08.71.23/24 Tölvunarfræði 2/2a

Makeup exam **August 25th, 2004**
Teacher: Hjálmtýr Hafsteinsson **Time: $13^{30} – 16^{30}$**

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 ***the five best problems out of six count****. All problems have the same value.*

All written material and a calculator are allowed.

- Please note that when asked to "Describe" or "Show", then it is enough to do that using words and 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. Write a C++ function that receives a pointer `p` to a circular linked list and the integer `k`. The function is to remove from the list all nodes with values lower than `k` and build a new circular list from them. The function should then return a pointer to this new list. The function header is

```
node* SplitOnNumber(node* &p, int k)
```

Note that the parameter `p` is a reference to a pointer, because `p` might have to change.

2. A particular type of queue allows both insertion and deletion from both ends of the queue. The operation `putfront` inserts an item at the front of the queue (remember that `put` inserts an item at the back of the queue) and the operation `getback` removes the item at the back of the queue (remember that `get` removes the item at the front). Add these two new operations to the definition of a QUEUE and show their implementation in C++, **both** for the array implementation **and** the linked-list implementation.

3. A node in a binary tree is said to be *imbalanced* if there is a difference of two or more in the height of its children. The number of such nodes in a binary tree could be used as an indication of how balanced the tree is. Show a recursive C++ function that counts the number of imbalanced nodes in a given binary tree.

4. *a)* Is the result from an Alpha-Beta search always the same as the one from a Minimax search of a game tree? Justify your answer or show a counter-example.
b) Does it matter at all in Minimax search in which order we consider the children of a node? How about in Alpha-Beta search? Justify your answers.
c) Assume a game tree of height 5, where each inner node has 10 children. If the Alpha-Beta search eliminates 30% of the nodes at each level, how many nodes of the tree does Alpha-Beta look at compared to how many Minimax looks at?

5. Implement in C++ a new version of the heap function `getmax`. In this new version the hole that forms in the root is pushed down to the leaves by always moving up the child with the larger value. Then the last item in the heap (i.e. the item in `pq[N]`) is put into the hole and it moved upwards until the tree is heap ordered again.

Compare this method with the `getmax` function on page 386 in the textbook. Which method do you think is faster and why?

Only for the students of Tölvunarfræði 2:

6. One way to choose the partitioning element in Quicksort is to pick it at random. Then we call a random number generator to choose a number `k` between `l` and `r`, and use the item in location `a[k]` as the partitioning element. The problem is that good random number generators can take a long time. Calculate how often the generator would be called in Quicksort in the worst case. How about the best case?

Only for the students of Tölvunarfræði 2a (engineering students):

7. All the items of a heap are to be to put into a binary search tree. The heap is given as an *N*-item array and you can only use the operation `insert` on the binary search tree. *i)* Would it be sensible to insert the items in the order that they appear in the array, i.e. first `pq[1]`, then `pq[2]`, etc.? *ii)* How about the reverse order (i.e. start with `pq[N]`)? *iii)* Is there any other insertion order of items from the heap array that would be better than the two methods mentioned above? Justify all your answers.