Finding all candidate keys for a given relation

Given a particular relation and a set of functional dependencies over it, your program should output the list of all candidate keys. The key list should be sorted based on size. Within the keys of same size, the sorting order should follow the alphanumeric order (same as the dictionary order).

Example input 1:

```
1.  6
2.  4
3.  1 2
4.  4
5.  4
6.  3
7.  1 2
8.  5
9.  2
10. 6
```

Line 1: Number of attributes in the relation. This relation has six attributes: 1, 2, 3, 4, 5, and 6. In general, if the relation has N attributes then they are 1, 2, 3,……N.

Line 2: Number of functional dependencies. This relation has four functional dependencies.

From line 3 onwards, functional dependencies (FD) are described. Each FD is described using two lines. First line describes the left side or antecedent and the second line describes the right side or consequent. If the antecedent or the consequent consists of multiple attributes then they are separated by a space. For example lines 3 and 4 describe a FD 1,2 -> 4. The other FDs are:

4-> 3

1,2 -> 5

2 -> 6

In general, if there are M FDs on a relation, there will be 2M lines to describe those FDs.

1. 1
2. 1 2

Example output 1:

Line 1: Number of candidate keys. This relation has one candidate key.

From line 2 onwards, each candidate key is described using one line. If the candidate key consists of more than one attribute, then the attributes are separated by space. For example the candidate key for this relation is 1,2.

Note on code submission:

Make sure that all your code fits into a single file <roll number>.cpp

How your code should compile?

```
g++  <roll number>.cpp
```

How your code should run?

```
a.out < input_file_name
```

Your code should write the output on the stdout

A TA will collect your code on a USB drive. Make sure that you compute md5 hash of your code file before you submit. Use command md5sum <filename>.

Submit code to TA only when you are sure that it is your final code. We will not entertain requests to accept newer version of the code.

Preparing your system:

Any Linux flavor.

GNU C++ compiler version 11.2