buglab 实验报告

shuffle

1.首先观察main函数，发现判断边界的条件不正确，故将b>=n改成b>=m。
2.对于异或类型的swap函数，考虑到a和b相等时两者都变为0存在漏洞，故设置一个a!=b的条件即可。

polycalc

1.对于读写的变量，一定要明确其是否进行了初始化，本题一个关键要点就是将result分别初始化为0和1。
2.将node.exp这个unsigned数强制转成int再判断大小，防止程序无限循环。

violetStore

1.因为malloc不会调用构造和析构函数进行初始化，故将第一句改成price* test=new price，相应地末尾改成delete price,析构函数delete后加上[]，这样保证了内存不会泄露。
2.对于*prices++这个语句，为了避免指针移动，考虑将private里的n初始化成0，用prices[n]调用即可。

swapCase

1.由于scanf遇到空格会终止，故使用cin.getline来读取一行字符串。
2.此时直接提交会超时，这是因为strlen函数调用过多次，故调用一次保存其值传入for循环即可。

xorsum

1.首先将ans初始化为0，防止未初始化就进行异或操作。
2.接着调试发现ReanInt调用次序为从右往左，故分别设置a，b=ReadInt(),再Replace(a,b)使其从左往右传递参数。

mergeIntervals

1.由于规范的compare函数不允许等号，故改<=为<。
2.在语义上注意到，合并时last.r应该为last.r和it->r中较大者，添加一条if语句判断即可。

8num

1.对于State curs = State(que.top().first);这行代码，由于curs是局部变量，每一层循环结束后即被清理，故导致后续state无法回溯找到parent，改成State* curState = new State(que.top().first);保证其不被清理。
2.在while(curState)内,构造临时的State* temp= curState来防止free掉curState的parent。

segtree

运行发现程序输出答案不正确，且观察到输出的数据出现了负数，推测可能是溢出问题，故将add、sum、Query改成long long以防止值溢出即可通过此题。

antbuster

1.一些非读取的全局变量(如clk、spn、end等)没有赋予初值，将其赋值。
2.根据题目描述一步步推理，发现Move函数里对于特殊情况的处理出现错误，且specialMove函数中dir为-1直接返回-1，否则应进行dir=(dir+3)%4来确保dir非负。
3.在1和2处理后发现程序依然不正确，仔细观察Fire函数，发现了dx没有检验是否为0就直接相除了，因此补充上dx为0时的情况。
4.最后在调试时发现sign数组出现了负数，重读题目发现sign数组值至少为0，DecreaseSignal函数没有判断sign是否大于0就直接递减，对此处进行修补后发现即可通过此题。

softDouble

1.首先运行1+1发现程序根本无法输出，编译器提示有很多可能溢出的地方，故将1<<51等较大的移位值设置为uint_64并定义在全局变量中，修改相对应的引用区域防止溢出。
2.接着陷入了僵局，不过根据助教在群里的提示发现了ediff移位需要在64位后直接变为0，故添加条件作相应的判断。
3.调试发现输入-1/0得到的是inf，故推测程序应该漏掉了对NINF的讨论，检查后发现divide和writetostring函数只考虑了INF的情况，故补充上NINF的情况。
4.此时程序依旧不对，想了想还是舍入部分很可能会有问题，去仔细看了下rounding，发现程序只有四舍六入，缺少五向偶数进位，故补上(ansf&3)>2即可。