图形输出

题目给定一些规则，考生根据规则进行画图：

- 按照题目规律，直接输出即可；
- 构造一个二维字符数组，通过对规律的变形进行填充，然后输出整个二维数组；

一、乙类题目：

1、打印沙漏

本题要求你写个程序把给定的符号打印成沙漏的形状。例如给定17个“*”，要求按下列格式打印

```
*****
 ***
  *
 ***
*****
```

所谓“沙漏形状”，是指每行输出奇数个符号；各行符号中心对齐；相邻两行符号数差2；符号数先从大到小顺序递减到1，再从小到大顺序递增；首尾符号数相等。给定任意N个符号，不一定能正好组成一个沙漏。要求打印出的沙漏能用掉尽可能多的符号。

输入格式:

输入在一行给出1个正整数N（≤1000）和一个符号，中间以空格分隔。

输出格式:

首先打印出由给定符号组成的最大的沙漏形状，最后在一行中输出剩下没用掉的符号数。

输入样例:

```
19 *
```

输出样例:

```
*****
 ***
  *
 ***
*****
2
```

核心代码：

- [ ] 设底边的字符数为 x，总共需要输出的非空字符为：
- [ ] 由此可得：
- [ ] 向下取整，可以使用 int 类型强制转换实现；
- [ ] 又因为 x 是一个奇数，因此当 x 的计算结果为一个偶数时，还要减 1；
- [ ] sqrt 的参数必须是浮点数，因此将使用因数 2.0；

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
    int n, n_bottom;
    char c;
    scanf("%d %c", &n, &c);
    n_bottom = (int)sqrt(2.0 * (1+n)) - 1;      // 计算倒三角底边所需要的字符
数
    if(n_bottom % 2 == 0) n_bottom--;

    for(int i=n_bottom;i>=1;i-=2){
        for(int j=0;j<(n_bottom-i)/2;j++) printf(" ");
        for(int j=0;j<i;j++) printf("%c", c);
        printf("\n");
    }
    for(int i=3;i<=n_bottom;i+=2){
        for(int j=0;j<(n_bottom-i)/2;j++) printf(" ");
        for(int j=0;j<i;j++) printf("%c", c);
        printf("\n");
    }
    int sum = (n_bottom+1)*(n_bottom+1) / 2 - 1;
    printf("%d", n-sum);
    return 0;
}
```

二、甲类题目：

1、Hello World for U

Given any string of *N* (≥5) characters, you are asked to form the characters into the shape of `U`. For example, `helloworld` can be printed as:

```
h  d
e  l
l  r
lowo
```

That is, the characters must be printed in the **original order**, starting top-down from the left vertical line with n_1 characters, then left to right along the bottom line with n_2 characters, and finally bottom-up along the vertical line with n_3 characters. And more, we would like **U** to **be as squared as possible** -- that is, it must be satisfied that:

$$n_1 = n_3 = max\{k \mid k \le n_2 \text{ for all } 3 \le n_2 \le N\} \text{ with } n_1 + n_2 + n_3 - 2 = N.$$

Input Specification:

Each input file contains one test case. Each case contains one string with **no less than 5 and no more than 80** characters in a line.

The string contains no white space.

Output Specification:

For each test case, print the input string in the shape of U as specified in the description.

Sample Input:

```
helloworld!
```

Sample Output:

```
h   !
e   d
l   l
lowor
```

key Code:

- [ ] n_1 和 n_3 都包含底部拐角处的字符，且 $n_1 == n_3$；
- [ ] $n_1 <= n_2$，底部横线部分的字符个数不少于左右单侧竖线包含的字符数；
- [ ] 在满足上述条件的前提下，使 n_1、n_3 尽可能大；

```cpp
#include <iostream>
using namespace std;
int main(){
    string str;
    getline(cin, str);
    int len = str.size();

    int n1, n2, n3;
    n1 = n3 = (len + 2) / 3;
    n2 = len +2 -n1 - n3;

    char ans[40][40];
    for(int i=0;i<40;i++){
        for(int j=0;j<40;j++){
            ans[i][j] = ' ';    // 初始化，全部赋空格值
        }
```

```
    }

    int pos = 0;                // 用于标记在 str 中的位置;
    for(int i=0;i<n1;i++){
        ans[i][0] = str[pos++];
    }
    for(int i=1;i<n2;i++){
        ans[n1-1][i] = str[pos++];
    }
    for(int i=n1-2;i>=0;i--){
        ans[i][n2-1] = str[pos++];
    }

    for(int i=0;i<n1;i++){
        for(int j=0;j<n2;j++){
            printf("%c", ans[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```