Institut for Matematik og Datalogi
Syddansk Universitet

December 3, 2001
JFB

Introduction to Information Technology
E01 – Note 7

Lecture, November 16

We covered sections 3.5.2, 3.5.4 and most of 3.6 in chapter 3, plus gave an example from computational molecular biology on protein alignment. Spreadsheets and statistics in Maple were demonstrated.

Lecture, November 30

We will finish section 3.6 and begin on chapter 14, with emphasis on section 14.7 and cryptography. Sorting from the algorithms lab will be discussed, including the algorithm QuickSort.

Lecture, December 14

We will continue with cryptography: classical and public key cryptography, RSA, and digital signatures. Encryption and digital signatures with PGP will be demonstrated.

You can find a short introduction to cryptography and PGP through the WorldWideWeb. From the Web page for this course, you can find a link. The most relevant sections are "What is cryptography?", "Public key cryptography", "How PGP Works", "Keys", "Digital signatures", and "What is a passphrase?". If anyone ever asks you to certify his/her certificate, you should read the appropriate parts of that article before you do it.

Primary Lab 7 - for week 49

The goal of this lab is to help you to gain some understanding of the fact that most problems have more than one algorithmic solution and that these solutions can differ greatly as to how practical they are. You will experiment with three different sorting algorithms and compare them. Review section 3.5.2 in the textbook before coming to the lab.

To get approval for this lab, send an e-mail to your lab instructor with the answers to all of the questions asked below. The questions (some of which are requests for explanations) are in *italics* below.

To start up the program you should use, click on **Start** → **Programmer** → **Accessories** → **Command Prompt**. Then, in the window, type `javaw salsa.Salsa`. Note that in

its standard mode, the program sorts bars of different lengths, rather than numbers. It is easy to think of the bars as numbers, and it is easiest to see what is happening with the bars.

Exercise 1

In the window that opens, select **Algorithm Type** → **Sort**. Under **Sort Type**, choose **Selection Sort**. You can get a description of the algorithm by clicking on the + boxes at the left. The + means that the text can be expanded. Read about the algorithm. There is a major difference between this and the one described in the textbook (and in lectures). *What is the difference?*

Set the # **of Bars** to 8 and the **Speed** down to 1 (click on the left arrow or move the bar to the left). Click on **Step** and watch the algorithm execute (both the pseudocode on the left and the sorting of the bars on the right). *What does a red bar mean? A blue bar? A green bar? A yellow bar?*

Change the **Display Type** to **Number**. Set the **Speed** to 0. Click on **Start**. *What numbers do you get? (Write them down in the order they appear.)* Change the **Speed** to 1 again and click on **Step**. Click on **Pause** after three numbers have been sorted (placed in their final positions). *Write down the current order for the numbers.* (If the **Pause** button has changed to a **Resume** button, you may need to click once to change it back to **Pause** and again to actually pause.) Click on **Resume** to make the algorithm finish running.

Exercise 2

Running time of Selection Sort. Write down the current values for # **of Bars**, # **of Swaps**, and # **of Compares**. The analysis in the textbook (and done in lecture) says that the number of comparisons should be $\frac{1}{2}n^2 - \frac{1}{2}n$. (See page 92 in the textbook.) *How many comparisons should there theoretically be in this case, where* $n = 8$*? How does this compare with practice? Explain why there were 7 swaps.*

Switch back to bars for the **Display Type**, and increase the **Speed**. Try running Selection Sort with 25 bars, 50 bars, and 100 bars. *Write down in each case the* # **of Bars**, # **of Swaps**, *and* # **of Compares**. *How do these compare with the predicted values? What values would you expect if the number of bars was 10000?*

Exercise 3

Change the **Sort Type** to **Quick Sort**. Expand the description on the left to see the algorithm and the explanation. Note that the explanation under efficiency just assumes that the pivot element ends up in the middle every time. This does not happen every time, but a complete average case analysis is beyond the scope of this course. It is not so hard to show that if the pivot element is always within the middle 15/16 of the elements, then you get $O(n \log n)$ time anyway. Intuitively, it seems likely that this will happen most of

the time, so that is why one generally gets this behavior. Start the algorithm with 25 bars and **Speed** 1 to see how it works. *What are the blue bars? The red bars?*

Increase the **Speed**. Try running it three times with 25 bars, writing down the # **of Bars**, # **of Swaps**, and # **of Compares**. *Why didn't you get the same answer every time?*

Try running Quick Sort with 25 bars, 50 bars, and 100 bars. *Write down in each case the* # **of Bars**, # **of Swaps**, *and* # **of Compares**. *What do you conclude about Quick Sort's running time? Is* $O(n \log n)$ *believable? Try to make an estimate as to how long Quick Sort would take with 10000 bars.*

Exercise 4

Change the **Arrangement** of the bars to **Ascending**, so your initial data starts out sorted, instead of random. Try running Quick Sort with 25 bars. *How does this compare with Selection Sort? Explain your results.*

Exercise 5

Try another sorting algorithm, preferably one of the first three on the menu. *Which one did you try? How does it compare with Selection Sort and Quick Sort?*

Optional: Explain how your algorithm works and the running time.

Exercise 6

E-mail your answers to your lab instructor. Remember to logoff your computer, but do not push any of the buttons on it.