

Lab 11: Permutation

Date: 4 September 2009

Number of Problems: 1

Objective

- The student must be able to enumerate every permutation of objects.

Guessing Password

Password is a very common form of authentication. Valid users know their own password while an adversary is supposed to know nothing about the password. However, the adversary can break into the system simply by trying out every possible password for the system. Luckily, this method is very time consuming, there are approximately 100 difference characters in standard US-English keyboard and assuming that the average password length is 8 characters. These amounts to 100^8 possible passwords, guessing all passwords would take enormous time.

However, if some information about the password is known, guessing password is simpler. For example, if we know that the password consists only of a numeric, the number of possible passwords greatly reduced. A common technique for password theft is Shoulder Surfing, an act that an adversary looks over the shoulder of a valid user, trying to remember the password that the valid user is typing.

In this problem, we will take the role of password hacker using shoulder surfing. Let us assume that the hacker knows the characters used in the password but not their order. We will generate all possible passwords from the known information and test whether it is the actual password.

The student has been given a set of .RAR files (an archived file similar to .ZIP), all of which are protected by a password. For each .RAR file, there will be an accompanying text file giving the information of the password, which is the character and the number that that character appears in the password. Your task is to try to find the password of each .RAR file.

The Input

The input is given as a text file. The first line in the text file gives the name of the .RAR file. The next line gives the number N of distinct character used in the password for that .RAR file. It is followed by N lines, each line contains a character c_i and r_i indicating that the character c_i appears r_i times in the password.

The Output

No mandatory output is required. Your task is to find the password. If the password is right, you will be able to open the .RAR files.

Example

Input

```
test.rar <-- The .RAR file is named "test.txt"
2 <-- there are two distinct characters in the password
a 2 <-- the letters "a" appears twice in the password
b 1
```

This input indicates that the possible password for "test.rar" are "aab", "aba" and "baa".

The Starting Code

As always, the starting code can be downloaded at <http://www.nattee.net/2110327/2552-a>. The source code has only one file, "main.c". The code already read the password information from the text file. There is a function `int testUnpack(char* filename, char* password)` that test whether the .RAR file in `filename` can be opened by the password in `password`. Please see inside the starting code for the usage example.

The code has five .RAR files. The password information is given in the file named "profile*.txt".

Generating All Permutation

Essentially, a permutation of a set is a particular order of elements in the set. For example, let the set be $\{a, b, c\}$, some examples of permutation of this set are (c, b, a) or (b, c, a) . A following code can be used to generate all permutations of a set $\{1, 2, 3, \dots, N\}$. The generated permutation is stored in the array `sol`.

```
void permu(int step, int *sol, bool *used)
{
    if (step < N) {
        for (int i = 0; i < N; i++) {
            if (!used[i]) {
                used[i] = true;           // mark that the number I is being used
                sol[step] = i;           // set the solution at position [step] to be i
                permu(step, sol, used); // recursive
                used[i] = false;        // reset the mark
            }
        }
    } else {
        print(sol);                      // display the permutation
    }
}
```

Remark

The given code simply generates all permutation of a set of non-repeating members. Be noted that our problem has repeating members. You need to modify the given code.

Grading Sheet

Name: _____	Date: _____
ID: _____ Section: _____	Time: _____

Lab 11: Permutation

Task Grading

Task	Description	Grader Signature	Remark
1	Complete the problem		

Question

1. In the file "profile5.txt", what is the number of possible passwords? How many passwords are generated by your program before it could find the actual password?

2. (Optional) What is in the file "price.rar"?