实验二　　控制结构和数组

1　实验目的

（1）灵活掌握控制结构及其逻辑特点，学会逐步求精的算法设计。

（2）学习如何把逻辑结构相同的部分抽象为函数，以提高代码的可重用性，达到提高程序的可维护性的目的。

（3）学习使用数组作为函数参数的方法。

2　实验内容

2.1　打印温度柱状图

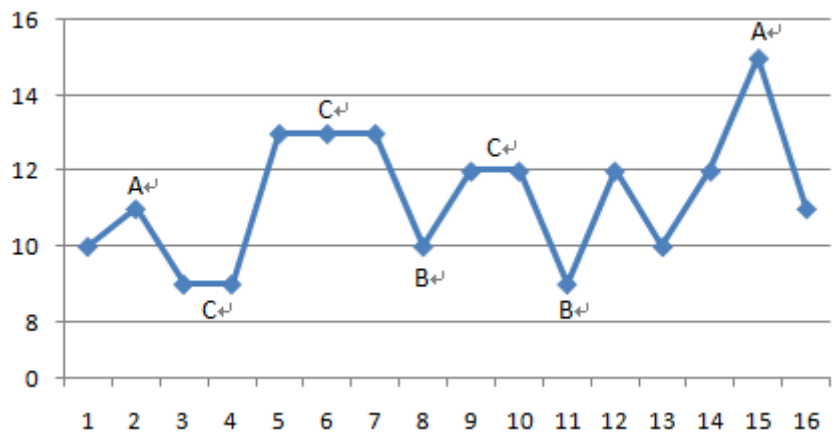
（1）问题描述

下图是某城市 15 天的气温变化曲线。其中标注为 A 的地方称为峰点，标记为 B 的地方称为谷点，而标记为 C 的地方称为崮。要求编写 1 个函数输入 15 天的气温，然后通过 3 个函数分别实现下述功能：

（1）打印每天温度的柱状图（仅考虑温度为正值的情况）。

（2）打印所有峰点的位置（该月的第几天）及峰值。如果没有，则打印没有峰值。

（3）打印最长的崮的长度。**只使用一重循环即可求出。**



（2）问题要求

请实现以下函数声明，要求能得到如下图所示的运行结果。

```
/*************************************************
 *    文件名: WeatherForecast.cpp
 *    概要:   模拟天气预报，一个月的温度分析
 *    函数:
 *      1. displayTemps，显示月间温度的柱状图
 *      2. displayPeaks，显示月间温度中的峰值
 *      3. displayFlat，  显示月间持续最久的温度
 *      4. inputTemps，   接收用户输入
 *************************************************/
#include <iostream>

// 输入n个温度
void inputTemps(int temp[], int n);
// 显示柱状图
void displayTemps(int temp[], int n);
// 显示月间温度中的所有峰值
void displayPeaks(int temp[], int n);
// 显示月间持续最久的温度
void displayFlat(int temp[], int n);

// 主函数
int main()
{
    int temps[30];

    inputTemps(temps, 30);
    displayTemps(temps, 30);
    displayPeaks(temps, 30);
    displayFlat(temps, 30);

    return 0;
}
```

```
Please input the tempratures:
11 12 13 11 11 11 11 10 9 13 13 11 16 14 15

显示柱状图如下：
1       ***********
2       ************
3       *************
4       ***********
5       ***********
6       ***********
7       ***********
8       **********
9       *********
10      *************
11      *************
12      ***********
13      ****************
14      **************
15      ***************

显示峰值如下：
Max at day 3 is 13
Max at day 13 is 16

显示崮的长度如下：
The length of longest flat is 4
Press any key to continue
```

2.2 处理零下温度

（1）要求柱状图能够处理多个零下温度的情况，以如下形式打印。（10 分）

```
-2      **|
 4        |****
-5  ******|
 6        |******
 3        |***
```

（2）求出现次数最多的温度，及其出现次数。（10 分）

例如：12 13 12 12 14 13 13 12 13 13 中，出现次数最多的是 13 度，出现了 5 次。

2.3 滑动积木块游戏

（1）问题描述

滑动积木块游戏的棋盘结构及某一种将牌的初始排列结构如图所示。

图 1. 3 滑块游戏的初始格局

其中，B 表示黑色将牌，W 表示白色将牌，E 表示空格。我们称将牌的排列结构称为格局，而根据单色将牌的个数，将游戏分别称为 3 滑块或 4 滑块游戏等。所以，上图就是 3 滑块游戏的初始格局。我们可以用字符串来代表格局，代表上图中初始格局的字符串为 BBBWWWE。

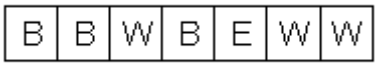
游戏的规定走法是：

（1）任意一个将牌可以移入相邻的空格；

（2）任意一个将牌可相隔 1 个或 2 个其他的将牌跳入空格。

游戏要达到的目标是使所有白将牌都处在黑将牌的左边（左边有无空格均可），我们称为目标格局。很显然，3 滑块游戏的目标格局共有 7 种。

随着将牌的移动，我们会得到一些中间格局，例如：



滑块游戏的某个中间格局

对于某个格局，通过一次移动滑块而得到的格局，称为其后继格局。**我们需要找到某个中间格局的所有后继格局，即每种可能的走法所能得到的格局。**

（2）输入

输入的第一行是一个整数 N，表示共有 N 个格局。后续紧跟 N 行，每行由两部分构成，第一部分是一个整数 n，表示这是一个 n 滑块游戏，第二部分是 2n+1 个字符，表示该游戏的某个格局。

（3）输出

首先判断输入的格局是否是目标格局，如果是，则输出“目标格局”后结束；

如果不是目标格局，则按顺序（将格局看作字符串后，按照字典序排序，即 B < E < W）输出该格局的所有后继格局。例如，BBEBWWW 格局应该排在 BBWBWEW 格局的前面，因为在英文字母表中，第一个格局中的第 3 个字符 E 排在第二个格局的第 3 个字符 W 前面

（4）示例

输入

2

3 BBWBEWW

4 WWWWBBEBB

输出

结果_1

BBEBWWW

BBWBWEW

BBWBWWE

BBWEBWW

BEWBBWW

结果_2

目标格局