A 指针、指针、还是指针

题目背景

铃兰在学习 C 语言，在诸多好伙伴的帮助下，她迅速掌握了条件、循环、数组、字符串等知识点，但是唯独搞不懂指针。现在博士给铃兰出了一套指针相关的选择题，但是她一道也不会做，希望求助于你，让你利用丰富的程序设计知识帮帮她。

特殊说明

本题目将给出数道单选题，采用 $SpecialJudge$ 进行评测。具体评测要求如下：

- 上机考试期间能且仅允许进行**一次提交**，不允许更改。请各位同学仔细斟酌，认真思考后给出自己的答案。
- 上机考试期间的提交将统一记为 $Accepted$ ，不进行正确性判断。
- 考试结束后将更改评测程序，统一重测，对各题的正确性逐一进行判断，按正确的题目数给分。
- 请提交一个输入**阿拉伯数字题号（从1开始）**，输出对应选项答案的程序。

比如总共五道题，你的答案分别是ACCDB，那么请提交一份类似这样的代码：

```
#include <stdio.h>
char ans[10] = "ACCDB";
int main(){
  int x;
  scanf("%d", &x);
  printf("%c",ans[x-1]);
  return 0;
}
```

题目内容

总共五道选择题，均为单选题，请选择四个选项中最符合题目要求的一项，按照前述要求输出。

第一题

下述代码中，能正确进行字符串赋值，并能确保字符串以'\0'结尾的是？

```
(A) char s[5] = {"ABCDE"};
(B) char s[5] = {'A', 'B', 'C', 'D', 'E'};
(C) char *s; s = "ABCDE";
(D) char *s; scanf("%s", s);
```

第二题

下面给出一段输出给定字符串 a 的第 i 个字符（i从0开始）的函数，请问空位应当填什么？

```
void print_ith([1]_____, int i){
  printf("%c", [2]_____);
}
```

选项如下：

```
(A)   [1] char a   [2] *(a+i)
(B)   [1] char *a  [2] a+i
(C)   [1] char a   [2] a+i
(D)   [1] char* a  [2] *(a+i)
```

第三题

给出下列程序的执行结果：

```
#include<stdio.h>
void sub(int x,int y,int *z){
    *z=y-x;
}
int main(){
    int a,b,c;
    sub(10,5,&a);
    sub(7,a,&b);
    sub(a,b,&c);
    printf("%d,%d,%d\n",a,b,c);
}


(A)  -5, -12, -7
(B)  5, -2, -7
(C)  -5, -7, -2
(D)  -12, -12, 0
```

第四题

有如下代码实现两个变量 x, y 的值交换。

```
void swap(int *x, int *y){
  int tmp = *x;
  *x = *y;
  *y = tmp;
}
```

请问应当如何调用 $swap$ 函数？（假设定义了两个 int 型变量 a, b）

```
(A)  swap(&a, &b)
(B)  swap(*a, *b)
(C)  swap(*&a, *&b)
(D)  swap(a,b)
```

第五题

设有如下定义，则四个选项中结果数值不为 3 的表达式是？

```
char x[8] = {'B', 'U', 'A', 'A', '7', '0', 't', 'h'}, *p1;

(A)  (int)(*(x + 1) - *(x + 3) + 1) / 7
(B)  (x[6] - x[7]) / 4
(C)  p1 = x + 1, 'D'-*(++p1)
(D)  p1 = x + 1, 'D'-*(p1++)
```

HINT

'A', 'B', 'U', 'h', 't'对应的ASCII码分别为0x41, 0x42, 0x55, 0x68, 0x74。 逗号在C语言中被认为是表达式的运算符，具有最低优先级，运算规则是直接返回逗号右侧的子表达式的值。比如定义有int变量x的情况下，表达式 "x = 1, x + 1" 的值是 2 . （如果你听不懂也没关系，只要把第五题里面的逗号当成分号，计算逗号后表达式的值即可）

