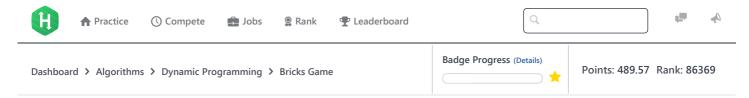
12/21/2017 HackerRank







Problem Submissions Leaderboard Discussions Editorial

Русский \| 中文

You and your friend decide to play a game using a stack consisting of N bricks. In this game, you can alternatively remove 1, 2 or 3 bricks from the top, and the numbers etched on the removed bricks are added to your score. You have to play so that you obtain the maximum possible score. It is given that your friend will also play optimally and you make the first move.

Input Format

First line will contain an integer T i.e. number of test cases. There will be two lines corresponding to each test case: first line will contain a number N i.e. number of elements in the stack and next line will contain N numbers i.e. numbers etched on bricks from top to bottom.

Constraints

```
1 \le T \le 5

1 \le N \le 10^5

0 \le \text{ each number on brick} \le 10^9
```

Output Format

For each test case, print a single line containing your maximum score.

Sample Input

```
2
5
999 1 1 1 0
5
0 1 1 1 999
```

Sample Output

1001 999

Explanation

In first test case, you will pick 999,1,1. If you play in any other way, you will not get a score of 1001.

In second case, best option will be to pick up the first brick (with 0 score) at first. Then your friend will choose the next three blocks, and you will get the last brick.



12/21/2017 HackerRank

```
C++14
 Current Buffer (saved locally, editable) &
 1 ▼ #include <cmath>
 2 #include <cstdio>
 3 #include <vector>
 4 #include <iostream>
    #include <algorithm>
 6
   using namespace std;
 8 vint main() {
9
        int T;
10
        cin>>T;
        while(T > 0) {
11 ▼
12
            int N;
13
            cin>>N;
14 ▼
            long long *bricks = new long long[N];
15 ▼
            long long *sum = new long long[N]; // sum[i] = sum of bricks up to i
            for(int i = 0; i < N; i++) {
16 ▼
17 ▼
                cin>>bricks[N-i-1];
18
            }
19 ▼
            sum[0] = bricks[0];
            for(int i = 1; i < N; i++) {
20 ▼
21 ▼
                sum[i] = sum[i-1] + bricks[i];
22
23 ▼
            if(N < 4) {
24
                // Edge case
25 ▼
                cout<<sum[N-1]<<"\n";</pre>
26
                T--;
27 ▼
            } else {
28
                // Notice that taking Y bricks is equivalent to playing a new game with N-Y bricks where
29
                // your friend goes first and you start with a score of the sum of the Y bricks.
30 ▼
                long long *dp = new long long[N];
31 ▼
                dp[0] = sum[0];
32 ▼
                dp[1] = sum[1];
33 ▼
                dp[2] = sum[2];
                for(int i = 3; i < N; i++) {
34 ▼
35
                     // Take the # of bricks which maximizes your score
36 ▼
                     dp[i] = sum[i] - dp[i-3]; // 3 bricks
37 ▼
                     dp[i] = max(dp[i], sum[i] - dp[i-2]); // 2 bricks
38 ▼
                     dp[i] = max(dp[i], sum[i] - dp[i-1]); // 1 brick
39
40
                cout<<dp[N-1]<<"\n";</pre>
41
                T--;
42
            }
43
        }
44
        return 0;
45
    }
46
                                                                                                                Line: 1 Col: 1
```

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature

1 Upload Code as File

Test against custom input

Run Code

Submit Code