**1、月饼**

月饼是中国人在中秋佳节时吃的一种传统食品，不同地区有许多不同风味的月饼。现给定所有种类月饼的库存量、总售价、以及市场的最大需求量，请你计算可以获得的最大收益是多少。

注意：销售时允许取出一部分库存。样例给出的情形是这样的：假如我们有3种月饼，其库存量分别为18、15、10万吨，总售价分别为75、72、45亿元。如果市场的最大需求量只有20万吨，那么我们最大收益策略应该是卖出全部15万吨第2种月饼、以及5万吨第3种月饼，获得 72 + 45/2 = 94.5（亿元）。

**输入格式：**

每个输入包含1个测试用例。每个测试用例先给出一个不超过1000的正整数N表示月饼的种类数、以及不超过500（以万吨为单位）的正整数D表示市场最大需求量。随后一行给出N个正数表示每种月饼的库存量（以万吨为单位）；最后一行给出N个正数表示每种月饼的总售价（以亿元为单位）。数字间以空格分隔。

**输出格式：**

对每组测试用例，在一行中输出最大收益，以亿元为单位并精确到小数点后2位。

**输入样例：**

3 20

18 15 10

75 72 45

**输出样例：**

94.50

简单贪心，按照单个月饼的价格排序，然后优先拿价值高的月饼，居然还错了几发，月饼库存量和总售价有可能是小数，有可能是小数样例就别给那么规整的啊，水个小题容易么......

**#include <cstdio>**

**#include <cstring>**

**#include <iostream>**

**#include <algorithm>**

using namespace std;

const int maxn = 1000;

struct node {

double num, mon, per;

}tn[maxn + 10];

bool cmp(node x, node y) {

return x.per > y.per;

}

int main()

{

int N;

double D;

scanf("%d%lf", &N, &D);

for (int i = 0; i < N; i++) {

scanf("%lf", &tn[i].num);

}

for (int i = 0; i < N; i++) {

scanf("%lf", &tn[i].mon);

tn[i].per = tn[i].mon / tn[i].num;

}

sort(tn, tn + N, cmp);

double ans = 0;

for (int i = 0; i < N; i++) {

if (D > tn[i].num) {

D -= tn[i].num;

ans += tn[i].mon;

}

else {

ans += tn[i].per \* D;

D = 0;

break;

}

}

printf("%.2f\n", ans);

return 0;

}

**2、菱形**

题目内容：

华老师给了你们一个打印任务，让你们打印边长为n的菱形。

输入格式:

输入一行一个整数n（1<=n<=50)。

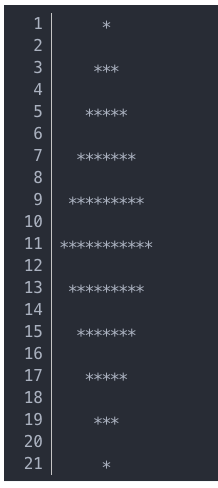
输出格式：

打印边长为N的菱形（菱形的打印形式见样例），每行均以’'结尾，每行的’’之前如果需要用空格补齐对齐，’\*’之后不要有多余的空格。

输入样例：

6

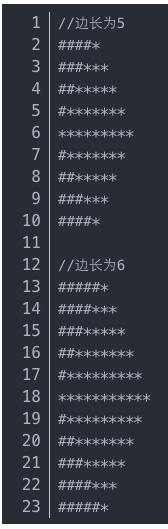
输出样例：



## 题解

题目中要求为打印**边长**为n的菱形。

下面为边长为5和边长为6的菱形，为了更加直观的看到规律，结果中空格用#代替。



在此题目中边长为n其实可以理解为从第一行到最多点行的行数，那么我们就可以根据当前行数来控制输出的\*与 （空格）。

从上面的结果中，我们发现：

第一行会先输出n-1个空格，再输出1（行数 2 \* i - 1 ）个\*；

第二行先输出 n-2 个空格，再输出3（行数 2 \* i - 1）个\*；

那么第n行则会先输出 n-n 个空格，再输出2 \* n - 1个\*。

n行之后的行与之前的输出规则与上述部分相反，即：

n+1行，先输出 1 个空格，再输出 2 \* (n-1)-1 个 \*；

n+2行，先输出 2 个空格，再输出 2 \* (n-2)-1 个 \*；

······

2n-1行（最后一行），先输出n-1个空格，再输出 2 \* (n-n+1)-1 个\*。

具体代码如下所以。

————————————————

#include<iostream>

using namespace std;

int main()

{

int i = 1, j, k, n;

//输入n

cin >> n;

//先输出菱形的上半部分

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

{

for (j = 1; j <= (n - i); j++)

{

//先输出空格

cout << " ";

}

for (k = 1; k <= 2 \* i - 1; k++)

{

//再输出\*

cout << "\*";

}

cout << '\n';

}

for (i = 1; i <= n - 1; i++)

{

for (j = 1; j <= i; j++)

{

//先输出空格

cout << ' ';

}

for (k = 1; k <= 2\*(n - i)-1; k++)

{

//再输出\*

cout << "\*";

}

cout << endl;

}

return 0;

}

# 3、数素数

令 Pi 表示第 i 个素数。现任给两个正整数 M≤N≤10^4，请输出 PM 到 PN 的所有素数。

输入格式：

输入在一行中给出 M 和 N，其间以空格分隔。

输出格式：

输出从 PM 到 PN 的所有素数，每 10 个数字占 1 行，其间以空格分隔，但行末不得有多余空格。

输入样例：

5 27

输出样例：

11 13 17 19 23 29 31 37 41 43

47 53 59 61 67 71 73 79 83 89

97 101 103

#include <stdio.h>

int main()

{

int m,n;

int i,j,f,q;

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

i=2;f=1;q=0;

while(f<=n){

for(j=2;j\*j<=i;j++)

if(i%j==0) break;

if(j\*j>i){

if(q%10>0)

printf(" ");

if(q>0&&q%10==0)

printf("\n");

if(f>=m){

printf("%d",i);

q++;

}

f++;

}

i++;

}

printf("\n");

return 0;

}

# 4、福尔摩斯的约会

大侦探福尔摩斯接到一张奇怪的字条：我们约会吧！ 3485djDkxh4hhGE 2984akDfkkkkggEdsb s&hgsfdk d&Hyscvnm。大侦探很快就明白了，字条上奇怪的乱码实际上就是约会的时间星期四 14:04，因为前面两字符串中第 1 对相同的大写英文字母（大小写有区分）是第 4 个字母 D，代表星期四；第 2 对相同的字符是 E ，那是第 5 个英文字母，代表一天里的第 14 个钟头（于是一天的 0 点到 23 点由数字 0 到 9、以及大写字母 A到 N 表示）；后面两字符串第 1 对相同的英文字母 s 出现在第 4 个位置（从 0 开始计数）上，代表第 4 分钟。现给定两对字符串，请帮助福尔摩斯解码得到约会的时间。

输入格式：

输入在 4 行中分别给出 4 个非空、不包含空格、且长度不超过 60 的字符串。

输出格式：

在一行中输出约会的时间，格式为 DAY HH:MM，其中 DAY 是某星期的 3 字符缩写，即 MON 表示星期一，TUE 表示星期二，WED 表示星期三，THU 表示星期四，FRI 表示星期五，SAT 表示星期六，SUN 表示星期日。题目输入保证每个测试存在唯一解。



//注意小时或者分钟不满足两位的要记得补零。

//再者注意，每一次判断的字母范围，第一次与第二次的范围不一样通过题意隐含得出。

#include <stdio.h>

int main(){

char a1[61],a2[61],b1[61],b2[61];

int times=0;

char Weekday[7][4]={"MON","TUE","WED","THU","FRI","SAT","SUN"};//每个星期由三个字母组成所以设置数组时长度应为4，因为还有结束符

if((scanf("%s%s%s%s",a1,a2,b1,b2))!=EOF){

for(int i=0;times<2&&i<=60;i++){

if((a1[i]==a2[i])&&(a1[i]>='A'&&a1[i]<='G')&&times==0){

times++;

int num=a1[i]-'A';

printf("%s ",Weekday[num]);

continue;

}

if((((a1[i]==a2[i])&&(a1[i]>='A'&&a1[i]<='N'))||((a1[i]==a2[i])&&(a1[i]>='0'&&a1[i]<='9')))&&times==1){

times++;

if(a1[i]>='0'&&a1[i]<='9'){

printf("0%c:",a1[i]);

