1. Assume that the nodes of a binary search tree maintain the size of the subtree rooted at that node, i.e., finding the size of a subtree requires \( O(1) \) time. Also assume the keys in the tree are distinct. Describe an algorithm which given an integer \( i \) between 1 and \( n \) returns the \( i \)th smallest key in the \( n \)-node tree. E.g., if \( i = 1 \) it should return the minimum key and if \( i = n \) it should return the maximum key. Your algorithm should run in \( O(h) \) time in the worst-case, where \( h \) is the height of the tree.

2. (a) Draw the balanced (2,4) search tree that would result when items with the following key values were inserted into an initially empty tree: 1,2,3,4,5,6,7,8,9.

(b) Repeat part (a) using an AVL tree.

3. 10,000 items are to be stored using a hash table with external chaining. You are told in advance that \( 4/5 \) of the successful searches (search hits) that are performed are searches for items among a group of 2,000 frequently accessed items. The remaining \( 1/5 \) of the search hits are for items among the remaining 8,000 items. Assume the 2,000 frequently accessed items are known in advance. Two strategies are proposed for storing the items:

(a) All 10,000 items are to be stored in a single table of size 1,000.

(b) The 2,000 frequently accessed items are stored in a table of size 500 and the other 8,000 in a second table of size 500. When a search is performed, the first table is examined first. If the item is not found the second table is tried.

Assume the hash functions randomly distribute the keys over the tables and that unordered lists are used for the external chains. What is the expected number of items examined by each method in the case of a search hit? What is the expected number of items examined in the case of a search miss?
4. Assume that you are hashing a $k$-bit integer using a hash function which takes the integer mod $M$ where $M$ is a prime number. Show that the result of the hashing depends on every bit of the integer in the following sense: Show that for each bit position, there are two keys that differ in only that bit, which are mapped to different places in the table.

5. A playing card is described by two integer quantities: the suit, which is a number between 0 and 3; and the rank, which is a number between 0 and 12. Derive a perfect hash function which takes as input a playing card $(a, b)$ and outputs a value between 0 and 51, i.e., the hash function should assign each card a different hash value.