■ **HackerRank** | **Prepare** **Certify** Compete **Apply**   Q Search

Bank Robbery   🔒 locked

Problem	Submissions	Discussions

Problem Statement

A thief is planning to steal gold from bank lockers. There are **N** lockers in the bank, and each of them has **A[i]** gold bars in it. For security reasons, the keys of the lockers are connected. This means that if you can open the **ith** locker, you'll also get the key to the next lockers inside that locker. The thief is very powerful and can break down any locker, but he can only do this once in the time frame of stealing the entire bank.

However, there are some trackers in the lockers. If the thief opens a locker with a tracker, he will get caught, and all the gold bars will be lost. So he wants to avoid this. Also, if he opens two lockers which were presented there consecutively, the tracker will also be activated.

Can you determine the maximum number of gold bars that the thief can obtain?

Input Format

- First line will contain **N**, the number of lockers in the bank.

- Next line will contain **N** values, the number of gold bars in ith locker. (1 <= i <= N)

Constraints

1. 1 <= **N** <= 100000

2. 0 <= **A[i]** <= 1000000000 or -1; -1 means there is a tracker.

Output Format

- Output maximum number of gold bars the thief can steal.

Sample Input 0

```
5
10 20 5 7 15
```

Sample Output 0

```
35
```

Sample Input 1

```
5
10 20 -1 -1 15
```

Sample Output 1

35

Facebook   Twitter   LinkedIn

Submissions: 9
Max Score: 1

Rate This Challenge:
☆ ☆ ☆ ☆ ☆

More

C++20   ▾

```cpp
1   #include <cmath>
2   #include <cstdio>
3   #include <vector>
4   #include <iostream>
5   #include <algorithm>
6   using namespace std;
7
8
9   int main() {
10      /* Enter your code here. Read input from STDIN. Print output to STDOUT */
11      return 0;
12  }
13
```

Line: 1 Col: 1

⬆ Upload Code as File   ☐ **Test against custom input**   Run Code   Submit Code

---

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |