# Assignment #2: 字符串相关

Updated 0014 GMT+8 Sep 20, 2023

2023 fall, Complied by 刘思瑞 元培学院

**说明：**

1）第2周课上讲到了计算机相关的历史，介绍了ASCII表。

2）请把每个题目解题思路（可选），源码Python, 或者C++/C（已经在Codeforces/Openjudge上AC），截图（包含Accepted, 学号），填写到下面作业模版中（推荐使用 typora https://typoraio.cn，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

3）提交时候先提交pdf文件，再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、作业评论有md或者doc。

4）同学完成作业的时候，就是这个模版文件中修改补充好。为便于助教批改作业，请尽量不要删除其他文字。

5）如果不能在截止前提交作业，请写明原因。

**编程环境**

操作系统：Windows 11 22H2 22621.2283

Python编程环境：Visual Studio (1.82.2); python 3.11.3

C/C++编程环境：无

# 1. 必做题目

## 71A. Way Too Long Words

strings, 1000, http://codeforces.com/problemset/problem/71/A

Sometimes some words like "*localization*" or "*internationalization*" are so long that writing them many times in one text is quite tiresome.

Let's consider a word *too long*, if its length is **strictly more** than 10 characters. All too long words should be replaced with a special abbreviation.

This abbreviation is made like this: we write down the first and the last letter of a word and between them we write the number of letters between the first and the last letters. That number is in decimal system and doesn't contain any leading zeroes.

Thus, "*localization*" will be spelt as "*l10n*", and "*internationalization*" will be spelt as "i18n".

You are suggested to automatize the process of changing the words with abbreviations. At that all too long words should be replaced by the abbreviation and the words that are not too long should not undergo any changes.

**Input**

The first line contains an integer $n$ ($1 \le n \le 100$). Each of the following $n$ lines contains one word. All the words consist of lowercase Latin letters and possess the lengths of from 1 to 100 characters.

**Output**

Print $n$ lines. The $i$-th line should contain the result of replacing of the $i$-th word from the input data.

Examples

input

```
4
word
localization
internationalization
pneumonoultramicroscopicsilicovolcanoconiosis
```

