**A strange lift**

**Time Limit: 2000/1000 MS (Java/Others)    Memory Limit: 65536/32768 K (Java/Others)  
Total Submission(s): 19482    Accepted Submission(s): 7214**

Problem Description

There is a strange lift.The lift can stop can at every floor as you want, and there is a number Ki(0 <= Ki <= N) on every floor.The lift have just two buttons: up and down.When you at floor i,if you press the button "UP" , you will go up Ki floor,i.e,you will go to the i+Ki th floor,as the same, if you press the button "DOWN" , you will go down Ki floor,i.e,you will go to the i-Ki th floor. Of course, the lift can't go up high than N,and can't go down lower than 1. For example, there is a buliding with 5 floors, and k1 = 3, k2 = 3,k3 = 1,k4 = 2, k5 = 5.Begining from the 1 st floor,you can press the button "UP", and you'll go up to the 4 th floor,and if you press the button "DOWN", the lift can't do it, because it can't go down to the -2 th floor,as you know ,the -2 th floor isn't exist.  
Here comes the problem: when you are on floor A,and you want to go to floor B,how many times at least he has to press the button "UP" or "DOWN"?

Input

The input consists of several test cases.,Each test case contains two lines.  
The first line contains three integers N ,A,B( 1 <= N,A,B <= 200) which describe above,The second line consist N integers k1,k2,....kn.  
A single 0 indicate the end of the input.

Output

For each case of the input output a interger, the least times you have to press the button when you on floor A,and you want to go to floor B.If you can't reach floor B,printf "-1".

Sample Input

5 1 5

3 3 1 2 5

0

Sample Output

3

分析：

在要求得范围内构建一个图 然后dijkstra算法求最短路

主要是如何构造图

在n个点都有与之对应的上升或者下降多少层 那么我们可以根据这个条件来构造一个有向图

**for(int** i **=** 1**;** i **<=** n**;** i**++)**

**{**

scanf**(**"%d"**,&**a**);**

**if(**i **+** a **<=** n**)**

map**[**i**][**i**+**a**] =** 1**;**

**if(**i **-** a **>=** 1**)**

map**[**i**][**i**-**a**] =** 1**;**

**}**

最后所求的“最短路“就是要按按钮的次数

AC代码：

#include <iostream>

#include <cstring>

#include <cstdio>

#define INF 0x3f3f3f3f

#define MAX 1000

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

**int** n**;**

**int** map**[**MAX**][**MAX**],**dist**[**MAX**],**s**[**MAX**];**

**void** dij**(int** v**)**

**{**

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

**for(int** i **=** 1**;** i **<=** n**;** i**++)**

**{**

dist**[**i**] =** map**[**v**][**i**];**

s**[**i**] =** 0**;**

**}**

dist**[**v**] =** 0**;**

s**[**v**] =** 1**;**

**int** u **=** 0**,**min**;**

**for(int** i **=** 2**;** i **<=** n**;** i**++)**

**{**

min **=** INF**;**

**for(int** j **=** 1**;** j **<=** n**;** j**++)**

**{**

**if(**s**[**j**]==**0**&&**dist**[**j**]<**min**)**

**{**

min **=** dist**[**j**];**

u **=** j**;**

**}**

**}**

**if(**min **==** INF**)**

**break;**

s**[**u**] =** 1**;**

**for(int** j **=** 1**;** j **<=** n**;** j**++)**

**{**

**if(**s**[**j**]==**0**&&**map**[**u**][**j**] +** dist**[**u**] <** dist**[**j**])**

dist**[**j**] =** map**[**u**][**j**] +** dist**[**u**];**

**}**

**}**

**}**

**int main()**

**{**

**while(~**scanf**(**"%d"**,&**n**))**

**{**

**if(**n**==**0**)**

**break;**

**int** A**,**B**,**a**;**

memset**(**map**,**INF**,sizeof(**map**));**

scanf**(**"%d%d"**,&**A**,&**B**);**

**for(int** i **=** 1**;** i **<=** n**;** i**++)**

**{**

scanf**(**"%d"**,&**a**);**

**if(**i **+** a **<=** n**)**

map**[**i**][**i**+**a**] =** 1**;**

**if(**i **-** a **>=** 1**)**

map**[**i**][**i**-**a**] =** 1**;**

**}**

dij**(**A**);**

**if(**dist**[**B**] <** INF**)**

printf**(**"%d\n"**,**dist**[**B**]);**

**else**

printf**(**"-1\n"**);**

**}**

**return** 0**;**

**}**