# Assignment 2

**Autumn 2014**

**Due at 23:59 on 05–12–2014**

**Requirements:** Assignments have to be handed in by each student separately. Please write your name and student number on the top of the first sheet that you hand in. Remember that assignments have to be typeset in English and submitted through the peach system. **Always justify your answers!**

You can score 20 points in total for this assignment; your grade will be the number of points you score divided by two.

---

**Problem 1:** [4 points] Consider the simple sorting algorithm shown below.

**Algorithm** BasicSort$(A, n)$
**Input:** array $A$ with $n \geq 0$ elements.
**Output:** The elements of A are sorted into nondecreasing order.

```
1.  for i = 1 to n
2.      do for j = 1 to n − i
3.          do if A[j] > A[j + 1]
4.              then Swap A[j] with A[j + 1]
5.  return A
```

We want to prove that this algorithm is correct.

(a) Prove that the following invariant holds for the inner loop (lines 2–4):
*At the start of step $j$, $A[j]$ contains the maximum of the subarray from $A[1]$ to $A[j]$.*
The useful property obtained from termination should simply be stated.

(b) Give an invariant for the outer loop, and use it to prove that BasicSort correctly sorts the array $A$. Note that you should use the termination property of the inner loop to prove the maintenance of your loop invariant.

**Problem 2:** [4 points] Are the following statements true or false? Briefly explain your answers.

(a) $(n + 4)^2 = \Theta(n^2)$

(b) $n + \sqrt{n} = \Theta(n \log n)$

(c) $2^{2n} = O(2^n)$

(d) $n^3 - 4n^2 - 15 = \Omega(n^2)$

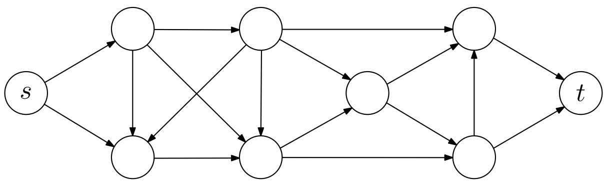
**Problem 3:** [4 points] We define the problem CountInInterval as follows: *Given an array $A$ of $n$ integers, and two integers $p$ and $q$, count the number of elements of $A$ that are at least $p$ and at most $q$.*

(a) Give an algorithm for CountInInterval. What is the running time of your algorithm?

(b) Now assume that $A$ is sorted. Using this assumption, give an algorithm for CountInInterval that runs in $O(\log n)$ time. Don't forget to prove the running time of your algorithm.

**Problem 4:** [4 points] We define the problem CountPaths as follows: *Given a special type of directed acyclic graph (DAG) $G = (V, E)$ with exactly one vertex $s$ without incoming edges (the source) and exactly one vertex $t$ without outgoing edges (the sink), count the number of distinct (directed) paths from $s$ to $t$.*

(a) How many paths are there from $s$ to $t$ in the DAG in the figure? Explain your method of counting.




(b) For a vertex $v$, let $c[v]$ be the number of paths from $s$ to $v$, and let $\text{In}[v]$ be the set of vertices that have an edge to $v$. Express $c[v]$ in terms of $c[u]$ for $u \in \text{In}[v]$. Argue why your formula is correct.

(c) The number of paths from $s$ to $t$ is simply $c[t]$, but can only be computed if $c[u]$ has been computed for all $u \in \text{In}[t]$. Give an algorithm for CountPaths that runs in $O(n + m)$ time, where $n$ is the number of vertices and $m$ is the number of edges. The input consists of $G = (V, E)$, $s$, and $t$. You may use sets $\text{In}[v]$ and counts $c[v]$ (if computed) in your algorithm. *(Hint: Order the nodes cleverly.)*

**Problem 5:** [4 points] Run Dijkstra's algorithm on the directed graph in the figure, first using vertex $s$ as source and then using vertex $w$ as source. Show each iteration in the style of the textbook (page 95).