$Arthor : Untitled$

B 字符串猫猫

题目描述

$\mathcal{R}ed$ 家的苏茜猫猫又来啦。这次小猫咪将一串字符抓破拆成了很多小块。你可以按顺序将它们拼回原形，并告诉 $\mathcal{R}ed$ 原字符串 $[L, R]$ 处（下标从 1 开始）的字符是什么吗。

输入格式

第一行两个整数 L, R，表示询问的区间。

接下来不超过 100 行，每行一个字符串（长度不超过 10），表示小猫咪拆得的字符串。（按顺序拼接起来就是原字符串）

输出格式

一行一个字符，表示原字符串 $[L, R]$ 处的字符。

样例输入

```
3 6
Miao
MiaoWu
```

样例输出

```
aoMi
```

Hint

可以使用字符串猫猫 `strcat()` 函数拼接字符串。

$Aurthor: \mathcal{R}ed$

C 大小端转化

题目描述

在计算机中，存储的基本单位为**字节**，即8个2进制bit位。

因此，在我们常见的数据类型中：

- `char`型占据1个字节
- `short`型占据2个字节
- `int`与`unsigned int`占据4个字节
- `long long`与`unsigned long long`占据8个字节

在计算机的存储中，有两种存储方式，即：

- **小端存储**：即数据的低字节存储在低地址中，高字节存储在高地址中。
- **大端存储**：即数据的高字节存储在低地址中，低字节存储在高地址中。

在Windows系统中，采用的存储模式均为**小端存储**。

现在，给你两个`unsigned int`型的非负整数，请输出它在**更换存储模式后所表示的十进制数**的乘积，并对10000取模。

例如，加入有一个数`x = 305419896 = 0x12345678`，在小端存储下，从低地址到高地址分别存储为`0x78`，`0x56`，`0x34`，`0x12`。在更换存储模式后，低地址存储的为数字的高字节，因此所表示的数为`0x78563412 = 2018915346`。

下图为大小端存储的图示：

输入

两个`unsigned int`型非负整数，用一个空格隔开

输出

一个非负整数，即转化存储模式后的乘积

输入样例

```
12345 56789
```

输出样例

```
6560
```

样例解释

$12345 = 0x3039, 56789 = 0xDDD5$，转化存储模式后分别为$0x39300000$和$0xD5DD0000$。转化为十进制为$959447040$和$3588030464$，相乘后对$10000$取模得到$6560$。

数据范围

两个乘数均在`unsigned int`范围内。

HINT

此题的本质为将一个数字按字节颠倒。

- 首先，采用之前的位运算方式肯定是行得通的。
- 其次，可以采用指针的方式，如果用一个`char`型的指针`p`指向一个`unsigned int`型的数，则这个指针提取到的是这个数字的**存储在计算机最低位**的字节，相应的，`p`，`p+1`，`p+2`，`p+3`分别为这个数在计算机中存储的从低到高四个字节。

Author: Olivaw

D 矩阵的幂次 mini

题目背景

Yakumo Ran 正在给 Cirno 讲解矩阵乘法。由于 Cirno 不会快速幂也不会高精度，所以 Ran 将习题的难度降低了许多。

题目描述

给定两个 $m \times p$ 的矩阵 A, B 和一个整数 n，试求 $(AB^T)^n$。（B^T 表示 B 的转置。）

输入

共 $2m + 1$ 行。

第一行三个整数 m, p, n，含义见上。

接下来 m 行，每行 p 个整数，第 i 行 j 列的整数表示 A_ij。

最后 m 行，每行 p 个整数，第 i 行 j 列的整数表示 B_ij。

输出

共 m 行，每行 m 个整数，以一个空格分隔。

记答案为矩阵 C，第 i 行 j 列的整数表示 C_ij。

样例

输入

```
3 2 2
-4 -4
5 -4
3 2
5 -3
-1 -2
5 0
```

输出

```
328 80 160
40 278 -290
-196 -18 -130
```

数据范围

$1 \leq m, p, n \leq 10$，保证结果在 `int` 范围内。

HINT

矩阵的 n 次幂就是 n 个这样的矩阵相乘，如 A^2 就是 $A \times A$。

E 成绩分析!

