PAT_A1008.Elevator (20)

1. Abstraction

1.1 Algorithm and idea

```
* simple stimulation
```

1.2 Notice

```
1.即使在最后一层也需要等待
```

2. Problem

The highest building in our city has only one elevator. A request list is made up with N positive numbers. The numbers denote at which floors the elevator will stop, in specified order. It costs 6 seconds to move the elevator up one floor, and 4 seconds to move down one floor. The elevator will stay for 5 seconds at each stop.

For a given request list, you are to compute the total time spent to fulfill the requests on the list. The elevator is on the 0th floor at the beginning and does not have to return to the ground floor when the requests are fulfilled.

Input Specification:

Each input file contains one test case. Each case contains a positive integer N, followed by N positive numbers. All the numbers in the input are less than 100.

Output Specification:

For each test case, print the total time on a single line.

Sample Input:

```
3 2 3 1
```

Sample Output:

```
41
```

3.Code

3.1 Edit 0

3.1.1 Algorithm abstraction

```
* None
```

3.1.2 Notice

```
* None
```

```
#include <cstdio>
#include <cstring>

int main(){
    int n,total=0,now=0,to;
    scanf("%d",&n);
    for(int i=0;i<n;++i){
        scanf("%d",&to);
        if(to>now){
            total+=((to-now)*6);
        }
        else{
            total+=((now-to)*4);
        }
        total+=5;
        now=to;
    }
    printf("%d\n",total);
    return 0;
}
```

```
#include <iostream>
#include <string>
#include <vector>
const int twait=5;
const int tup=6;
const int tdown=4;

using std::cin;
```

```cpp
using std::cout;
using std::flush;
using std::endl;
using std::string;
using std::vector;



int main(){
    int counter;
    cin>>counter;
    int nowLocation=0;
    int objLocation=0;
    int total_time=0;

    while(counter--){
        cin>>objLocation;
        if(objLocation>nowLocation){
            total_time+=(objLocation-nowLocation)*tup;
        }
        else if(objLocation<nowLocation){
            total_time+=(nowLocation-objLocation)*tdown;
        }
        else if(objLocation==nowLocation){
            ;
        }
        nowLocation=objLocation;
        total_time+=twait;
    }
    cout<<total_time<<endl;
    return 0;
}
```