1. State true or false: (5 points)
   - The inorder traversal of a binary search tree T will visit its nodes in sorted search-key order.
   - The minimum height of a binary tree with n nodes is $log_2(n + 1)$.
   - The average case complexity of insertion in a binary search tree is $O(log_2 n)$.
   - Updating the key item of a Table node is not permitted.
   - A heap can become unbalanced with insertions and deletions over time.

2. Given the following tree categories: (a) binary tree, (b) complete binary tree, (c) full binary tree, (d) general tree, and (e) binary search tree, apply all the applicable categories to the following trees. (5 points)
3. Arrange the following nodes in a binary search tree form in the order given: 25, 6, 4, 30, 2, 55, 6, 60, 12. (4 points)
Is the tree balanced? Why or why not? (2 points)
Print out the nodes in (a) pre-order, (b) post-order, (c) in-order. (3 points)

4. Given the following binary search tree, illustrate the binary tree structure after each of the following node deletions in the order: 6, 22, 27. (5 points)
5. Given a pointer-based representation of binary tree (as on class slides 30–47), write an algorithm to deallocate all nodes in the tree. The TreeNode structure is as below. (5 points)

```cpp
class TreeNode {
private:
    TreeNode() {};
    TreeNode(const int& nodeItem, TreeNode *left = NULL, TreeNode *right = NULL): item(nodeItem), leftChildPtr(left), rightChildPtr(right) {};
    int item; // data portion
    TreeNode *leftChildPtr; // pointer to left child
    TreeNode *rightChildPtr; // pointer to right child
    friend class BinaryTree; // friend class
};
```

6. Illustrate (with figure) the array representation of the following binary tree. (5 points)

```
22
 /   \
16    34
 /  \
10  16
 /  \
\5  17
```

7. Explain the Table implementation you will choose in the following cases (sorted/unsorted array/LL, BST). Why? (4 points)
   (a) English dictionary; only needs a retrieval operation.
   (b) Word processor spell checker; compares words in your document with words in its dictionary; can add new words to its dictionary when necessary; needs frequent retrieval and occasional insertion.

8. Why is a binary search tree not particularly suitable for implementing a priority queue? What data structure would you choose instead? Why? (5 points)
9. Use open hashing and the hash-function: \( h(x) = x \mod 11 \) to insert the following elements in the hash-table: 3, 5, 7, 82, 58, 69, 113.
\( (3 \text{ points}) \)

10. Use closed hashing to and the hash-function: \( h(x) = x \mod 11 \) to insert the following elements in the hash-table: 3, 5, 7, 82, 58, 69, 113.
\( (4 \text{ points}) \)
Use the following collision resolution techniques (one at a time): (a) Linear probing (b) Quadratic probing
11. Given the following max-heap, illustrate the individual steps in which the heap changes for each for the following operations: (a) Insert 16, (b) delete. (5 points)