Name:

Algorithms and Data Structures
Jacobs University Bremen
Dr. Florian Rabe

Quiz 4
given: 2017-03-30

You have 20 minutes.

Problem 1

Points: 2+3+5

1. Give the Θ-class of the time complexity of inserting an element into a well-implemented heap (in terms of the number n of elements in the heap).
2. Concisely explain the commonality and difference between stacks and queues.
3. Implement the following function on iterators in pseudo-code:

```
fun forall[A](x : Iterator[A], p : A → bool) : bool =
```

Solution:

1. $\Theta(\log_2 n)$ (The condition of begin well-implemented is necessary because "heap" is just a specification. Any specification can be implemented arbitrarily inefficiently.)
2. Both are implementations of mutable lists that are optimized for a restricted set of operations, namely inserting/deleting at the end. Stacks allow inserting/deleting at the same end only; queues allow inserting at one and deleting at the other end.
3. Multiple implementations are possible. All nice ones call $x.getNext$ once and then call $forall(x, p)$ recursively. The base case must return *true* for the empty iterator. Efficient implementations return *false* as soon as p is *false* for an element of x (That's why we use && below.), i.e., they do not necessarily call p on all elements of x.

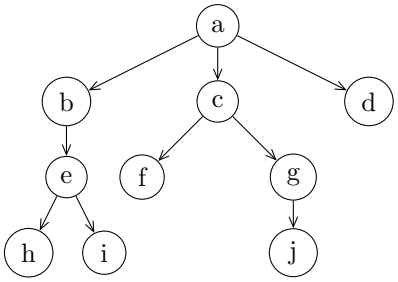
```
fun forall[A](x : Iterator[A], p : A → bool) : bool =
   if x.hasNext
      p(x.getNext) && forall(x, p)
   else
      true
```

Name:

Problem 2

Points: 1+2+3+(1+1+2)

1. Give the number of nodes in a perfect binary tree of height n.
2. Give an example of a heap of integers with respect to the $\leq$-ordering.
3. Briefly explain how DFS and BFS compare with respect to the expected run-time?
4. Consider the following tree:



Give the

(a) root.
(b) the descendants of c.
(c) list of elements in DFS-order.

Solution:

1. $2^{n+1} - 1$
2. Any tree with exactly one node.
3. If we need to iterate over all elements of the tree anyway, both take the same time. If we search for an element (i.e., we can stop once we found it), they differ: The run time of BFS is predictable because it only depends on the depth at which there is a solution; the run time of DFS can be much smaller or much larger depending on the order in which we recurse into the children of a node.
4. (a) a
   (b) c, f, g, j (ancestors would be a,c)
   (c) abehicfgjd (BFS would be abcdefghij)