题目描述

期中考试结束了，Uanu想让你对各班级的成绩进行分析，你需要求出**各班**的成绩最高分、最低分、平均分和中位数。

中位数：排序后居于中间位置的成绩，总人数为偶数时，中位数为中间两人成绩的平均值。

输入格式

不定行输入。

每行多个整数，空格隔开，表示该班每个人成绩，-1为每行结束标志，不计入。

输出格式

每行输出四个数，空格隔开，依次表示对应班的最高分、最低分、平均分和中位数。平均分和中位数均保留2位小数，其他均为整数。

输入样例

```
1 2 3 -1
1 10 2 3 -1
```

输出样例

```
3 1 2.00 2.00
10 1 4.00 2.50
```

HINT

当需要向外部传递多个数值时，可以使用**指针类型的函数参数。**

数据范围

班级总数不超过50。

每班人数不超过500人且不为0。

成绩为不超过100的正整数。

author: $\mathfrak{Uanu}$

F 横幅设计

题目描述

校庆啦~~小羊学姐在帮忙布置校庆现场，她负责某会场的横幅制作。

小羊学姐先电脑上设计横幅，已知横幅上**有且仅有一个字符串**，且该字符串只由**数字和大小写英文字母**组成。但是，突然有病毒入侵电脑，当她清除完病毒之后，再次打开横幅设计稿时，发现横幅上的字符串出现了问题，部分子串的顺序完全颠倒了。小羊学姐很着急，想请你帮忙：

她给你两个字符串：Mum 和 Son，其中 Mum 是横幅上待处理的错误母串，Son 是母串中顺序颠倒的子串。你需要将Mum 中**所有**与Son 匹配的子串都**逆置**后，得到 Mum' 输出，交给小羊学姐一个正确母串，来协助完成横幅设计。

Hint

1. Mum 中所有与 Son 匹配的子串互不相交；
2. 逆置后，不会出现“逆置部分”与“其他附近的字符”构成新的 Son 子串的情况；
3. 注意：可能会有回文串逆置后不变的情况，这并不算新的子串，因此在成功匹配到符合条件 的子串后，应该将指针后移继续匹配；
4. $strstr$函数:

   `char *p = strstr(haystack, needle);`

   在字符串 $haystack$ 中找到第一次出现 $needle$ 子串的位置，返回指向该位置的指针，如果未找到则返回 `NULL`。

输入格式

共输入一行，两个字符串 Mum 和 Son，用空格隔开，且字符串的长度满足 $1 \leq SonLength \leq MumLength \leq 200$。

输出格式

共输出一行，经过子串逆置变化后的新字符串Mum'。

样例输入1

```
HelloWorld123 123
```

样例输出1

```
HelloWorld321
```

样例输入2

```
aaaa aa
```

样例输出2

```
aaaa
```

样例输入3

```
ababab ab
```

样例输出3

```
bababa
```

G 二分查找 AC版

题目描述

眼镜决定给大家整点有用的东西：
他给出一个单调不减的数列，请你在其中查询某个数 key 的是否存在：

1. 如果这个数**不存在**，请你输出 `-1`
2. 如果这个数存在，请你输出它在数列中的**最左端位置**和**最右端的位置**，**用空格分开。如果****两个位置相同，即该数仅仅出现一次，**只需要输出它的位置即可**。位置从 0 开始标号。

输入

第一行为数据组数 `n` 和查询次数 `t`，以空格分开

第二行有 `n` 个数，为给出的单调不减数列，以空格分开

最后输入 t 行，每行 1 个数，为每次需要查询的数 key。

输出

对于每次查询，输出一行为查询结果

输入样例

```
5 3
1 2 2 3 4
1
0
2
```

输出样例

```
0
-1
1 2
```

样例解释

`1` 出现的最左端和最右端位置为 `0`

`0` 没有出现在数列中

`2` 出现的最左端位置为 `1`，最右端位置为`2`

数据范围

$1 \leq n \leq 500000$

$1 \leq t \leq 3000$

数列各项元素都在 int 范围内。

后记

这道题肯定得用二分法，如何写好判断条件来实现对应边界的查找呢?