output

```
word
l10n
i18n
p43s
```

【刘思瑞，元培物理方向，2023年秋】

思路：

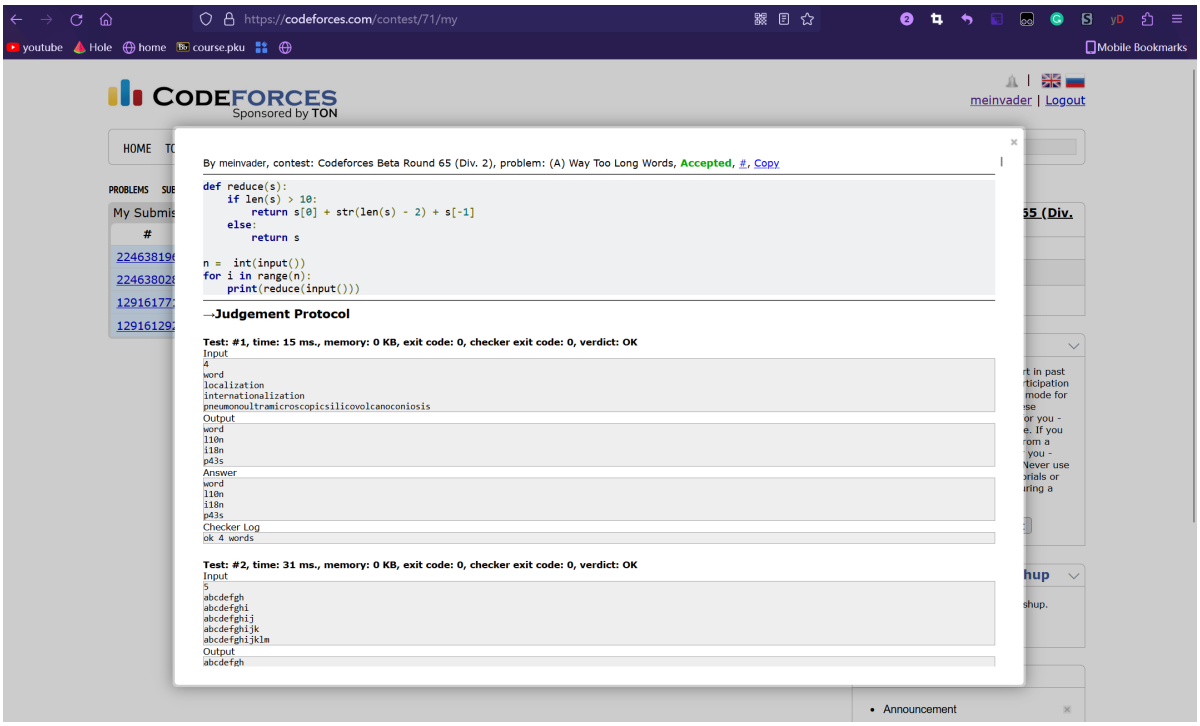
直接利用字符串的切片功能即可

**Python3 代码**

```
'''
刘思瑞 2100017810
'''
def reduce(s):
    if len(s) > 10:
        return s[0] + str(len(s) - 2) + s[-1]
    else:
        return s

n =  int(input())
for i in range(n):
    print(reduce(input()))
```

Python代码运行截图




## 112A. Petya and Strings

implementation/strings, 800, http://codeforces.com/problemset/problem/112/A

Little Petya loves presents. His mum bought him two strings of the same size for his birthday. The strings consist of uppercase and lowercase Latin letters. Now Petya wants to compare those two strings lexicographically. The letters' case does not matter, that is an uppercase letter is considered equivalent to the corresponding lowercase letter. Help Petya perform the comparison.

### Input

Each of the first two lines contains a bought string. The strings' lengths range from 1 to 100 inclusive. It is guaranteed that the strings are of the same length and also consist of uppercase and lowercase Latin letters.

### Output

If the first string is less than the second one, print "-1". If the second string is less than the first one, print "1". If the strings are equal, print "0". Note that the letters' case is not taken into consideration when the strings are compared.

Examples

input

```
aaaa
aaaA
```

output

```
0
```

input

```
abs
Abz
```

output

```
-1
```

input

```
abcdefg
AbCdEfF
```

output

```
1
```

Note

If you want more formal information about the lexicographical order (also known as the "dictionary order" or "alphabetical order"), you can visit the following site:

