



## 4847 - Binary Search Tree

Asia - Daejeon - 2010/2011

A binary search tree is a binary tree. It may be empty. If it is not empty, it satisfies the following properties:

- (1) Every node has a key, and no two nodes have the same key.
- (2) The keys in a nonempty left subtree must be smaller than the key in the root of the subtree.
- (3) The keys in a nonempty right subtree must be larger than the key in the root of the subtree.
- (4) The left and right subtrees are also binary search trees.

Sample binary search trees are shown in Figure 1.

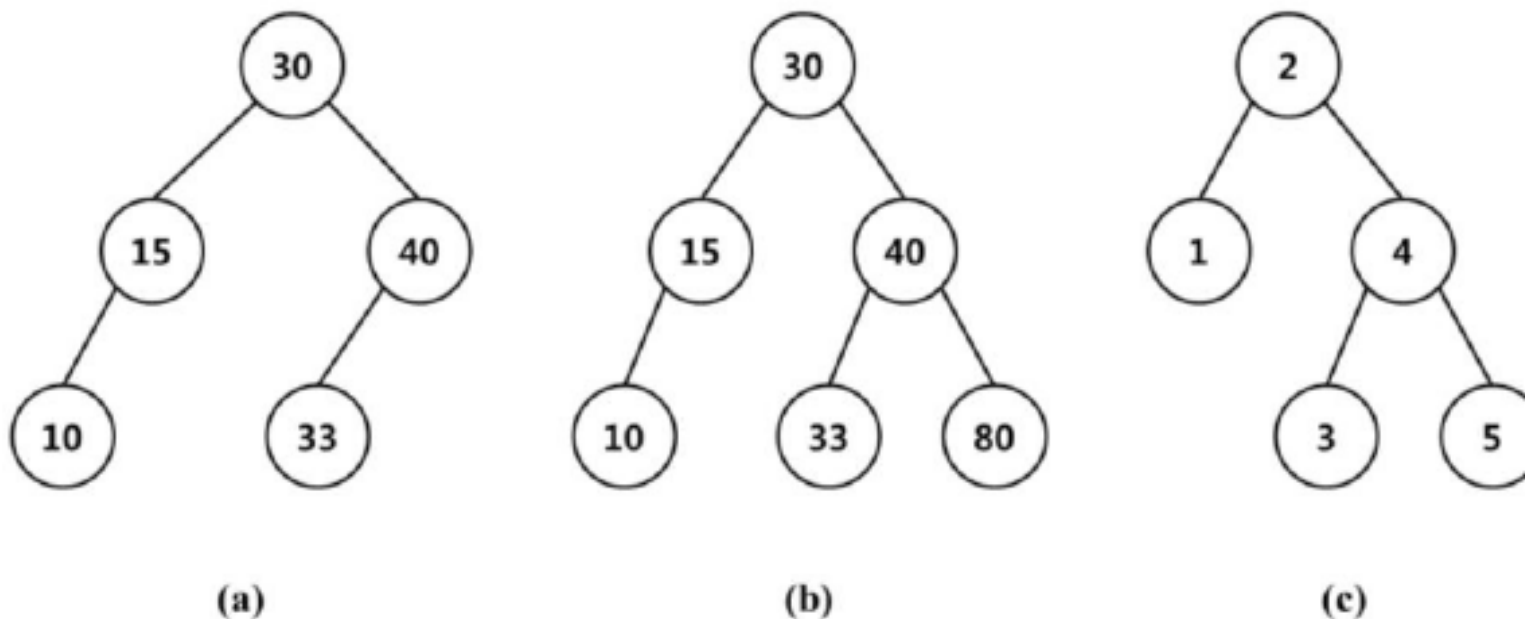


Figure 1. binary search trees

To search for a node with a key  $k$  in a binary search tree  $T$ , we begin at the root. If  $T$  is empty,  $T$  contains no keys and the search is unsuccessful. Otherwise, we compare  $k$  with the key in root. If  $k$  equals root's key, then the search terminates successfully. If  $k$  is less than root's key, we search the left subtree of the root. If  $k$  is larger than root's key, we search the right subtree of the root. In the same way, we can proceed the search in the left or right subtree of  $T$ .

To insert a new key  $k$  into a binary search tree  $T$  where  $k$  is different from those of existing keys in  $T$ , we first search the tree  $T$ . The search will be unsuccessful, then we insert the key at the point the search terminated. For instance, to insert a key 80 into the Figure 1(a), we first search the tree for 80. This search terminates unsuccessfully, and the last node examined has key 40. We insert a new node containing 80 as the right child

of the node. The resulting search tree is shown in Figure 1(b).

In this problem, we consider binary search trees with  $N$  keys  $1, 2, \dots, N$ . For a permutation  $a_1 a_2 \dots a_N$  of  $\{1, 2, \dots, N\}$ , inserting  $a_1 a_2 \dots a_N$  successively into an initially empty binary search tree will produce a binary search tree. For instance, the permutation  $2\ 1\ 4\ 3\ 5$  will produce the tree in Figure 1(c). Also,  $2\ 4\ 3\ 1\ 5$  will produce the same tree. Actually, 8 permutations among all possible permutations of  $1, 2, 3, 4, 5$  will produce the same tree to the tree in Figure 1(c).

We are interested in finding the number of permutations of  $\{1, 2, \dots, N\}$  such that all those permutations produce a binary search tree identical to the tree produced by a given permutation  $P$ . Given  $N$  and  $P$ , you are to write a program that calculates the number of permutations satisfying the above condition.

## Input

Your program is to read from standard input. The input consists of  $T$  test cases. The number of test cases  $T$  is given in the first line. Each test case starts with a line containing an integer  $N$  representing the number of keys,  $1 \leq N \leq 20$ . In the next line, a permutation of length  $N$  is given. There is a single space between the integers representing keys in the permutation.

## Output

Your program is to write to standard output. Print exactly one line for each test case as follows: Let  $B$  be the number of permutations that produce the binary search tree identical to the tree produced by the input permutation. Print  $B \bmod 9,999,991$  for each test case. For example, if  $B = 20,000,000$ , the output should be 18 for that test case.

The following shows sample input and output for three test cases.

## Sample Input

```
3
5
2 1 4 3 5
4
2 4 1 3
12
1 2 3 4 5 6 7 8 9 10 11 12
```

## Sample Output

```
8
3
1
```

---

Daejeon 2010-2011