如果你想用一次二分查到 key ，然后逐个向左或向右历遍，请你估计一下在本题数据范围下你需要计算的最大可能次数。

——致TLE

Author: Op zb

H 四季映姬的名单查找

题目描述

在做题之前四季映姬大人希望你还记得ppt上的例题7-4

两种方式的对比

/*指针版*/

```
#include<stdio.h>
#include<string.h>
#define MAX 100
int main()
{
   char arr_1[MAX], arr_2[MAX] = "";
   char *in_line = arr_1, *longest = arr_2, *tmp;
   int max_len = 0, len;
   while(gets(in_line) != NULL)
   {
      len = strlen(in_line);
      if(len > max_len)
      {
         max_len = len;
         tmp = in_line;
         in_line = longest;
         longest = tmp;
      }
   }
   printf("%d:%s\n",max_len, longest);
}
```

/*数组版*/

```
#include <stdio.h>
#define MAXLINE 1024
int str_len(char s[ ]);
void str_copy(char s[ ], char t[ ]);

int main() // find longest line
{
    int len, max;  // 当前行，目前最长行的长度
    char line[MAXLINE]; // 保存当前输入行 的内容
    char save[MAXLINE]; // 保存最长行的内容
    max = 0;
    while( gets(line) != NULL )
    {
        len = str_len(line);
        if( len > max )
        {
            max = len;
            str_copy(save, line);
        }
    }
    if( max > 0)
        printf("%d: %s", max, save);
    return 0;
}
```

与数组实现方式相比，指针实现方式减少了每当发现新的更长行时所进行的字符数组拷贝(通过调用函数str_copy)。显然指针实现方式代码执行速度要快。

等待审判的灵魂是在是太太太太太太太太多啦！即使是四季映姬这样认真且较真的阎魔大人也难以承受如此大的工作量。

因此，四季映姬决定优先审判一些特定的灵魂，以提高工作效率。

现在给出等待审判的灵魂的名单，请你找出其中名字最长的，名字次长的，名字最短的灵魂，并给出名单上灵魂的总数量。

输入格式

不定行输入

每一行为一个字符串 $name\_i$ ，为名单上第 i 个灵魂的名字。$name_i$ 由大小写字母组成且不为空字符串。

输出格式

输出共四行

第一行输出一个正整数 n ，为灵魂的数量。

第二，三，四行为分别输出一个字符串，为名字最长的，名字次长的，名字最短的三个灵魂的名字。

如果存在长度相等的情况，优先输出在名单上顺序靠前的，具体可以看样例

样例

样例输入

```
HakureiReimu
KirisameMarisa
HiedaNoAkyuu
IzayoiSakuya
SatsukiRin
```

样例输出

```
5
KirisameMarisa
HakureiReimu
SatsukiRin
```

样例解释

名单上共有五个灵魂，其名字中 `KirisameMarisa` 最长， `SatsukiRin` 最短，其余三个长度相等，都是次长，则输出名单上最靠前的 `HakureiReimu`

数据范围

$name_i$ 不为空串且长度不超过 2000

名单上的名字数量不少于 3 个且不超过 10000 个

保证最短的名字的长度小于次长的名字的长度

本题限制了内存大小，大概是不够你把所有字符串装起来的，小心 `MLE`

我也不想整这么难，但是出题要求如是要求，我能有什么办法呢ヘ(;´Д｀)ヘ

Author:星辰的微光

I 这是第几个子串?