- http://en.wikipedia.org/wiki/Lexicographical_order

【刘思瑞，元培物理方向，2023年秋】

思路：

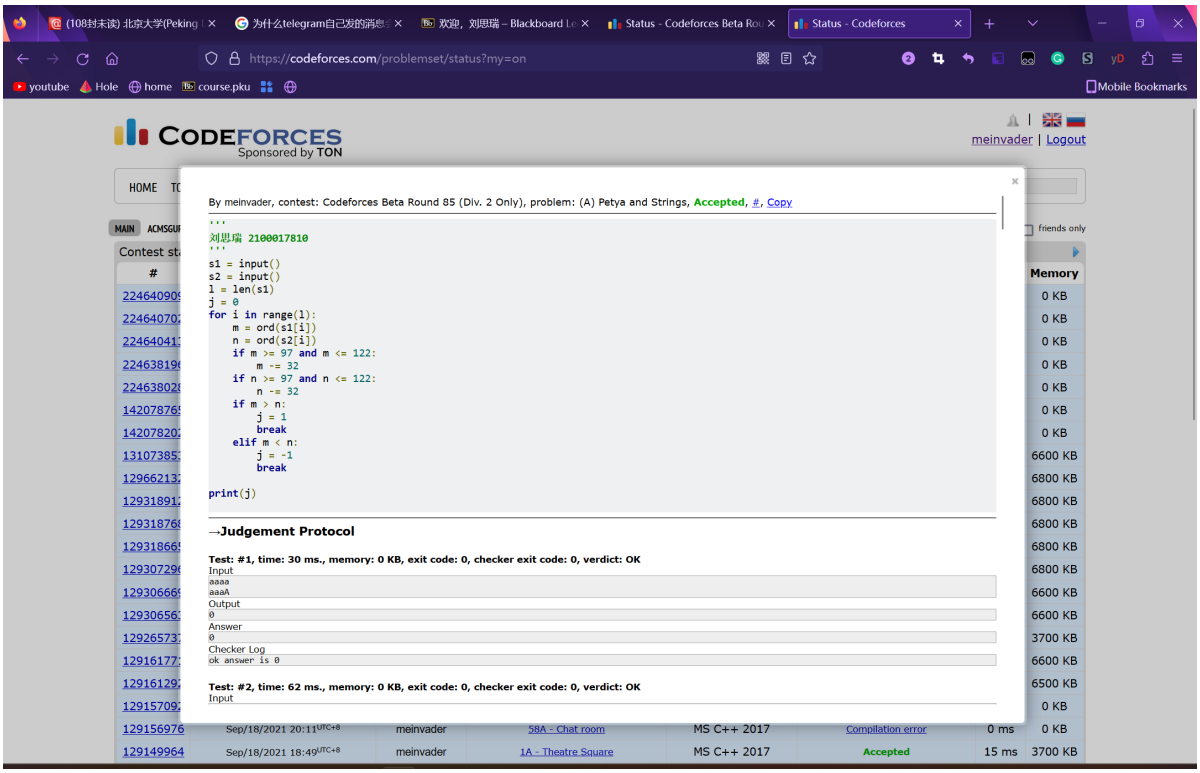
还是直接利用字符串切片依次比较即可

**Python3 代码**

```python
'''
刘思瑞 2100017810
'''
s1 = input()
s2 = input()
l = len(s1)
j = 0
for i in range(l):
    m = ord(s1[i])
    n = ord(s2[i])
    if m >= 97 and m <= 122:
        m -= 32
    if n >= 97 and n <= 122:
        n -= 32
    if m > n:
        j = 1
        break
    elif m < n:
        j = -1
        break

print(j)
```

Python代码运行截图




## 158A. Next Round

*special problem/implementation, 800, http://codeforces.com/problemset/problem/158/A

"Contestant who earns a score equal to or greater than the *k*-th place finisher's score will advance to the next round, as long as the contestant earns a positive score..." — an excerpt from contest rules.

A total of *n* participants took part in the contest ($n \geq k$), and you already know their scores. Calculate how many participants will advance to the next round.

**Input**

The first line of the input contains two integers *n* and *k* ($1 \leq k \leq n \leq 50$) separated by a single space.

The second line contains *n* space-separated integers $a_1, a_2, ..., a_n$ ($0 \leq a_i \leq 100$), where $a_i$ is the score earned by the participant who got the *i*-th place. The given sequence is non-increasing (that is, for all *i* from 1 to *n* - 1 the following condition is fulfilled: $a_i \geq a_i + 1$).

**Output**

Output the number of participants who advance to the next round.

Examples

input

```
8 5
10 9 8 7 7 7 5 5
```

output

```
6
```

input

```
4 2
0 0 0 0
```

output

```
0
```

Note

In the first example the participant on the 5th place earned 7 points. As the participant on the 6th place also earned 7 points, there are 6 advancers.

In the second example nobody got a positive score.

【刘思瑞，元培物理方向，2023年秋】

思路：

其实直接写就好，但是要注意到分数大于0这个条件，我加了$max(omega,1)$来确保这个条件，但是实际上还是在前面判断更节省复杂度，我这里这样写只是为了省点代码hhhh

**Python3 代码**

```python
'''
刘思瑞 2100017810
'''
n , k = map(int, input().split())
contester = list(map(int, input().split()))
omega = contester[k-1]
count = 0
for i in range(n):
    if contester[i] >= max(omega,1):
        count += 1
print(count)
```

Python代码运行截图

By meinvader, contest: VK Cup 2012 Qualification Round 1, problem: (A) Next Round, **Accepted**, #, Copy

```python
'''
刘思瑞 2100017810
'''
n , k = map(int, input().split())
contester = list(map(int, input().split()))
omega = contester[k-1]
count = 0
for i in range(n):
    if contester[i] >= max(omega,1):
        count += 1
print(count)
```

