

# Password Cracker FP ★

Points: 553.180000000001 Rank: 1150

Problem Submissions Leaderboard Editorial

### RATE THIS CHALLENGE



There are N users registered on a website CuteKittens.com. Each of them have a unique password represented by pass[1], pass[2], ..., pass[N]. As this a very lovely site, many people want to access those awesomely cute pics of the kittens. But the adamant admin don't want this site to be available for general public. So only those people with passwords can access it.

Yu being an awesome hacker finds a loophole in their password verification system. A string which is concatenation of one or more passwords, in any order, is also accepted by the password verification system. Any password can appear 0 or more times in that string. He has access to each of the N passwords, and also have a string loginAttempt, he has to tell whether this string be accepted by the password verification system of the website.

For example, if there are 3 users with password {"abra", "ka", "dabra"}, then some of the valid combinations are "abra" (pass[1]), "kaabra" (pass[2]+pass[1]), "kadabraka" (pass[2]+pass[3]+pass[2]), "kadabraabra" (pass[2]+pass[3]+pass[1]) and so on.

#### **Input Format**

First line contains an integer T, the total number of test cases. Then T test cases follow.

First line of each test case contains N, the number of users with passwords. Second line contains N space separated strings, pass[1] pass[2] ... pass[N], representing the passwords of each user. Third line contains a string, loginAttempt, for which Yu has to tell whether it will be accepted or not.

### Constraints

- $1 \le T \le 10$
- $1 \le N \le 10$
- $pass[i] \neq pass[j], 1 \leq i < j \leq N$
- $1 \leq length(pass[i]) \leq 10$ , where  $i \in [1,N]$
- 1 < length(loginAttempt) <= 2000
- loginAttempt and pass[i] contains only lowercase latin characters ('a'-'z').

#### **Output Format**

For each valid string, Yu has to print the actual order of passwords, separated by space, whose concatenation results into loginAttempt. If there are multple solutions, print any of them. If loginAttempt can't be accepted by the password verification system, then print WRONG PASSWORD.

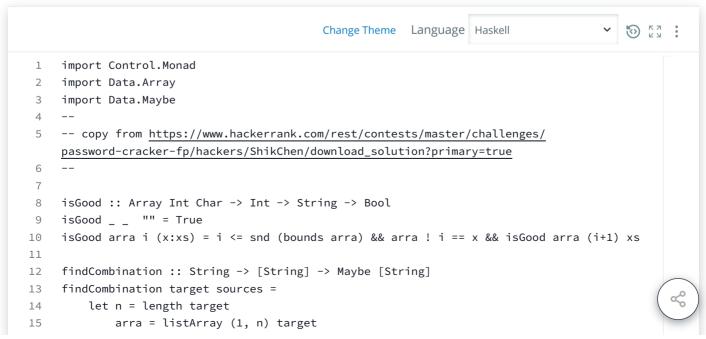
## Sample Input 0

3
6
because can do must we what wedowhatwemustbecausewecan 2
hello planet helloworld 3
ab abcd cd abcd

# Sample Output 0



```
we do what we must because we can
  WRONG PASSWORD
  ab cd
Explanation 0
Sample Case #00: "wedowhatwemustbecausewecan" is the concatenation of passwords {"we", "do", "what", "we", "must", "because",
"we", "can"}. That is
  loginAttempt = pass[5] + pass[3] + pass[6] + pass[5] + pass[4] + pass[1] + pass[5] + pass[2]
Note that any password can repeat any number of times.
Sample Case #01: We can't create string "helloworld" using the strings {"hello", "planet"}.
Sample Case #02: There are two ways to create loginAttempt ("abcd"). Both pass[2] = "abcd" and pass[1] + pass[3] = "ab cd" are
valid answers.
Sample Input 1
  3
  ozkxyhkcst xvglh hpdnb zfzahm
  zfzahm
  gurwgrb maqz holpkhqx aowypvopu
  gurwgrb
  aaaaaaaaaab
Sample Output 1
  zfzahm
  gurwgrb
  WRONG PASSWORD
```



```
dp = listArray (1, n+1) (map f [1..n] ++ [Just []])
16
17
             f idx = listToMaybe $ mapMaybe try $ filter (isGood arra idx) sources
                 where try pre = case dp ! (idx + length pre) of
18
                                      Nothing -> Nothing
19
                                      Just rs -> Just (pre:rs)
20
          in dp ! 1
21
22
    main = do t <- readLn</pre>
23
24
               forM_ [1..t] (\_ -> do n <- readLn :: IO Int
                                       sources <- fmap words getLine</pre>
25
26
                                       target <- getLine</pre>
                                       let output = case (findCombination target sources) of
27
                                                                                         Line: 34 Col: 1
                                                                           Run Code
                                                                                         Submit Code
Test against custom input
```

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature

