**1.memset用法总结**

给定如下程序:

int main(){

int a[10];

**memset(a,0,sizeof(a));**

for(int i=0;i<10;i++)

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

return 0;

}

memset(a,0,sizeof(a));以字节为单位进行内存填充,首先将第二个参数转换为16进制，若该16进制数不够一字节，则在前面补0，若大于一字节，则将该十六进制数高位舍掉使得该数字占一字节，填充进去的内容被机器当作补码。可见如下几例:  
**memset(a,0,sizeof(a));**

输出0 0 0 0 0 0 0 0 0 0

**memset(a,1,sizeof(a));**

将1转换成一字节的16进制是01，则每个int占4字节是0x01010101，该十六进制补码表示十进制数字16843009

**memset(a,11,sizeof(a));**

将11转换成一字节的16进制是0b，则每个int占4字节是0x0b0b0b0b，该十六进制补码表示十进制数字185273099

**memset(a,17,sizeof(a));**

将17转换成一字节的16进制是11，则每个int占4字节是0x11111111，该十六进制补码表示十进制数字286331153

**memset(a,150,sizeof(a));**

将150转换成一字节的16进制是96，则每个int占4字节是0x96969696，该十六进制补码表示十进制数字**-**1768515946

**memset(a,693,sizeof(a));**

将693转换成16进制是2b5，舍去高位是:b5，则每个int占4字节是0xb5b5b5b5，该十六进制补码表示十进制数字**-**1246382667

**memset(a,0x3f3f3f3f,sizeof(a));**

十六舍去高位是3e，则每个int占4字节是0x3f3f3f3f，该十六进制补码表示十进制数字1061109567

**2.continue语句**

continue语句将保持循环，但又使得当前迭代立即结束。

例子：

int testVal = 0;

while (testVal < 10)

{

testVal++;

if (testVal== 4) continue; //终止循环的该次迭代

cout << testVal << " ";

}

输出:1 2 3 5 6 7 8 9 10

**3.1308题解**

首先通过bfs，得到到达各点需要去掉的最少的障碍物，这通过一种贪心的思想来实现，即:

如果本轮bfs到达该点所需去掉的障碍物更少，那么就应该以该轮bfs为基点继续bfs，也就是应该将该点压入队列。否则不压入队列。

然后进行枚举，当三点相互可达时，则三点可相聚于一点，于是问题转化为最少需要去掉多少个格子的障碍物，可以使他们三人之间相聚于一点。我们想求的相聚的那点一定具有这样的性质:在到达该点的过程中，障碍物不会被重复计算，可知这样的聚点一定存在，我们可称其为“关节聚点”。