## A: 大整数乘法

**描述**

计算两个大整数的积。

**输入**

第1行是一个整数n，表示随后有n组数据。每组数据占一行，包含两个整数a，b，用空格隔开。

0 <= a,b <= 10^500(10的500次方)

**输出**

对每组数据输出两个整数的积，并换行。

**样例**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 3  1 2  5 8  2 9999 | 2  40  19998 |

#include<stdio.h>

#include<math.h>

#include<string.h>

#define M 10005

char s1[M],s2[M],s[M];

int a[M],b[M],c[M];

int main()

{

int i,j,m,n,k,t;

scanf("%d",&t);

for(i=0;i<t;i++)

{

while(~scanf("%s%s",s1,s2))

{

memset(c,0,sizeof(c));

n=strlen(s1);

m=strlen(s2);

k=n+m;//保证相乘后的位数不会大于k

/\*把字符串s1和s2逆序用数字排列\*/

for(i=0; i<n; i++)

a[i]=s1[n-i-1]-'0';

for(i=0; i<m; i++)

b[i]=s2[m-1-i]-'0';

/\* 乘运算\*/

for(i=0; i<n; i++)

for(j=0; j<m; j++)

c[i+j]+=a[i]\*b[j];

for(i=0; i<=k; i++)

{

if(c[i]>=10)

{

c[i+1]+=c[i]/10;

c[i]%=10;

}

}

/\*去除前导0\*/

i=k;

while(c[i]==0) i--;

/\*判断两个非负数之积是否为0，以及逆序打印c[]\*/

if(i<0) printf("0");

else

{

for(; i>=0; i--)

printf("%d",c[i]);

}

printf("\n");

}

}

return 0;

}

## B: 字符串循环右移问题

**描述**

将一个字符串循环向右移M（M>=0）个字符位置，即将字符串A中的数据由（A0 A1……AN-1）变换为（AN-M …… AN-1 A0 A1……AN-M-1）（N是字符串的长度，最后M个数循环移至最前面的M个位置）。

**输入**

每个输入包含一个测试用例，第1行输入M（M>=0）；第2行为输入字符串，长度不超过200。

**输出**

在一行中输出循环右移M位以后的字符串。

**样例1**

**输入**                                           **输出**

2

Hello World Here I Come                        meHello World Here I Co

**提示**

可以使用scanf读入整数，但不能使用scanf读入字符串，因为字符串中可能包含空格。可以使用fgets读入字符串，但要注意处理上一行末和本行末的的回车换行符。

#include<stdio.h>

int main()

{

int m;

scanf("%d",&m);

getchar();

char str[200]={'\0'};

int len=0;

while((str[len]=getchar())!='\n')len++;

for(int i=0;i<len;i++)

{

printf("%c",str[(len-m+i)%len]);

}

return 0;

}

## C: 回文数

**描述**

程序中输入一个正整数(32位int可以存放下)，请判断他是否为回文数。

提示：回文数：类似1234321，123321等这样，顺序和逆序相同的整数称为回文数。

**输入**

每个测试输入包含1个测试用例，格式为

  第1行：正整数n

  第2行：正整数x1

  第3行：正整数x2

  ... ... ...

  第n+1行：正整数xn

**输出**

若n是回文数，输出Yes，否则输出No。

**样例1**

**输入**             **输出**

3

1231             No

12321            Yes

12345            No

#include<stdio.h>

int main(void)

{

int n,i;

scanf("%d",&n);

getchar();

for(i=0;i<n;i++)

{

int t,m,k;

scanf("%d",&t);

k=t;

m=0;

while(k>0)

{

m=m\*10+k%10;

k=k/10;

}

if(m==t)

{

printf("Yes\n");

}

else

{

printf("No\n");

}

}return 0;

}

## D: 电子表格

**描述**

有一个r行c列(1<=r,c<=50)的电子表格，行从上到下编号为1~r，列从左到右编号是1~c。上面定义了几种操作：(x, y等都是整数)

l  SR x1 x2  
交换x1行和x2行

l  SC y1 y2  
交换y1列和y2列

l  DR x  
删除x行

l  DC y  
删除y列

l  IR x  
在x行上面插入一行，新插入的行的所有单元格值为0

l  IC y  
在y列前面插入一列，新插入的列的所有单元格值为0

**输入**

第一行是2个整数r, c，表示电子表格有r行c列。接下来的r行，每行c个整数，表示每个单元格的值。再接下来的一行有1个整数n，表示随后有n行，每行是一个操作命令。

