注意到n的最大数据为1e9，因此用数组存储所有层数的情况是不可能的，即使我们记录的是各层到s层的距离也需要5e8的空间

但是这道题同样有一个有趣的数据，k的最大值是min(n-1,1000)，也就是k<=1000，关闭的厕所最多只有1000个

那么实际上我们最多只需要移动1000层（如果起始位置在1层或者n层你只有单向可以走所以不是500层）就肯定能找到一层未关闭的厕所，因而我们实际只需要记录距离s层在1000层以内厕所的开关情况即可

C语言标程:

#include<stdio.h>

#include<string.h>

#include<math.h>

#define ll long long

**int** main()

{

**int** t,n,s,k,flag[1007];//flag[i]记录了距离s层i层的位置关闭了几层

    scanf("%d",&t);

**while**(t--)

    {

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

        scanf("%d%d%d",&n,&s,&k);

**for**(**int** i=0;i<k;i++)

        {

**int** x;scanf("%d",&x);

**if**(abs(s-x)<1007) flag[abs(s-x)]++;

        }

**if**(flag[0]==0) printf("0\n");//当前s层未关闭直接输出0

**else**

        {

**for**(**int** i=01;i<1007;i++)

            {

**if**(s-i>0&&s+i<=n)//这里判断距离s层i层的两个层数是否都在1--n层这个范围内

                {

**if**(flag[i]<2)

                    {

                        printf("%d\n",i);

**break**;

                    }

                }

**else** **if**(flag[i]<1)

                {

                    printf("%d\n",i);

**break**;

                }

            }

        }

    }

}

C++标程:

#include<bits/stdc++.h>

#define ll long long

#define IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

//取消cin,cout同步，可以使得cin,cout提速。

//取消同步后在cf上与scanf,printf同速度，其他大部分oj上仍然慢与scanf,printf

**using** **namespace** std;

**int** main()

{

    IOS

**int** t,n,s,k;//flag[i]记录了距离s层i层的位置关闭了几层

    vector<**int**>flag;

    cin>>t;

**while**(t--)

    {

        flag.clear();

        flag.resize(1007,0);

        cin>>n>>s>>k;

**for**(**int** i=0;i<k;i++)

        {

**int** x;cin>>x;

**if**(abs(s-x)<1007) flag[abs(s-x)]++;

        }

**if**(flag[0]==0) cout<<0<<endl;//当前s层未关闭直接输出0

**else**

        {

**for**(**int** i=01;i<1007;i++)

            {

**if**(s-i>0&&s+i<=n)//这里判断距离s层i层的两个层数是否都在1--n层这个范围内

                {

**if**(flag[i]<2)

                    {

                        cout<<i<<endl;

**break**;

                    }

                }

**else** **if**(flag[i]<1)

                {

                    cout<<i<<endl;

**break**;

                }

            }

        }

    }

}