P1135 奇怪的电梯

题目描述

呵呵，有一天我做了一个梦，梦见了一种很奇怪的电梯。大楼的每一层楼都可以停电梯，而且第 i层楼（1≤i≤N）上有一个数字 Ki（0≤Ki≤N）。电梯只有四个按钮：开，关，上，下。上下的层数等于当前楼层上的那个数字。当然，如果不能满足要求，相应的按钮就会失灵。例如：3,3,1,2,5 代表了 Ki（K_1=3，K_2=3，……），从 1 楼开始。在 1 楼，按“上”可以到 444 楼，按“下”是不起作用的，因为没有 -2 楼。那么，从 A 楼到 B 楼至少要按几次按钮呢？

输入格式

共二行。

第一行为三个用空格隔开的正整数，表示 N,A,B（1≤N≤200，1≤A,B≤N）。

第二行为 N 个用空格隔开的非负整数，表示 K_i。

输出格式

一行，即最少按键次数，若无法到达，则输出 -1。

输入样例

```
5 1 5
3 3 1 2 5
```

输出样例

```
3
```

解析

一道简单的搜索题，我们同时使用广搜和深搜进行解决。

深搜：只有上下两个方向，注意边界值的判断。代码如下

```
#include<bits/stdc++.h>

using namespace std;
//楼梯层数，起始点和终止点
int n, startPoint, endPoint;
//最大值常量
int const maxNum = 0x3f3f3f3f;
//想求最小值，需要定义一个极大值
int ans = maxNum;
//当前楼层所能到达的层数列表
```

```
int to[205];
//访问列表
bool vis[205];

//now表示当前搜到的楼层,sum表示按钮次数
void dfs(int now, int sum) {
    //判断是否到达了目标楼层
    if (now == endPoint) {
        //计算最小操作次数
        ans = min(ans, sum);
    }
    //剪枝优化，如果当前按下的次数已经大于了最小的次数，则放弃
    if (sum > ans) {
        return;
    }
    //将当前点标记已经访问
    vis[now] = true;
    int upFloor = now + to[now];
    int downFloor = now - to[now];
    //不越界就搜
    //向上搜索
    if (upFloor <= n && !vis[upFloor]) {
        dfs(upFloor, sum + 1);
    }
    //向下搜索
    if (downFloor >= 1 && !vis[downFloor]) {
        dfs(downFloor, sum + 1);
    }
    //撤销标记
    vis[now] = false;
}

int main() {
    //读入总的楼梯层数，起始层数，终点层数
    scanf("%d%d%d", &n, &startPoint, &endPoint);
    //读入每层能够通行的层数
    for (int i = 1; i <= n; i++) {
        scanf("%d", &to[i]);
    }
    //启动深搜
    dfs(startPoint, 0);
    //可以到达，输出最小路径步数
    if (ans != maxNum) {
        printf("%d", ans);
    }
        //无法到达，输出-1
    else {
        printf("-1");
    }
```

```
    return 0;
}
```

接下来，我们用广搜来进行解答，同样注意不要越界。

```
#include<bits/stdc++.h>

using namespace std;
int n, startPoint, endPoint, to[205];
//访问列表
bool vis[205];
//节点的结构体
struct node {
    int floor; //楼层号
    int step;  //按下的层数
} x;
//待搜索列表
queue<node> q;

int main() {
    //读入楼层数量、起点楼层、终点楼层
    scanf("%d%d%d", &n, &startPoint, &endPoint);
    //读入每层的到达层数
    for (int i = 1; i <= n; i++) {
        scanf("%d", &to[i]);
    }
    //将起始楼层放入待搜索列表
    q.push((node) {startPoint, 0});
    //遍历全部待搜索列表
    while (q.size()) {
        //取出当前点
        x = q.front();
        q.pop();
        //找到目标终点
        if (x.floor == endPoint) {
            break;
        }
        //将向上的目标楼层加入待搜索列表
        int upFloor = x.floor + to[x.floor];
        if (upFloor <= n && !vis[upFloor]) {
            q.push((node) {upFloor, x.step + 1});
            vis[upFloor] = true;
        }
        //将向下的目标楼层加入待搜索列表
        int downFloor = x.floor - to[x.floor];
        if (downFloor >= 1 && !vis[downFloor]) {
            q.push((node) {downFloor, x.step + 1});
            vis[downFloor] = true;
        }
```

```
        }
        //找到目标楼层，输出使用步数
        if (x.floor == endPoint) {
            printf("%d", x.step);
        }
            //不存在可到达方案，输出-1
        else {
            printf("-1");
        }
        return 0;
}
```