}

if(a1[i]>='A'&&a1[i]<='N'){

int num=a1[i]-'A'+1;

printf("%d:",9+num);

}

break;

}

}

times=0;

for(int i=0;times<1;i++){

if(b1[i]==b2[i]&&((b1[i]>='A'&&b1[i]<='Z')||(b1[i]>='a'&&b1[i]<='z'))){

times++;

if(i<10){

printf("0%d",i);

}else{

printf("%d",i);

}

}

}

}

return 0;

}

# 5、旧键盘打字

## 题目描述

旧键盘上坏了几个键，于是在敲一段文字的时候，对应的字符就不会出现。现在给出应该输入的一段文字、以及坏掉的那些键，打出的结果文字会是怎样？

## 输入描述:

输入在2行中分别给出坏掉的那些键、以及应该输入的文字。其中对应英文字母的坏键以大写给出；每段文字是不超过10^5个字符的串。可用的字符包括字母[a-z, A-Z]、数字0-9、以及下划线“\_”（代表空格）、“,”、“.”、“-”、“+”（代表上档键）。题目保证第2行输入的文字串非空。  
注意：如果上档键坏掉了，那么大写的英文字母无法被打出。

## 输出描述:

在一行中输出能够被打出的结果文字。如果没有一个字符能被打出，则输出空行。

## 输入例子:

复制代码

7+IE.

7\_This\_is\_a\_test.

## 输出例子:

\_hs\_s\_a\_tst

具体代码如下：

#include <stdio.h>

int main() {

int shuru = 0;

char shu[128];

char numb[100001];

int same[128];

gets(shu);

gets(numb);

int brok = strlen(shu);

int chan = strlen(numb);

int i,j;

for (i = 0; i < brok; i++) {

same[shu[i]] = 1;

if (shu[i] >= 'A' && shu[i] <= 'Z') {

same[shu[i] + 32] = 1;

} else if (shu[i] >= 'a' && shu[i] <= 'z') {

same[shu[i] - 32] = 1;

}

}

for (j = 0; j < chan; j++) {

if (same[numb[j]]) {

continue;

}

if (numb[j] >= 'A' && numb[j] <= 'Z' && same['+']) {

continue;

}

putchar(numb[j]);

shuru++;

}

if (shuru == 0) {

printf("\n");

}

}

## 网上的

#include <stdio.h>

#include <ctype.h>

int main() {

char A[123] = {

0

}, ch;

while ((ch = getchar()) != '\n') {

A[ch]++;

}

while ((ch = getchar()) != '\n') {

if ((isupper(ch) && A[43]) || A[toupper(ch)]) {

continue;

}

else {

putchar(ch);

}

}

return 0;

}

# 6、螺旋三角形

方阵的主对角线之上称为“上三角”。

请你设计一个用于填充n阶方阵的上三角区域的程序。填充的规则是：使用1，2，3….的自然数列，从左上角开始，按照顺时针方向螺旋填充。

例如：当n=3时，输出：

1 2 3

6 4

5

当n=4时，输出：

1  2  3  4

9 10 5

8  6

7

当n=5时，输出：

1  2  3  4  5

12 13 14 6

11 15 7

10 8

9

程序运行时，要求用户输入整数n（3~20）

程序输出：方阵的上三角部分。

要求格式：每个数据宽度为4，右对齐。

解题；

#include<stdio.h>

#include<string.h>

#define MAXLEN 22

#define NOZERO 1

int main()

{

int arr[22][22] = { 0 };

int num = 1;

int i;

int j;

int value;

while (~scanf\_s("%d", &value, sizeof(value))) {

for (i = 0; i < MAXLEN; i++) {

arr[i][0] = NOZERO;

arr[0][i] = NOZERO;

}

arr[num][value + 1] = NOZERO;

for (i = 1, j = 0; num <= (value \* value - value) / 2 + value;) {

while (!arr[i][j + 1]) {

arr[i][++j] = num++;

}

while (!arr[i + 1][j - 1]) {

arr[++i][--j] = num++;

}

while (!arr[i - 1][j]) {

arr[--i][j] = num++;

}

}

for (i = 1; i <= value; i++) {

for (j = 1; j <= value + 1 - i; j++)

printf("%4d", arr[i][j]);

printf("\n");

}

memset(arr, 0, sizeof(arr));

num = 1;

}

return 0;

}