

Name: Santhosh G

Reg no: 240701473

Week 11: String Handling Function

1. What is your phone number?

Problem statement:

These days Bechan Chacha is depressed because his crush gave him list of mobile number some of them are valid and some of them are invalid. Bechan Chacha has special power that he can pick his crush number only if he has valid set of mobile numbers. Help him to determine the valid numbers.

You are given a string "S" and you have to determine whether it is Valid mobile number or not. Mobile number is valid only if it is of length 10 , consists of numeric values and it shouldn't have prefix zeroes.

Input:

First line of input is T representing total number of test cases.

Next T line each representing "S" as described in in problem statement.

Output:

Print "YES" if it is valid mobile number else print "NO".

Note: Quotes are for clarity.

Program:

```
1 #include<stdio.h>
2 #include<string.h>
3 int main()
4 {
5     int t;
6     scanf("%d",&t);
7     while(t-->0)
8     {
9         char s[20];
10        scanf("%s",s);
11        if((strlen(s)==10)&&(s[0]!='0'))
12        {
13            printf("YES\n");
14        }
15        else
16        {
17            printf("NO\n");
18        }
19    }
20    return 0;
21 }
```

	Input	Expected	Got	
✓	3	YES	YES	✓
	1234567890	NO	NO	
	0123456789	NO	NO	
	0123456.87			

Passed all tests! ✓

2. Alice and string

Problem statement:

Two strings **A** and **B** comprising of lower case English letters are compatible if they are equal or can be made equal by following this step any number of times:

- Select a prefix from the string **A** (possibly empty), and increase the alphabetical value of all the characters in the prefix by the same valid amount. For example, if the string is **xyz** and we select the prefix **xy** then we can convert it to **yx** by increasing the alphabetical value by 1. But if we select the prefix **xyz** then we cannot increase the alphabetical value.

Your task is to determine if given strings **A** and **B** are compatible.

Input format

First line: String **A**

Next line: String **B**

Output format

For each test case, print **YES** if string **A** can be converted to string **B**, otherwise print **NO**.

Program:

```
1 #include<stdio.h>
2 #include<string.h>
3 int main()
4 {
5     char a[1000000],b[1000000];
6     int flag=1;
7     scanf("%s",a);
8     scanf("%s",b);
9     int c=strlen(a);
10    int d=strlen(b);
11    if(c==d)
12    {
13        for(int i=c-1;i>=0;i--)
14        {
15            while(a[i]!=b[i])
16            {
17                for(int j=0;j<=i;j++)
18                {
19                    if(a[j]<'z')
20                        a[j]++;
21                    else
22                    {
23                        flag=0;
24                        break;
25                    }
26                }
27                if(flag==0)
28                    break;
29            }
30        }
31    }
32    else
33        flag=0;
34    if(flag==0)
35        printf("NO");
36    else
37        printf("YES");
38    return 0;
39 }
```

Test cases:

	Input	Expected	Got	
✓	abaca cdbda	YES	YES	✓

Passed all tests! ✓

3. Pizza confusion

Problem statement:

Joey loves to eat Pizza. But he is worried as the quality of pizza made by most of the restaurants is deteriorating. The last few pizzas ordered by him did not taste good :(. Joey is feeling extremely hungry and wants to eat pizza. But he is confused about the restaurant from where he should order. As always he asks Chandler for help.

Chandler suggests that Joey should give each restaurant some points, and then choose the restaurant having **maximum points**. If more than one restaurant has same points, Joey can choose the one with **lexicographically smallest** name.

Joey has assigned points to all the restaurants, but can't figure out which restaurant satisfies Chandler's criteria. Can you help him out?

Input:

First line has N, the total number of restaurants.

Next N lines contain Name of Restaurant and Points awarded by Joey, separated by a space. Restaurant name has **no spaces**, all lowercase letters and will not be more than 20 characters.

Output:

Print the name of the restaurant that Joey should choose.

Program:

```
1 #include<stdio.h>
2 #include<string.h>
3 int main()
4 {
5     int n;
6     scanf("%d",&n);
7     char res[n][27];
8     int rate[n];
9     for(int i=0;i<n;i++)
10    {
11        scanf("%s",res[i]);
12        scanf("%d",&rate[i]);
13    }
14    int max=rate[1];
15    char ans[20];
16    strcpy(ans,res[0]);
17    for(int i=1;i<n;i++)
18    {
19        if(rate[i]>max)
20        {
21            max=rate[i];
22            strcpy(ans,res[i]);
23        }
24        else if(rate[i]==max)
25        {
26            if(strcmp(res[i],ans)<0)
27                strcpy(ans,res[i]);
28        }
29    }
30    printf("%s",ans);
31    return 0;
32 }
```

Test cases:

	Input	Expected	Got	
✓	3 Pizzeria 108 Dominos 145 Pizzapizza 49	Dominos	Dominos	✓

Passed all tests! ✓

4. Password

Problem statement:

Danny has a possible list of passwords of Manny's facebook account. All passwords length is odd. But Danny knows that Manny is a big fan of palindromes. So, his password and reverse of his password both should be in the list.

You have to print the length of Manny's password and it's middle character.

Note: The solution will be unique.

INPUT

The first line of input contains the integer N, the number of possible passwords.

Each of the following N lines contains a single word, its length being an odd number greater than 2 and lesser than **14**. All characters are lowercase letters of the English alphabet.

OUTPUT

The first and only line of output must contain the length of the correct password and its central letter.

Program:

```
1 #include<stdio.h>
2 #include<string.h>
3 int main()
4 {
5     int n,flag=0;
6     char temp[10];
7     scanf("%d",&n);
8     char words[n][14];
9     for(int i=0;i<n;i++)
10     scanf("%s",words[i]);
11     char reverse[14];
12     for(int i=0;i<n;i++)
13     {
14         strcpy(reverse,words[i]);
15         int size=strlen(reverse);
16         for(int k=0;k<size/2;k++)
17         {
18             temp[0]=reverse[k];
19             reverse[k]=reverse[size-k-1];
20             reverse[size-k-1]=temp[0];
21         }
22         for(int j=i+1;j<n;j++)
23         {
24             if(strcmp(reverse,words[j])==0)
25             {
26                 flag=1;
27                 break;
28             }
29         }
30         if(flag==1)
31             break;
32     }
33     int len=strlen(reverse);
34     printf("%d %c",len,reverse[len/2]);
35     return 0;
36 }
```

Test cases:

	Input	Expected	Got	
✓	4 abc def feg cba	3 b	3 b	✓

Passed all tests! ✓