题目描述

给出一个**仅由小写字母组成**的字符串 str ，其长度为 len ，定义 str 中从第 i 个字符到第 j 个字符（从 0 开始计， $0\le i\le j< len$ ）的**连续字符序列**为 str 的子串 $[i,j]$ ，显然 str 应该有 $\dfrac{len(len+1)}{2}$ 个子串。

将 str 的所有子串**按照字典序从小到大排列**。现给出两个整数 $i, j\ (0\le i\le j< len)$ ，求 str 的子串 $[i,j]$ 在上述排列中的位置。由于 str 中可能不止 1 个子串与子串 $[i,j]$ 相同，因此请输出 str 的所有子串按字典序升序排列中第一个和最后一个与子串 $[i,j]$ 相同的子串的位置（**下标从 1 开始计**）。

请仔细看样例解释。

字典序：两个字符串 a, b 的字典序大小，一般情况下等价于二者**第一个不相同的字符的大小**；特别地，若 a 的前 $len(b)$ 个字符恰好为字符串 b ，则 a 的字典序大于 b 。如：`a` < `b` , `az` < `b` , `abcz` < `abd` , `abc` < `abca` , `abac` = `abac` 。

输入

输入共两行

第一行一个**仅由小写字母组成**的字符串，表示字符串 str

第二行两个用空格隔开的整数 i, j ，表示 str 的子串 $[i, j]$

输出

输出两个整数，用一个空格隔开，表示 str 的子串 $[i, j]$ 在 str 所有子串字典序升序排列中**第一次出现和最后一次出现的位置**（下标从 1 开始计）

样例

输入#1

```
ababa
0 2
```

输出#1

```
6 7
```

样例#1解释

str 为 `ababa`， 它的所有子串按字典序升序排列为： `a`, `a`, `a`, `ab`, `ab`, `aba`, `aba`, `abab`, `ababa`, `b`, `b`, `ba`, `ba`, `bab`, `baba`

str 的子串 $[0,2]$ 为 `aba`，在上方排列中的第 6 个到第 7 个。

输入#2

```
cddd
1 1
```

输出#2

```
5 7
```

样例#2解释

str 为 `cddd`， 它的所有子串按字典序升序排列为： `c`, `cd`, `cdd`, `cddd`, `d`, `d`, `d`, `dd`, `dd`, `ddd`

str 的子串 $[1,1]$ 为 `d`，在上方排列中的第 5 个到第 7 个。

数据范围

str 仅由小写字母组成

$0 < len < 10000$

$0 \leq i \leq j \leq n$

Hint

可能用到的头文件 $string.h$ 中的库函数（其实这些函数都可以根据今天学的内容自己写）：

$strlen$ 函数原型为 `unsigned int strlen(const char *str);`，它接收一个指向字符串的指针 str，返回字符串 str 的长度。

$strcmp$ 函数原型为 `int strcmp(const char *str1, const char *str2);`，把 $str1$ 所指向的字符串和 $str2$ 所指向的字符串进行比较：若按字典序 $str1 > str2$ 的返回正值，若 $str1 < str2$ 则返回负值，若 $str1 = str2$ 则返回 0 .

$strcpy$ 函数原型为 `char *strcpy(char *dest, const char *src);` 把 src 所指向的字符串复制到 $dest$，返回值为 $dest$ 。

