Knapsack

by trophies

Given a list of n integers, $A = \{a_1, a_2, \ldots, a_n\}$, and another integer, k representing the *expected sum*. Select zero or more numbers from A such that the sum of these numbers is as near as possible, but not exceeding, to the *expected sum* (k).

Note

- Each element of A can be selected multiple times.
- If no element is selected then the sum is 0.

Input Format

The first line contains T the number of test cases.
Each test case comprises of two lines. First line contains two integers, n k, representing the length of list A and *expected sum*, respectively. Second line consists of n space separated integers, $a_1, a_2, \ldots, a_n$, representing the elements of list A.

Constraints
$1 \le T \le 10$
$1 \le n \le 2000$
$1 \le k \le 2000$
$1 \le a_i \le 2000, where\ i \in [1, n]$

Output Format

Output T lines, the maximum sum for each test case which is as near as possible, but not exceeding, to the expected sum (k).

Sample Input

```
2
3 12
1 6 9
5 9
3 4 4 4 8
```

Sample Output

```
12
9
```

Explanation

In the first test case, one can pick {6, 6}. In the second, we can pick {3,3,3}.

Submissions: 11145
Max Score: 60
Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

More

Current Buffer (saved locally, editable)

Java 8

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int tst = Integer.parseInt(br.readLine());

        for(int i = 0 ; i < tst ; i++){

            String[] numbers = br.readLine().split("\\s");
            int N = Integer.parseInt(numbers[0]);
            int sum = Integer.parseInt(numbers[1]);

            numbers = br.readLine().split("\\s");

            int gcd = Integer.parseInt(numbers[0]);

            for(int j = 1 ; j < N ; j++){

                if(gcd == 1){
                    break;
                }

                gcd = findGCD(gcd,Integer.parseInt(numbers[j]));
            }

            if(gcd == 1){
                System.out.println(sum);
            }
            else{
                System.out.println(sum - (sum % gcd));
            }

        }

    }


    public static int findGCD(int a, int b){

        if(b > a){
            int temp = b;
            b = a;
            a = temp;
        }

        while(true){

            if(a % b == 0){
                break;
            }
            else{
                int temp = b;
                b = a % b;
                a = b;
            }

        }

        return b;

    }
}
```

Line: 1 Col: 1

Upload Code as File

Test against custom input

Run Code

Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.

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