**输出**

输出经过这n次操作后的电子表格内容。

**样例输入**

3 5

1 2 3 4 5

3 4 5 6 7

5 6 7 8 9

4

SR 1 3

SC 2 4

IR 2

DC 2

**样例输出**

5 7 6 9

0 0 0 0

3 5 4 7

1 3 2 5

#include<iostream>

#include<string>

#include<string.h>

using namespace std;

int main()

{

int r,c,i,j,p;cin>>r>>c;

int table[r+1][c+1];

for(i=0;i<r;i++)//输入

{

for(j=0;j<c;j++)

{

cin>>table[i][j];getchar();

}

}

int n,x1,x2,y1,y2,x,y;//行 列

// const char commmand[6][3]={"SR","SC","DR","DC","IR","IC"};

char temp[1][10];//存命令

cin>>n;

for(i=0;i<n;i++)

{

scanf("%s",temp[0]);

// cout<<"temp="<<temp[0]<<endl;

//交换x1行和x2行

if(strcmp(temp[0],"SR")==0)

{

cin>>x1>>x2;

int temp1[1][c];

memcpy(temp1[0],table[x1-1],sizeof(int)\*1\*c);

memcpy(table[x1-1],table[x2-1],sizeof(int)\*1\*c);

memcpy(table[x2-1],temp1[0],sizeof(int)\*1\*c);

}

//交换y1列和y2列

else if(strcmp(temp[0],"SC")==0)

{

cin>>y1>>y2;

int temp2[1][c];

for(j=0;j<r;j++)

{

temp2[0][0] = table[j][y1-1];

table[j][y1-1]= table[j][y2-1];

table[j][y2-1] = temp2[0][0];

}

}

//删除x行

else if(strcmp(temp[0],"DR")==0)

{

cin>>x;

for(j=x-1;j<r-1;j++)

{

memcpy(table[j],table[j+1],sizeof(int)\*c);

}

r-=1;

}

//删除y列

else if(strcmp(temp[0],"DC")==0)

{

cin>>y;

for(j=0;j<r;j++)

{

for(p=y-1;p<c-1;p++)

{

table[j][p]=table[j][p+1];

}

}

c-=1;

}

//在x行上面插入一行全为0 ????????????

else if(strcmp(temp[0],"IR")==0)

{

cin>>x;r+=1;

for(j=r-1;j>x-1;j--)

{

memcpy(table[j],table[j-1],sizeof(int)\*c);

}

for(p=0;p<c;p++)

{

table[x-1][p]= 0;

}

}

//在y列前面插入一列全为0

else if(strcmp(temp[0],"IC")==0)

{

cin>>y;

for(p=0;p<r;p++)

{

for(j=c-1;j>y-1;j--)

{

table[p][j]=table[p][j-1];

}

}

c+=1;

for(p=0;p<r;p++)

{

table[y-1][p] = 0;

}

}

}

//输出结果

for(i=0;i<r;i++)

{

for(j=0;j<c;j++)

{

cout<<table[i][j]<<" ";

}

cout<<endl;

}

return 0;

}

## E: n个不同整数的所有组合

**描述**

计算n个不同整数的所有的组合

**输入**

第1行是一个整数m，表示随后有m组数据。

每组数据占一行，第一个整数n表示随后有n个不同的整数。

**输出**

对每组数据，输出这n个不同整数的不同组合，含空组合。每个组合输出前输出“--> ”。

要求必须按照上课讲的递归方法来写，先输出包含该元素的情况，再输出不包含的情况，这样才能保证输出顺序和标准答案一致。

**样例**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 2  2 1 2  3 1 2 3 | --> 1 2  --> 1  --> 2  -->  --> 1 2 3  --> 1 2  --> 1 3  --> 1  --> 2 3  --> 2  --> 3  --> |

#include <stdio.h>

#include <string.h>

int main(void)

{

void output(int array[], int status[], int x, int left);

int array[100], status[100], i, n, x, y, left;

(void) scanf("%d", &n);

for (i = 0; i < n; i++)

{

memset(array, 0, sizeof(array));

memset(status, 0, sizeof(status));

(void) scanf("%d", &x);

left = x;

for (y = 0; y < x; y++) (void) scanf("%d", &array[y]);

output(array, status, x, left);

}

return 0;

}

void output(int array[], int status[], int x, int left)

{

int count;

if (left == 0)

{

printf("-->");

for (count = 0; count < x; count++) if (status[count]) printf(" %d", array[count]);

printf("\n");

return ;

}

status[x-left] = 1;

output(array, status, x, left - 1);

status[x-left] = 0;

output(array, status, x, left - 1);

return ;

}