A phone directory needs a data structure with the following operations:

- **init** creates an empty directory
- **add(n, p)** adds name n with number p to directory
- **del(n)** deletes name n from directory
- **find(n)** gives the number of n
- **update(n, p)** updates the number of n to be p
- **list** lists number-name pairs in alphabetical order

1. Implement data structure and its operations based on *array*. Analyze the complexity of operators.

2. Implement data structure and its operations based on *linked list*. Analyze the complexity of operators.

3. How could one implement efficiently operator **find name(number)** which returns name of given number.
   Operation **find(name)** should remain to be efficient even after the addition. Implement the new operator and analyze its complexity.

4. Alter the implementations in such a way that one person can have multiple phone numbers. How does the complexity of operators change?

5. Implement operation **subdir(l1, l2)** which compares two directories l1 and l2, and returns true if and only if all name-number pairs in l1 are found also in l2. What is the complexity of the operator?

In above implement means a pseudo code level implementation.