关于解题的提示：

str 的子串 $[i,j]$ 随 j 增加，字典序严格增大。也许可能需要自定义一个函数实现某种功能。

Not Hint

其实这道题也有 $O(n)$ 的做法

Author: 哪吒

J void学计组

题目背景

作为一个6系的学生，void正在学计算机组成

计组中，需要编写MIPS汇编指令来对自己的CPU进行测试

现在void找到了你，他想请你帮他编写一个程序，模拟mips指令

补充知识

这些指令的操作对象只包括寄存器

寄存器一共有32个，用`$0~$31`表示，也可以称呼x号寄存器

其中，零号寄存器有一个特殊的性质，无论对它进行怎样的赋值，它的值都始终为0，不会改变

其余的寄存器均可以自由的赋值

在程序一开始的时候，所有寄存器的值均为0

---

void想请你实现的MIPS指令集包含8条指令，它们的格式是这样的：

add指令

```
add $1,$2,$3
#井号表示注释，相当于C中的//
#上面这句话的意思是，将2号寄存器和3号寄存器的值相加，存入1号寄存器
```

sub指令

```
sub $1,$2,$3
#将2号寄存器的值减去3号寄存器的值，存入1号寄存器
```

ori指令

```
ori $1,$2,2106
#将2号寄存器的值或上2106，存入1号寄存器
```

li指令

```
li $1,2106
#将2106加载到1号寄存器，现在$1==2106，也就是0x0000083a
```

lui指令

```
lui $1,2106
#将2106加载到1号寄存器的高16位，低16位赋值为0
#2106的四位十六进制表示为0x083a
#那么用十六进制来表示$1为：0x083a0000
```

beq指令（与mips稍有区别，不使用label）

```
beq $0,$1,3#如果$0和$1相等，则程序从下一条指令开始数三条指令再开始执行
nop
nop
nop
li $1,10#如果$0和$1相等的话，就从这一条开始执行
nop
```

syscall指令

```
#本题中的syscall仅仅用于结束程序
#当$2的值为10的时候，执行syscall，这个mips程序就会结束，可以理解为return 0;
#其他情况下均视为不起作用，相当于nop
li $1,20
li $2,10
syscall #程序在这里结束
li $3,5 #这一条不会执行
```

nop指令

```
nop
#这条指令没有任何作用，直接执行下一条就好
```

输入格式

输入共$n+1$行

第一行一个整数n，表示读入指令的行数

后续n行，每行一条指令

输出格式

输出有三种情况

1. 程序能正常结束（使用syscall），则输出一行`program is finished running`，然后每行4个，以空格分割，以十六进制输出32个寄存器的值（使用小写字母，补零至八位）
2. 程序运行到最后一条指令，如果没有正常使用syscall结束，则输出一行`program is finished running(dropped off bottom)`，然后和第一条一样，输出32个寄存器的值
3. 程序无法结束，一直在运行，执行的指令条数超过了$1e6$条，则输出一行`program can't be finished`

