08.71.23/24 Tölvunarfræði 2/2a

Final exam **May 7th, 2005**

Teachers: Anna Ingólfsdóttir and Hjálmtýr Hafsteinsson **Time: $9^{00} – 12^{00}$**

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

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. You are to write in C++ the function `MaxIncrSeq`, that receives as parameter a pointer to a node in a singly connected circular linked list. The function should find the length of the longest increment sequence in the list and return that value. An increment sequence is here defined as the longest sequence of increasing adjacent values. For example if the list is 14, 12, 8, 5, 10 then the longest increment sequence is 5, 10, 14. Below is the function header:

```
int MaxIncrSeq( node *h )
```

Explain your function well with comments and drawings, and remember to handle all special cases that can come up.

2. Write a recursive function in C++ that calculates partial sums for a *ternary tree*. The function receives a pointer to the root of the tree and returns the total sum of all the nodes of the tree. In addition it puts in a separate field in each node the partial sum of that subtree (i.e. the sum of the node and all its descendants). A node in a ternary tree is defined as follows:

```
struct tnode {
  int item;
  int psum;
  tnode *l, *m, *r;
}
```

Here the field `item` contains the value of the node, `psum` contains the partial sum (after it has been calculated), and the fields `l`, `m`, and `r` point to the (possibly) tree children of the node.

3. Let *A* be a set and < be an ordering on *A*. We define a binary tree as in the handout on compositional definitions and binary search tree as an ordered binary tree in the same handout; we call the set of binary search tree over *A* *TLT(A)*.

a) Assume that the set *A* above is the set of natural numbers (with zero). What does the following function do:

$F(\Lambda) = 0$

$F(\langle a, (T_1, T_2)\rangle) = F(T_1) + a + F(T_2)$

b) Define a function $F: TLT(A) \rightarrow TLT(A)$, that accepts as input a binary search tree and returns a binary search tree where the smallest item has been removed. The function should return an error if the input is an empty tree.

4. The main problem in Quicksort is how to choose the partitioning element. Describe in each instance what happens to the stability and time complexity of Quicksort if we can choose the partitioning element is the following ways:

a) We have an operation that returns the median element of an *n*-item list in O(1) time without moving the elements.

b) We have an operation that returns the median element of an *n*-item list in O(*n*) time without moving the elements.

c) We use Selection sort to sort half the list and thus find the median value.

5. Prove or disprove the following statement: The *k*-th largest element in a heap is never more than $\log_2(k)$ from the root of the heap.

6. *a)* Enter the following items in this order into an initially empty binary search tree: 1, 25, 20, 15, 10, 5. What is the height of the resulting tree?

b) Change the shape of the tree form part a) using rotations such that it becomes as low as possible for a binary search tree with 6 items. Show all the rotations that you do.
[Hint: Start by choosing which item you want as root.]