## 58A. Chat room

greedy/strings, 1000, http://codeforces.com/problemset/problem/58/A

Vasya has recently learned to type and log on to the Internet. He immediately entered a chat room and decided to say hello to everybody. Vasya typed the word *s*. It is considered that Vasya managed to say hello if several letters can be deleted from the typed word so that it resulted in the word "hello". For example, if Vasya types the word "ahhellllloou", it will be considered that he said hello, and if he types "hlelo", it will be considered that Vasya got misunderstood and he didn't manage to say hello. Determine whether Vasya managed to say hello by the given word *s*.

**Input**

The first and only line contains the word *s*, which Vasya typed. This word consisits of small Latin letters, its length is no less that 1 and no more than 100 letters.

**Output**

If Vasya managed to say hello, print "YES", otherwise print "NO".

Examples

input

```
ahhellllloou
```

output

```
YES
```

input

```
hlelo
```

output

```
NO
```

【刘思瑞，元培物理方向，2023年秋】

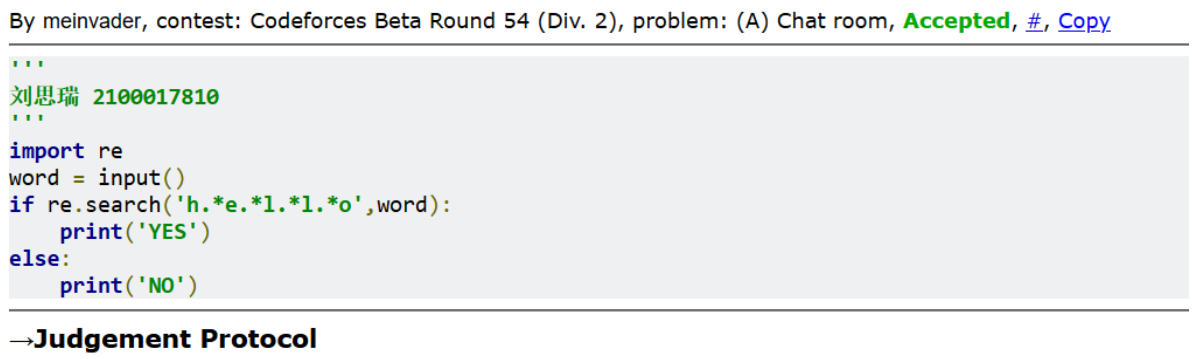
思路：

正则表达式直接秒杀

**Python3 代码**

```
'''
刘思瑞 2100017810
'''
import re
word = input()
if re.search('h.*e.*l.*l.*o',word):
    print('YES')
else:
    print('NO')
```

Python代码运行截图




## 04015: 邮箱验证

strings, http://cs101.openjudge.cn/practice/04015

POJ 注册的时候需要用户输入邮箱，验证邮箱的规则包括：
1)有且仅有一个'@'符号
2)'@'和'.'不能出现在字符串的首和尾
3)'@'之后至少要有一个'.'，并且'@'不能和'.'直接相连
满足以上3条的字符串为合法邮箱，否则不合法，
编写程序验证输入是否合法

**输入**

输入包含若干行，每一行为一个代验证的邮箱地址，长度小于100

**输出**

每一行输入对应一行输出
如果验证合法，输出 YES
如果验证非法：输出 NO

样例输入

```
.a@b.com
pku@edu.cn
cs101@gmail.com
cs101@gmail
```

样例输出

```
NO
YES
YES
NO
```

【刘思瑞，元培学院物理方向，2023年秋】

思路：

利用正则表达式，先把所有违规的都排除，再验证@.+\.可以被search到即可

**Python3 代码**

```
'''
刘思瑞 2100017810
'''
import re
while True:
    try:
        address = input()
        if not address:
            break

        if re.search('@.*@',address) or re.match('\s*[@.]',address) or re.search('[@.]$',address) or re.search('@\.',address) or re.search('\.@',address):
            print('NO')
        elif re.search('@.+\.',address):
            print('YES')
        else:
            print('NO')
    except EOFError:
        break
```