样例

样例输入

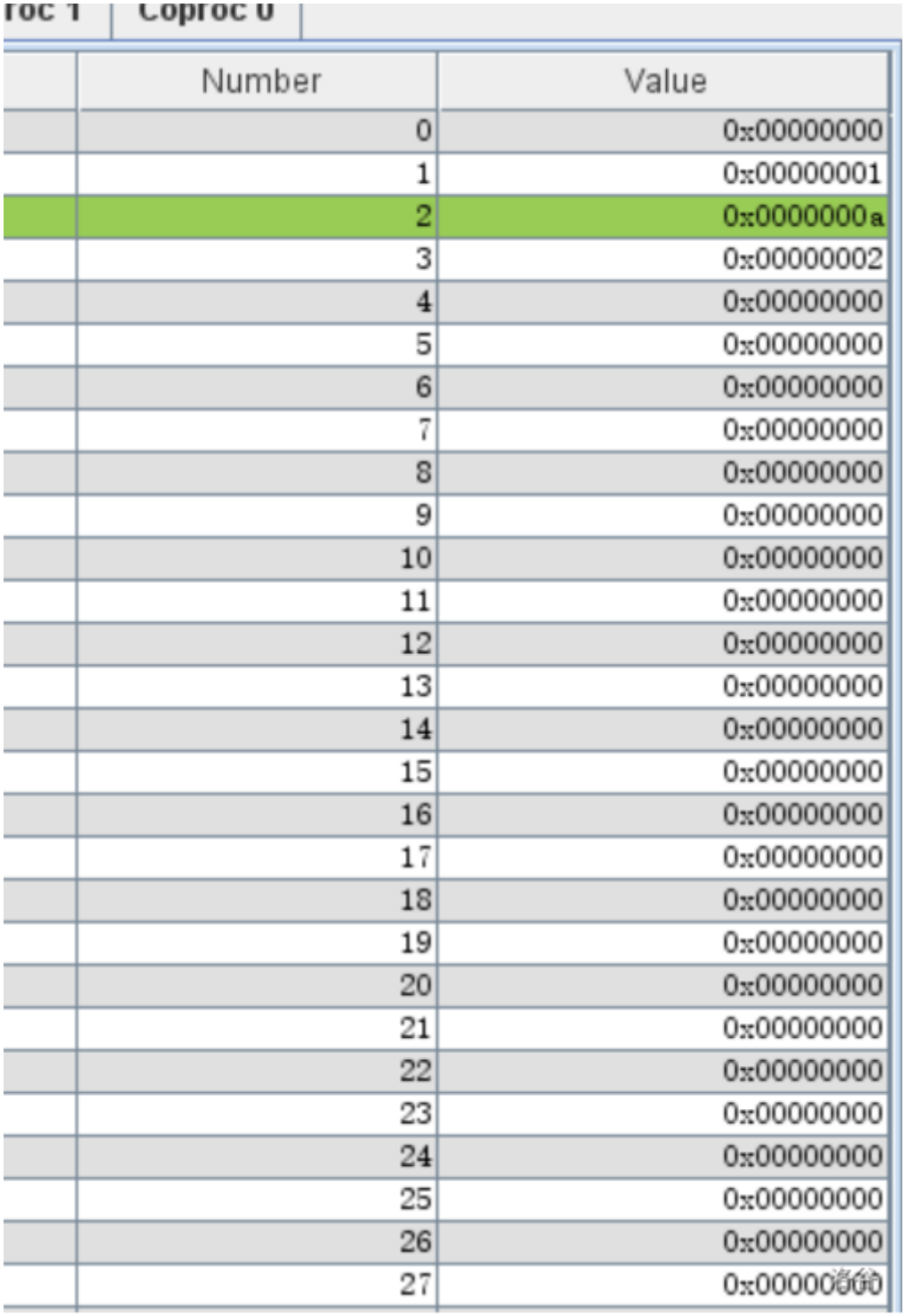
```
6
li $1,1
li $2,1
add $0,$1,$2
add $3,$1,$2
li $2,10
syscall
```

样例输出

```
program is finished running
00000000 00000001 0000000a 00000002
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
```

样例解释

给出部分寄存器的值作为参考



roc 1 | Coproc 0

	Number	Value
	0	0x00000000
	1	0x00000001
	2	0x0000000a
	3	0x00000002
	4	0x00000000
	5	0x00000000
	6	0x00000000
	7	0x00000000
	8	0x00000000
	9	0x00000000
	10	0x00000000
	11	0x00000000
	12	0x00000000
	13	0x00000000
	14	0x00000000
	15	0x00000000
	16	0x00000000
	17	0x00000000
	18	0x00000000
	19	0x00000000
	20	0x00000000
	21	0x00000000
	22	0x00000000
	23	0x00000000
	24	0x00000000
	25	0x00000000
	26	0x00000000
	27	0x00000000

数据范围

$1 \leq n \leq 127$

$1 \leq strlen(s) \leq 200$（s为每一行的字符串）

li,lui,ori指令中的数字，$0 \leq x \leq 65535$

beq指令跳转的范围在程序内，无需考虑非法情况

除了beq指令之外不会涉及负数

输入的所有字母均为小写

寄存器编号为`$0~$31`

不会有多余的空格输入，指令名称之后有一个空格，寄存器和数字均以英文逗号隔开