08.71.23/24 Tölvunarfræði 2/2a

Final exam **May 5th, 2003**
Teacher: Hjálmtýr Hafsteinsson **Time: 9^{00} ? 12^{00}**

The first 5 problems are for all students (both from Tölvunarfræði 2 and 2a) Problem 6 is only for students in Tölvunarfræði 2, but problem 7 is only for students in Tölvunarfræði 2a (engineering students). In both cases ***five best problems out of six count****. All problems have the same value.*

All written materials and a calculator are allowed.

?? Note that when asked to "Describe" or "Show" then it is enough to do that in words and with drawings. If you are to write C++ code you will be asked for that specifically.
?? Give supporting arguments for all answers and remember that it is not necessary to write up definitions from the book.

1. Could the following `id` arrays have come up in the solution of the Union-Find problem if we use the *Quick-union* algorithm (Program 1.2 on page 15)? If an array is impossible, explain why, otherwise show a sequence of pairs that will result in it.

a)

0	1	2	3	4	5	6	7
2	3	6	2	4	2	0	1

b)

0	1	2	3	4	5	6	7
0	4	0	6	4	5	0	0

c)

0	1	2	3	4	5	6	7
5	5	5	5	5	5	5	5

2. Implement in C++ the function

```
void removeGT(node *&h, int val)
```

that takes as input a reference to the node pointer `h`, which points to a singly linked list without a head node. The function also receives the integer `val` and it should remove from the list all items with values greater than `val`.

Remember to deal with all special cases in the function (empty list, etc.).

3. We define a new kind of queue with the methods `insert` and `gettop10`. The method `gettop10` removes from the queue, and returns, one of the front 10% of the items in the queue ("front" means here "first inserted"). The item removed is chosen at random (using a random number generator) out of the front 10% of the items.

Describe in words and drawings an implementation of this type of queue and analyse the time (using *O*-notation) of the two methods. Your description has to be accurate enough so that it should be easy to program the implementation, but you do not have to show any code.

4. *a)* Where do identical values go in *heaps*, e.g. where would all items with the value `k` be in a heap? Explain in words and drawings.

b) How could we count all items with the value `k` in a heap without looking at all items in the heap (unless when it is necessary). Describe in words an algorithm to do this.

5. Implement in C++ the recursive function

```
int countEQ( node *h, int k)
```

that returns the number of items in the **binary search tree** `h` that have the key `k` (e.g. `"t->item.key() == k"`). Write two versions of the function:

a) assuming that insertion puts identical keys into the right subtree (this is how the insertion method on page 517 does it).

b) assuming that identical keys can go into either right or left subtree.

In both cases your function is only to visit nodes that are necessary to visit in order to count nodes with the key `k`. It is not valid to just go through all the nodes of the tree and count the ones that have the key `k`.

Aðeins fyrir nemendur í Tölvunarfræði 2:

6. The Insertion sort given in the book (page 276) uses a *sentinel* to simplify the condition of the `while`-loop. On the other hand a sentinel is not used in the Insertion sort used by Shell sort on page 286.

a) Explain carefully what needs to be done to change Shell sort to use a sentinel-based Insertion sort.

b) Change the Shell sorting program on page 286 so that it uses a sentinel (similar to the program on page 276).

Aðeins fyrir nemendur í Tölvunarfræði 2a (verkfræðinema):

7. In a singly linked circular list with a head node you can either have the head node a part of the circle, or outside the circle.

a) Describe these two versions with simple drawings.

b) How would an empty list be represented in each version?

c) Which version would be better for deleting a node with a particular value, and why is it better?