Python代码运行截图

状态：Accepted

源代码

```
'''
刘思瑞 2100017810
'''
import re
while True:
    try:
        address = input()
        if not address:
            break

        if re.search('@.*@',address) or re.match('\s*[@.]',address) or re.
            print('NO')
        elif re.search('@.+\.',address):
            print('YES')
        else:
            print('NO')
    except EOFError:
        break
```

# 2. 选做题目

## OJ01008: Maya Calendar

math, http://cs101.openjudge.cn/practice/01008/

During his last sabbatical, professor M. A. Ya made a surprising discovery about the old Maya calendar. From an old knotted message, professor discovered that the Maya civilization used a 365 day long year, called Haab, which had 19 months. Each of the first 18 months was 20 days long, and the names of the months were pop, no, zip, zotz, tzec, xul, yoxkin, mol, chen, yax, zac, ceh, mac, kankin, muan, pax, koyab, cumhu. Instead of having names, the days of the months were denoted by numbers starting from 0 to 19. The last month of Haab was called uayet and had 5 days denoted by numbers 0, 1, 2, 3, 4. The Maya believed that this month was unlucky, the court of justice was not in session, the trade stopped, people did not even sweep the floor. For religious purposes, the Maya used another calendar in which the year was called Tzolkin (holly year). The year was divided into thirteen periods, each 20 days long. Each day was denoted by a pair consisting of a number and the name of the day. They used 20 names: imix, ik, akbal, kan, chicchan, cimi, manik, lamat, muluk, ok, chuen, eb, ben, ix, mem, cib, caban, eznab, canac, ahau and 13 numbers; both in cycles. Notice that each day has an unambiguous description. For example, at the beginning of the year the days were described as follows: 1 imix, 2 ik, 3 akbal, 4 kan, 5 chicchan, 6 cimi, 7 manik, 8 lamat, 9 muluk, 10 ok, 11 chuen, 12 eb, 13 ben, 1 ix, 2 mem, 3 cib, 4 caban, 5 eznab, 6 canac, 7 ahau, and again in the next period 8 imix, 9 ik, 10 akbal . . . Years (both Haab and Tzolkin) were denoted by numbers 0, 1, . . . , where the number 0 was the beginning of the world. Thus, the first day was: Haab: 0. pop 0 Tzolkin: 1 imix 0 Help professor M. A. Ya and write a program for him to convert the dates from the Haab calendar to the Tzolkin calendar.

**输入**

The date in Haab is given in the following format:
NumberOfTheDay. Month Year

The first line of the input file contains the number of the input dates in the file. The next n lines contain n dates in the Haab calendar format, each in separate line. The year is smaller then 5000.

**输出**

The date in Tzolkin should be in the following format:
Number NameOfTheDay Year

The first line of the output file contains the number of the output dates. In the next n lines, there are dates in the Tzolkin calendar format, in the order corresponding to the input dates.

样例输入

```
3
10. zac 0
0. pop 0
10. zac 1995
```

样例输出

```
3
3 chuen 0
1 imix 0
9 cimi 2801
```

来源

Central Europe 1995

【刘思瑞，元培学院物理方向，2023年秋】

思路：

这道题最烦的就是索引有点乱七八糟，所以干脆全和数组统一然后遇到1开头再加1就好。

**Python3 代码**

```
'''
刘思瑞 2100017810
'''
def transfer(day, month, year):
    y = int(year)
    d = int(day[:-1])
    m = Haab_month.index(month)
    y_t = (y * 365 + m * 20 + d )//260
    rest = (y * 365 + m * 20 + d )%260
    d_t = rest % 20
    n_t = rest % 13 + 1
    return Tzolian_day[d_t] , n_t , y_t

