Institut for Matematik og Datalogi
Syddansk Universitet

September 16, 2009

Peter Schneider-Kamp

Introduction to Computer Science
E09 – Week 4

!!! Tuesday Labs moved from 14-16 to 12-14 !!!

The labs scheduled for Tuesday 14-16 will be moved to Tuesday 12-14 to avoid holes in the schedule. The change is effective starting from Tuesday, September 22.

Lecture: Monday, September 14

Daniel Merkle lectured on operating systems covering the rest of Chapter 3. Additional material for the assignment is available from Blackboard.

Lecture: Wednesday, September 16

Kim Skak Larsen lectured on databases. He covered Chapter 9.

Lecture: Monday, September 21, 12-14 (U20)

Lene Monrad Favrholdt will lecture on algorithms based on Chapter 5.

Lecture: Wednesday, September 23, 14-16 (U28)

Lene Monrad Favrholdt will lecture on algorithms based on Chapter 5 and give a brief introduction to Maple (note that there are notes on Maple on the course's homepage).

Lecture: Monday, September 28, 12-14

Rolf Fagerberg will lecture on Merging and Hashing.

Lecture: Wednesday, September 30, 14-16

Rolf Fagerberg will lecture on Merging and Hashing.

Lab: September 22, 12:15-14 (terminal room above U49)

Discussion in groups (only two, or possibly three, people per group, since you will sit at a computer):

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 different sorting algorithms and compare them. Review sections 5.4, 5.5, and 5.6 in the textbook (including problem 6 on page 250) before coming to the lab.

You will use a sorting simulator, written by Jacob Aae Mikkelsen

http://imada.sdu.dk/~petersk/DM526/sorting.jar

Download the file to your own directory. Then start it up by typing

```
java -jar sorting.jar &
```

Note that 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.

1. Begin with the algorithm **Insertion Sort**. Click on **Start sorting** and watch the algorithm execute. *What do the red and green rectangles mean? How many comparisons are done and how many copies? What is a copy?*

2. Repeat the experiment above. *Do you get the same results?* Try with **Decreasingly selected elements** and **Increasingly selected elements**. *What are the minimum and maximum number of comparisons possible with 8 elements? With n elements?* Note that after you understand what is happening, you can increase the speed, so that it takes less time to run.

3. Repeat both of the above two steps with Selection sort, Quick sort, Merge sort, and Randomized Quick sort. *Which appears to be fastest?*

4. *How do Selection sort, Quick sort, Merge sort, and Randomized Quick sort work?*

5. For Selection sort,if the number of bars is n, the number of comparisons should be
$$\sum_{i=2}^{n}(i-1) = \frac{1}{2}n^2 - \frac{1}{2}n.$$
Why is this the number of comparisons? How many comparisons should there theoretically be in this case, where $n = 16$? How does this compare with practice?

6. Now change from **Visualization of algorithms** to *Timetaking of the algorithms* and use the **Settings** menu to choose other array sizes (numbers of elements to sort). Try all the algorithms with 100, 1000, 10,000, 100,000, and 500,000 (for the slower algorithms, you might not want to try them three times for 500,000) elements and random data. For each algorithm, run it three times and compute the average **Number of comparisons**. *What does the "E" mean in these numbers? How do these compare with the predicted values (for Insertion sort and Selection sort)? What values would you expect if the number of bars was 1,000,000?* Run the program and save the results for the values 100, 200, 300, 400, 500 for Insertion sort and Quick sort, for your next lab.

7. Discuss which algorithm you would use in which situations (number of elements, almost sorted vs. random data, for example).

Lab: September 24, 14:15-16 (terminal room above U49)

Read the notes on Maple

http://imada.sdu.dk/∼petersk/DM526/maple.pdf

on the course's homepage before coming to this lab.

Do the following in groups of two (or three):

1. You can start Maple by typing `xmaple`. Start by typing `restart;`. The `restart` command is useful when you want to change your worksheet a little and execute it again; it clears all the variables and assignments. (When you want to execute the entire worksheet again, you can do it through the **Edit** menu button and **Execute**.)

   In the **Help** menu, click on **Take a Tour of Maple**. Then go through the **Ten Minute Tour** and the section further down on **Programming, Code Generation, and OpenMaple**. Try doing the things suggested in the Tour, including right clicking and using the Slider on the animation toolbar. When you differentiate and then integrate, do you get back the original function? Notice the similarity between the code produced in different programming languages.

2. Plotting data: Start up the statistical package in Maple by typing `with(Statistics);`. Define two lists, the first being [100, 200, 300, 400, 500], and the second being the number of comparisons by Insertion Sort from your last lab. Try plotting the points with `ScatterPlot(X,Y);` (assuming that is what you called your lists), or add a second argument `color=''Red''` to change to a different color. You can change the shape of the symbols marking the points or add a legend by right clicking on the plot.

   Now try fitting a curve to these points. You can try a quadratic curve with `ScatterPlot(X,Y,fit=[a*x^2 + b*x + c,x])`. To find the constants, you can use `Fit(a*x^2 + b*x + c,X,Y,x)`. Try at least one other type of curve, maybe linear.

   Repeat this with your data from Quick sort. Use $ax \log_2 x + bx + c$ as one of the curves you try.

3. Write a procedure for Insertion sort in Maple. Test it on some random data created with the **Matrix** palette on the left, using integers.

4. Write a procedure for Merge sort in Maple. Test it on some random data.

Assignment due 14:15, October 1

Late assignments will not be accepted. Working together is not allowed. You may write this either in English or Danish. Write clearly if you do it by hand. Even better, use LaTeX.

All of the questions in this assignment relate to the database in Figure 9.5 on page 444.

Question 1

In words, state the question that is answered by the following program segment:

```
TEMP1 ← SELECT from JOB where JobTitle = "Secretary"
TEMP2 ← JOIN TEMP1 and ASSIGNMENT
        where TEMP1.JobId = ASSIGNMENT.JobId
TEMP3 ← JOIN TEMP2 and EMPLOYEE
        where TEMP2.EmplId = EMPLOYEE.EmplId
RESULT ← PROJECT Name from TEMP3
```

Question 2

In each of the following three subquestions, you must write a sequence of instructions (using the operations `SELECT`, `PROJECT`, and `JOIN`) to retrieve the specified (and only the specified) information.

1. Retrieve the `Address` of Cheryl H. Clark.
2. Retrieve all departments (`Dept`) that have at least one secretary.
3. Retrieve the names (`Name`) of everybody who started on the date 1-1-2009.