S. Schwoon / D. Suwimonteerabuth Winter Term 2008/09

Fundamental Algorithms

Exercise Sheet 2

1. Recall the algorithm multiplication *à la russe* from the previous exercise sheet:

   Input: Two positive integers m and n
   Output: $m \cdot n$

```
p := 0; M := m; N := n;
while m ≥ 1 do
    if m % 2 = 1 then
        p := p + n;
    fi
    m := ⌊m/2⌋;
    n := n + n;
od
return p;
```

   Prove that $mn + p = MN$ is a loop invariant of the above algorithm.

2. The Fibonacci numbers are a sequence of numbers beginning with 0 and 1, and each subsequent number is equal to the sum of the previous two numbers of the sequence. Formally, it is defined by the following recurrence:

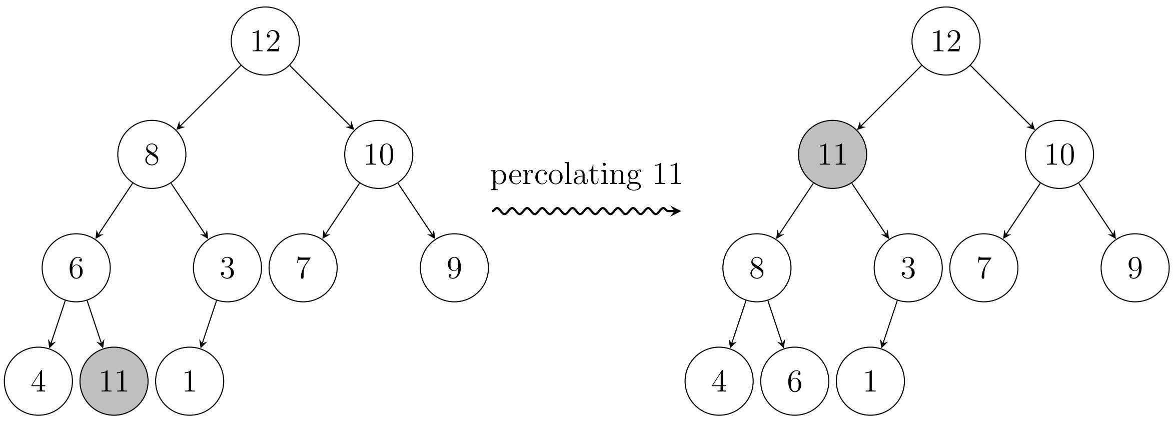
$$f_n = \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ f_{n-1} + f_{n-2} & \text{if } n \geq 1. \end{cases}$$

   Write down an algorithm using a for-loop that computes f_n for any natural number n. Prove the correctness of the algorithm and determine its complexity.

3. Given an array `a` and two indices `k` and `i`, the procedure `heapify` introduced in the lecture ensures that if initially the slice $\texttt{i}+1\ldots\texttt{k}$ is a heap, then afterwards the slice $\texttt{i}\ldots\texttt{k}$ is a heap. The idea is that if the value of the node `i` violates the heap property, i.e. its value `a[i]` is less than one of its children, the procedure exchanges the value with the larger value of its children. The process continues downwards in the forest until the heap property is no longer violated. We say the the value `a[i]` is *sifted down* to its new position.

   On the contrary, we now consider a new procedure that *percolate up* values in order to restore the heap property. The procedure, called `percolate`, takes two parameters: an array `a` and an index `i`. It ensures that if initially the slice $1\ldots\texttt{i}-1$ is a heap, then afterwards the slice $1\ldots\texttt{i}$ is a heap. The idea is that if the value of the node `i` is greater than the value of its parent, the procedure exchanges these two values. The process continues upwards in the forest until the heap property is no longer violated. For example, in the following left figure the node with value 11 violates the heap property. In this case, the procedure restores the heap property by exchanging the 11 with its parent

6 and then exchanging it again with its new parent 8, obtaining the result shown in the following right figure.



(a) Write down the algorithm `percolate` using the notation described in the lecture.

(b) Write down an algorithm that given an array it constructs a heap from the array by using the procedure `percolate`. Compare the algorithm with the one using `heapify`.