```

```
14 Haab_month = ['pop', 'no', 'zip', 'zotz', 'tzec', 'xul', 'yoxkin', 'mol',
   'chen', 'yax', 'zac', 'ceh', 'mac', 'kankin', 'muan', 'pax', 'koyab',
   'cumhu', 'uayet' ]
15 Tzolian_day = ['imix', 'ik', 'akbal', 'kan', 'chicchan', 'cimi', 'manik',
   'lamat', 'muluk', 'ok', 'chuen', 'eb', 'ben', 'ix', 'mem', 'cib', 'caban',
   'eznab', 'canac', 'ahau']
16 n = int(input())
17 print(n)
18 for i in range(n):
19     day , month , year = input().split()
20     day_t , num_t , year_t = transfer(day , month , year)
21     print(num_t,day_t,year_t)
```

Python代码运行截图

### #41304467提交状态

## 状态: Accepted

源代码

```
'''
刘思瑞 2100017810
'''
def transfer(day, month, year):
    y = int(year)
    d = int(day[:-1])
    m = Haab_month.index(month)
    y_t = (y * 365 + m * 20 + d )//260
    rest = (y * 365 + m * 20 + d )%260
    d_t = rest % 20
    n_t = rest % 13 + 1
    return Tzolian_day[d_t] , n_t , y_t

Haab_month = ['pop', 'no', 'zip', 'zotz', 'tzec', 'xul', 'yoxkin', 'mol', 'chen'
Tzolian_day = ['imix', 'ik', 'akbal', 'kan', 'chicchan', 'cimi', 'manik', 'lamat
n = int(input())
print(n)
for i in range(n):
    day , month , year = input().split()
    day_t , num_t , year_t = transfer(day , month , year)
    print(num_t,day_t,year_t)
```

# 3. 学习总结和收获

这次的题目里面有相对比较简单的，比如前三道题，可以帮助熟悉一下语法，对于chat room 以及邮箱验证，其实理论上写起来是会有一点点麻烦的，但是我发现对这类题目直接上regex就可以全部解决，虽然感觉正则表达式的复杂度不一定有直接写低，但是写正则表达式代码会非常短。总体来讲包括maya日历都差不多一次ac了

## 额外练习

OJ01833：排列

这道题也是字典序，思路上就是从后面开始看，找到第一个不是升序的数字然后再找到最小的比这个数字大的数字，把它pop掉，剩下的直接排列就好，最后迭代k次就可以了。主要从这题学会了不换行的输出还有利用切片进行数组反转的快捷写法，感觉收获挺多。

---

状态：Accepted

源代码

```
'''
刘思瑞 2100017810
'''
def next(a):
    a = a[::-1]
    for i in range(len(a)-1):
        if a[i+1] < a[i]:
            minum = a[i]
            index = i
            for j in range(i+1):
                if a[j] > a[i+1] and a[j] < minum:
                    minum = a[j]
                    index = j
            b = a[:i+2]
            del b[index]
            b.sort(reverse = True)
            c = b + [minum] + a[i+2:]
            return c[::-1]
    return a

num = int(input())
for i in range(num):
    n , k = map(int, input().split())
    a = list(map(int, input().split()))
    for j in range(k):
        a = next(a)
    for j in range(n):
        print(a[j],end = ' ')
    print()
```

OJ01961:前缀中的周期

一开始我试图利用正则表达式，但是正则表达式的内存直接超了，因此我直接用切片来写了，其中我还加入了从第一个和首字母相同的开始遍历，每次找到重复序列就从末尾开始遍历，但是一直出错，我一开始是想过周期重叠的问题的，但是当时忘了考虑最特殊的两次重复造成的重叠，最后还是看了闫老师给的测试数据才发现问题，对两次重复分类讨论，最终过了最后的数据点。

感觉我的算法有点笨，希望可以看看其它同学ac的代码学习一下。

## 状态: Accepted

源代码

```
def find(length , s , round):
    print('Test case #'+str(round))
    i = length//2 +5
    for k in range(1,length//2+1):
        if s[k] == s[0]:
            i = k-1
            break
    while i <= length//2+1:
        j =1
        while True:
            if s[:i+1] != s[j*(i+1):(j+1)*(i+1)]:
                break
            j += 1
            print(j*(i+1),j)
        if j != 2:
            i = j*(i+1)
        else:
            i += 1
    print()

round = 1
while True:
    length = int(input())
    if length == 0:
        break
    find(length , input() , round)
    round += 1

print()
```