1 Question 1.

Suppose we need a data container for storing multiple distinct integers, and we need to frequently run the following operations:

- searching, insertion, and deletion
- selection: identifying the i-th largest integer

We decide to implement a binary search tree (just a normal binary search tree, not AVL), as binary search tree's average performance is quite good for searching, insertion, and deletion. Then, to make selection more efficient, we decide that our tree nodes will store three pieces of information (the value itself, plus two additional pieces of book-keeping information):

- The integer value that we need to store
- How many nodes are in the left child branch
- How many nodes are in the right child branch

We call our new data structure *ADSA Binary Search Tree* (ABST). Now you are asked to fill in the implementation details. We are looking for algorithms that are efficient and elegant. This is an open ended question without a specific standard answer.

- Given an ABST, how to find the i-th largest integer? Please provide pseudocode.

  (2 marks: 1 for correctness and 1 for efficiency and elegance)

- How to perform insertion to an ABST? Please provide pseudocode. Note that you need to maintain the correctness of all book-keeping information, i.e, for every node, you need to update how many nodes are in the child branches.

  (4 marks: 2 for correctness, 1 for efficiency, 1 for elegance)

- How to perform deletion from an ABST? To make this problem easier, you only need to consider the case where the node to be deleted has only one child branch. Please provide pseudocode.

  (4 marks: 2 for correctness, 1 for efficiency, 1 for elegance)

2 Question 2.

Q2.1: A hash function, $h(x)$, hashes a name, x (the hash key), onto a hash value (the index into an array) as listed in the following table. The hash table has a length of 10 (indexed from 0 to 9).

The keys are inserted into a hash table in the order listed in the following table.

y	h(y)
Alice	3
Bob	6
Carol	2
Dave	3
Eve	7
Trent	6
Walter	0

Show the table contents after insertion with:

- chaining (2 marks)
- linear probing (2 marks)

Q2.2: In lectures, on insertion into a skiplist we flipped a coin until a *head* occured to determine the height of an element. So for example, if we flipped two tails before a head then we would insert an element of height 3. In all the examples in lectures the probability of flipping a head was exactly 1/2.

Now assume that the coin is biased so the probability of flipping a head on a given throw is now 9/10. Derive an expression for the probability of producing an element of height h with this biased coin. Describe in broad terms the consequence of this bias in terms of the average cost of insertion into the skiplist. (2 marks)

Q2.3: State the storage requirements of a graph with n nodes and m edges using:

1. an adjacency list (1 marks), and
2. an adjacency matrix (1 marks)

briefly justify each answer. (2 marks)