Prof. Dr. A. Goerdt
A. Lanka
L. Falke

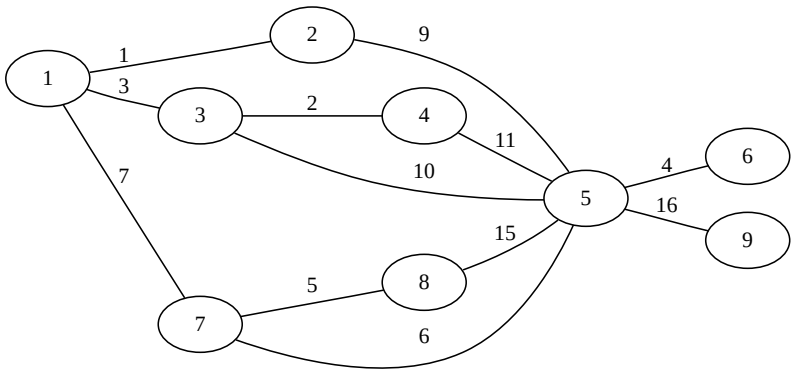
TU Chemnitz
Wintersemester 2009/2010
17.11.2009

Theoretische Informatik I

6. Übung

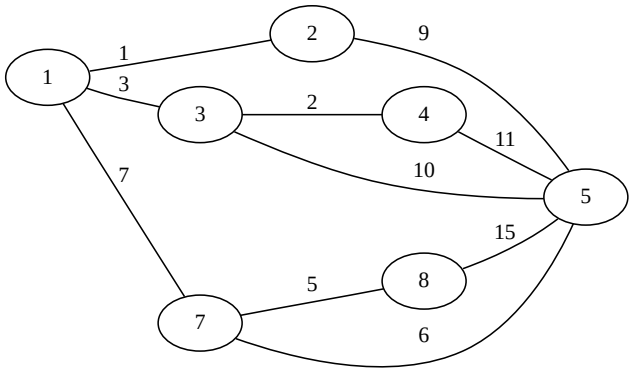
Geben Sie die Lösung der Aufgabe *2* bitte bis zum *27.11.2009* bei Ihrem Übungsleiter ab.

1. Aufgabe: Bestimmen Sie mit Hilfe *Kruskals Algorithmus* den minimalen Spannbaum des folgenden Graphen.



Benutzen Sie die Union-Find-Datenstruktur der Vorlesung sowie die Heuristiken *Union-By-Size* und *Wegkompression*.

2. Aufgabe: Bestimmen Sie mit Hilfe *Prims Algorithmus* (mit Heap) den minimalen Spannbaum des folgenden Graphen.



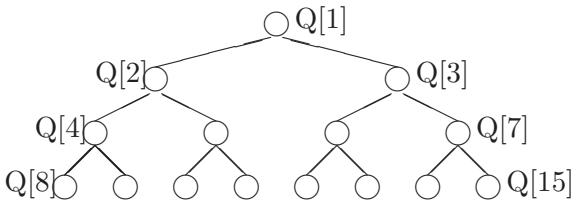
Geben Sie für jeden Schritt den Inhalt der Arrays `key` und `kante` an und beginnen Sie den Algorithmus bei Knoten `5`.

3. Aufgabe: Wir betrachten einen Heap, der in einem Array $Q[1..n]$ implementiert ist. Zeigen Sie folgende *Vater-Sohn-Beziehung*:

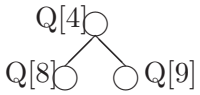
> Steht in einem Heap ein Element an Position i, dann steht der linke Sohn des Elements an Position $2 \cdot i$ und der rechte Sohn an Position $2 \cdot i + 1$.

4. Aufgabe: Zeigen Sie, dass der folgende Algorithmus die Laufzeit $O(n)$ hat.

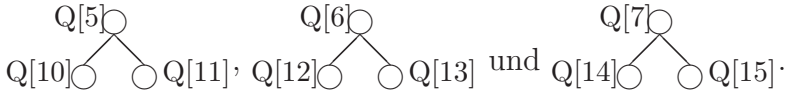
> Gegeben sei ein Feld $Q[1..n]$ von Elementen. Wir wollen aus Q einen Heap aufbauen. Wir nehmen $n = 2^k - 1$ an und betrachten Q als Baum (z. B. $k = 4$):



> Der Heap wird nun *bottom-up* aufgebaut. Das heißt, zuerst wird in dem Teil



> die Heap-Eigenschaft hergestellt, indem $Q[4]$ an die richtige Stelle sickert. Genauso verfahren wir mit den Teilen



> Danach lassen wir $Q[2]$ bzw. $Q[3]$ an die richtige Stelle sickern, um in den Teilbäumen

> die Heapeigenschaft herzustellen. Schließlich entsteht durch Sickern von $Q[1]$ ein korrekter Heap.

Hinweis: Überprüfen Sie, wie oft ein Knoten aus Ebene i maximal sickern kann, bis er an der richtigen Stelle steht. Summieren Sie diesen Wert über alle Knoten. Die entstehende Summenformel kann auf eine geometrische Reihe zurückgeführt werden, wenn man die Ungleichung $i < 1,5^i$ benutzt.