Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

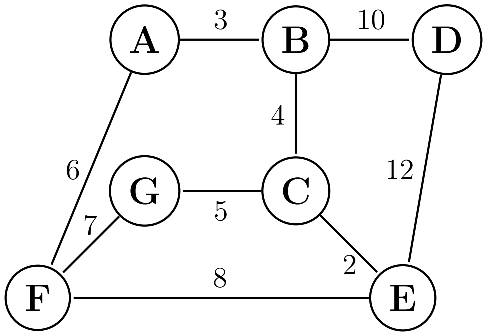
Institut für Theoretische Informatik
Peter Widmayer
Thomas Tschager
Antonis Thomas

27th April 2016

Datenstrukturen & Algorithmen — Exercise Sheet 9 — FS 16

Exercise 9.1 *Minimum Spanning Trees.*

Consider the following graph $G = (V, E)$.



a) Use Kruskal's algorithm to compute a minimum spanning tree for the graph below. Mark the edges contained in your solution.

b) In a graph, a cycle is a walk starting and ending at the same vertex. A graph is *acyclic* if it contains no cycles. Show that every undirected acyclic graph $G = (V, E)$ has at most $|V| - 1$ edges.

c) Show that the minimum spanning tree of a graph is uniquely determined if no edge weight occurs twice.

Exercise 9.2 *Union-Find Structures.*

In union-find structures, a set is represented by a tree. We consider the process of "unification by size" (see Chapter 6.2.2). We want to create a tree of height $h \in \mathbb{N}$. Describe how such a tree is generated with a sequence of Union operations. How many Union operations are necessary at least, and how many nodes does the resulting tree have?

Exercise 9.3 *Longest increasing subsequence.*

The following Java code contains an incomplete implementation of a dynamic program for the longest increasing subsequence problem. We are given a sequence $A = A[0], \ldots, A[n-1]$ with pairwise different numbers $A[i] \in \mathbb{N}^+$ for $i = 0, \ldots, n-1$. We compute a one-dimensional table T with $n+1$ entries. The entry $T[i]$ for $i > 0$ contains the smallest $A[j]$ with $j \in \{0, \ldots, n-1\}$, such that $A[j]$ is the last element of an increasing subsequence of length i. Initially, $T[0]$ is set to $-\infty$ and all other entries to ∞. In the i-th step, we search for the largest index $j < i$ with $T[j] < A[i]$. We can use binary search to compute this index, because the entries of T are always monotonically increasing. If there is no such entry, we set $j = 0$. Then, we update the entry

$T[j+1] = \min(T[j+1], A[i])$. After n steps, the length of the longest increasing subsequence is the largest index k with $T[k] \neq \infty$.

Complete the following Code snippet (lines $4, 5, 17$, and 18).

```
1  static int binarySearch(int A[], int l, int r, int key) {
2      while (l < r) {
3          int m = l + (r - l + 1)/2;
4          if (A[m] >= key) ___________;
5          else ___________;
6      }
7      return l;
8  }
9
10 static int computeTable(int A[], int size) {
11     int[] T = new int[size+1];      // DP-Tableau
12     for (int i = 1; i <= size; i++)  // Initialisierung
13         T[i] = Integer.MAX_VALUE;
14     T[0] = Integer.MIN_VALUE;
15     int l = 0;
16
17     for (int i = ____; i ______; i = ____) {
18         int j = binarySearch(___, ___, ___, ___);
19         if ( A[i] < T[j+1] ) T[j+1] = A[i];
20         if (l < j+1) l = j+1;
21     }
22     return l;
23 }
```

Exercise 9.4 *Minimum spanning tree problem.*

In this exercise we are going to implement Kruskal's algorithm for the *minimum spanning tree problem*. This problem is defined as follows. Given an undirected graph $G = (V, E)$ with the edge cost function $c : E \to \mathbb{Q}^+$, we search for an acyclic connected subset $T \subseteq E$ with $|T| = |V| - 1$ (i.e., a tree (V, T)) whose total cost $\sum_{e \in T} c(e)$ is minimum among all possible trees. The following image shows an example where the bold edges form a minimum spanning tree of the graph.

Input The first line contains only the number t of test instances. After that, we have exactly one line per test instance containing the description of the input graph $G = (V, E)$ in the form $n, m, u_1, v_1, c_1, \ldots, u_m, v_m, c_m$. We have $1 \leq n, m \leq 10000$ with $V = \{1, \ldots, n\}$ and $|E| = m$. For every i, $1 \leq i \leq m$, the numbers $u_i, v_i \in \{1, \ldots, n\}$ define the edge $\{u_i, v_i\} \in E$ having a cost of c_i, $1 \leq c_i \leq 1000$.

Output For every test instance we output only one line. This line contains the cost of a minimum spanning tree.

Example

Input:

```
1
10 15 1 2 1 2 3 2 1 4 7 2 5 2 3 6 3 3 7 5 4 5 4 5 6 3 6 7 2 4 8 1 5 9 3 6 9 4 7 10 6 8 9 5 9 10 3
```

Output:

```
21
```

Directions We provide you with a template. It contains the necessary code to read the input. Implement the class `UnionFind` and Kruskal's algorithm. There is only one testset for 100 point in this exercise.

Hand-in: Wednesday, 4th May 2016 in your exercise group.