Aufgabe 1:

1. Eine ArrayList funktioniert intern wie folgt:

   - Die ArrayList besteht aus einem "normalen" Array und einem int–Attribut mit dem Index des ersten unbesetzten Elements. Zu Beginn hat das Array die Länge 10 und der Index den Wert 0.
   - Bei jedem Anfügen wird das Array an der Stelle des Indizes auf den Wert des neuen Elements gesetzt. Anschließend wird der Index um 1 erhöht.
   - Ist das Array allerdings schon voll und ein weiteres Element soll angefügt werden, dann wird ein Array der doppelten Größe erzeugt und alle Element werden umkopiert. Danach wird das neue Element angehängt.

   Schreiben Sie eine Klasse MyArrayList<T>, die eine ArrayList für generische Daten implementiert.

   Fügen Sie folgende Methoden hinzu:

   (a) Anfügen (add).

   (b) Auslesen (get).

   (c) Löschen des Feldes (clear). Außerdem wird die Feldgröße wieder auf 10 gesetzt.

   (d) Rückgabe der Anzahl der Elemente (size).

   (e) Einfügen vorne (addFirst).

   Was müssten Sie tun, um ein Element vorne einzufügen? Welche $\mathcal{O}$-Klasse hat dieser Algorithmus?

   Testen Sie ihre Implementation!

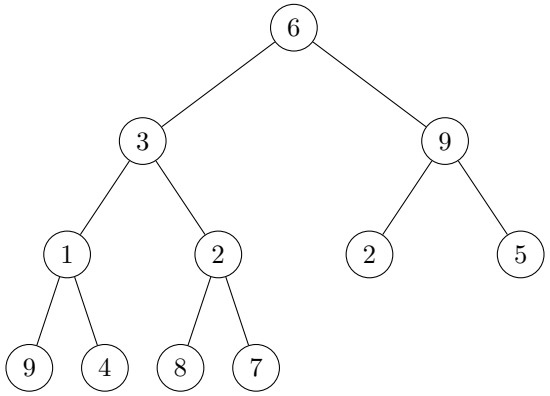

Figure 1: Links-vollständiger Binärbaum.

Aufgabe 2:

1. Gegeben sei ein vollständiger Binärbaum $B = (V, E)$ mit $|V| = n$ und Höhe h. Beantworten Sie kurz folgende Fragen:
   - Wie groß ist $|E|$ in Abhängigkeit von n?
   - Wie viele Blätter hat B in Abhängigkeit von h?
   - Wie viele Blätter hat B unabhängig von h?
   - In welchem Verhältnis stehen h und n zueinander?
2. Geben Sie für den links-vollständigen Binärbaum aus Figure 1 die Array Darstellung an:
3. Geben Sie für den links-vollständigen Binärbaum aus Figure 1 die Reihenfolgen der Zahlen an, wie sie nach Preorder, Postorder und Inorder durchlaufen werden: