Element 2 Element 3 Element 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -17 | -4 | 0 | 91 | 19 |

Element 0 Element 1 Element 2 Element 3 Element 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -17 | -4 | 0 | 19 | 91 |

Element 0 Element 1 Element 2 Element 3 Element 4

**LAB ASSIGNMENTS**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lesson 8A:**

Lab 8.1: Working with the linear search

Lab 8.2: Working with the binary search

**Lesson 8B:**

Lab 8.3: Working with sorts

Lab 8.4: Student generated code assignments

**LESSON 8A**

**LAB 8.1: Working with the Linear Search**

Lab 8.1 requires the student to use program linear\_search.cpp from the Lab 8

folder (this is also Sample Program 8.1 from the Pre-lab Reading Assignment). This

program performs a linear search on a character array. Students are asked to rewrite

the program so that it searches an array of integers rather than characters.

Of course, they should only need to change a few lines of code to accomplish

this. Then they are asked to search the array

3 6 -19 5 5 0 -2 99

for several different integers including some that are in the array and others that are not. In particular they are required to check 5. They should note that only the first occurrence of 5 is returned.

A solved program is found in linear\_searchKEY.cpp in the instructor’s folder for Lesson 8.

The students are then asked to change the program so that the user can search for more than one number. The program uses a sentinel value to terminate the loop.

A solved program is found in linear\_searchKEY2.cpp in the instructor’s folder for Lesson 8.

**LAB 8.2: Working with the Binary Search**

Students will need to access binary\_search.cpp from the Lab 8 folder (this is also

Sample Program 8.2 from the Pre-lab Reading Assignment). This program performs

a binary search on an integer array. In Exercise 1 students are asked if the statement

middle=first+(last-first)/2 is an integer. They should realize that

mathematically middle may not be an integer, but the computer converts it to an

integer since middle is defined as an integer. Students should also see that the

number of elements in the list, whether it be even or odd, determines how the

middle element is found. In Exercise 2 students are asked to search the array

int array[]={34,19,19,18,17,13,12,12,12,11,9,5,3,2,2,0};

for 19 and 12, both of which are repeated in the array. Upon running the program,

the students will see output indicating that 19 is in position 2 of the list and

12 is in position 8. It is important that they realize position 2 corresponds to

array[1] and position 8 corresponds to array[7]. Then it is clear that the program

found the first occurrence of 19 whereas it found the second occurrence of 12.

Exercise 2 concludes by asking why there is a difference in which occurrence was

found. The answer, of course, lies in the way middle is computed as the algorithm

proceeds. Since they should have completed Exercise 1 at this point, the

role of middle in the execution of the program should be clear.

**LAB 8.3: Working with Sorts**

For this lab either bubble\_sort.cpp or selection\_sort.cpp needs to be brought

in from the Lab 8 folder (these are Sample Programs 8.3 and 8.4, respectively, of

the Pre-lab Reading Assignment). These two programs perform sorts on integer

arrays so that the integers are arranged from smallest to largest. Exercise 1 asks the

students to choose one of the programs and re-write it so that the sort orders the

integers from largest to smallest. Then in Exercise 2 the students are asked to

modify the same program so that it prints the array at each step of the sorting algorithm.

A test array is given so that students can try the algorithm by hand and then

compare the results to what their program produced.

Solved programs are found in selection\_sortKEY.cpp and bubbleSortKEY.cpp in the instructor’s folder for Lesson 8.

**LAB 8.4: Student Generated Assignments**

In this last lab students are asked to write a program that prompts the user to enter

the number of elements to be placed in an integer array that holds a maximum of 50 elements.

The program will then direct the user to input that many integers which it places in the array. A sort routine is then called to sort the data in the array. The student may use either the selection sort or the bubble sort; however the sort must be implemented in its own function and not in main. A function to perform a binary search must be included as well as a value returning function that computes the mean of the data set. The purpose of this activity is to have students write an interactive program that utilizes the algorithms from this lesson.

A solved program is found in Search\_SortKEY.cpp in the instructor’s folder for Lesson 8.

An exercise to modify the program so that the data is entered from a file rather than from the keyboard is given as an option.

A solved program for this option is found in Search\_SortKEY2.cpp in the instructor’s folder for Lesson 8.

Possible solutions to all labs are given in the instructor’s folder for Lesson Set 8.