

SRISHA S

B.E COMPUTER SCIENCE & DESIGN

241701056

WEEK 6

Problem Statement 1:

In this challenge, we're going to use loops to help us do some simple math. Check out the Tutorial tab to learn more.

Task

Given an integer, n , print its first 10 multiples. Each multiple $n \times i$ (where $1 \leq i \leq 10$) should be printed on a new line in the form: $n \times i = \text{result}$.

Input Format

A single integer, n .

Constraints

$2 \leq n \leq 20$

Output Format

Print 10 lines of output; each line i (where $1 \leq i \leq 10$) contains the result of $n \times i$ in the form:

$n \times i = \text{result}$.

Sample Input

2

Sample Output

$2 \times 1 = 2$

$2 \times 2 = 4$

$2 \times 3 = 6$

$2 \times 4 = 8$

$$2 \times 5 = 10$$

$$2 \times 6 = 12$$

$$2 \times 7 = 14$$

$$2 \times 8 = 16$$

$$2 \times 9 = 18$$

$$2 \times 10 = 20$$

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main(){
3     int n;
4     scanf("%d",&n);
5     for(int i=1;i<=10;i++){
6         printf("%d x %d = %d\n",n,i,n*i);
7     }
8     return 0;
9 }
```

	Input	Expected	Got	
✓	2	2 x 1 = 2 2 x 2 = 4 2 x 3 = 6 2 x 4 = 8 2 x 5 = 10 2 x 6 = 12 2 x 7 = 14 2 x 8 = 16 2 x 9 = 18 2 x 10 = 20	2 x 1 = 2 2 x 2 = 4 2 x 3 = 6 2 x 4 = 8 2 x 5 = 10 2 x 6 = 12 2 x 7 = 14 2 x 8 = 16 2 x 9 = 18 2 x 10 = 20	✓

Passed all tests! ✓

Problem Statement 2:

A nutritionist is labeling all the best power foods in the market. Every food item arranged in a single line, will have a value beginning from 1 and increasing by 1 for each, until all items have a value associated with them. An item's value is the same as the number of macronutrients it has. For example, food item with value 1 has 1 macronutrient, food item with value 2 has 2 macronutrients, and incrementing in this fashion.

The nutritionist has to recommend the best combination to patients, i.e. maximum total of macronutrients. However, the nutritionist must avoid prescribing a particular sum of macronutrients (an

'unhealthy' number), and this sum is known. The nutritionist chooses food items in the increasing order of their value. Compute the highest total of macronutrients that can be prescribed to a patient, without the sum matching the given 'unhealthy' number.

Here's an illustration:

Given 4 food items (hence value: 1,2,3 and 4), and the unhealthy sum being 6 macronutrients, on choosing items 1, 2, 3 -> the sum is 6, which matches the 'unhealthy' sum. Hence, one of the three needs to be skipped. Thus, the best combination is from among:

$$2+3+4=9$$

$$1+3+4=8$$

$$1 + 2 + 4 = 7$$

Since $2 + 3 + 4 = 9$, allows for maximum number of macronutrients, 9 is the right answer.

Complete the code in the editor below. It must return an integer that represents the maximum total of macronutrients, modulo $1000000007(10^9 + 7)$.

It has the following:

n: an integer that denotes the number of food items

k: an integer that denotes the unhealthy number

Constraints

$$1 \leq n \leq 2 \times 10^9$$

$$1 \leq k \leq 4 \times 10^{15}$$

Input Format For Custom Testing

The first line contains an integer, **n**, that denotes the number of food items.

The second line contains an integer, **k**, that denotes the unhealthy number.

Sample Input 0

2

2

Sample Output 0

3

Explanation 0

The following sequence of $n = 2$ food items:

1. Item 1 has 1 macronutrients.
2. $1 + 2 = 3$; observe that this is the max total, and having avoided having exactly $k = 2$ macronutrients.

Sample Input 1

2

1

Sample Output 1

2

Explanation 1

1. Cannot use item 1 because $k = 1$ and $\text{sum} = k$ has to be avoided at any time.
2. Hence, max total is achieved by $\text{sum} = 0 + 2 = 2$.

Sample Input For Custom Testing**Sample Input 2**

3

3

Sample Output 2

5

Explanation 2

2 + 3 = 5, is the best case for maximum nutrients.

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main(){
3     long long int n,t,i,nut=0;
4     scanf("%lld %lld",&n,&t);
5     for(i=1;i<=n;i++){
6         nut=nut+i;
7         if(nut==t){
8             nut=nut-1;
9         }
10    }
11    printf("%lld",nut%1000000007);
12 }
```

	Input	Expected	Got	
✓	2 2	3	3	✓
✓	2 1	2	2	✓
✓	3 3	5	5	✓

Passed all tests! ✓

Problem Statement 3:

Determine all positive integer values that evenly divide into a number, its factors. Return the pth element of your list, sorted ascending. If there is no pth element, return 0.

For example, given the number $n = 20$, its factors are $\{1, 2, 4, 5, 10, 20\}$. Using 1-based indexing if $p = 3$, return 4. If $p > 6$, return 0.

Complete the code in the editor below. The function should return a long integer value of the pth integer factor of n .

It has the following:

n : an integer

p : an integer

Constraints

- $1 \leq n \leq 10^{15}$
- $1 \leq p \leq 10^9$

Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer n, the number to factor.

The second line contains an integer p, the 1-based index of the factor to return.

Sample Input 0

10

3

Sample Output 0

5

Explanation 0

Factoring n = 10 we get (1, 2, 5, 10}. We then return the p = 3rd factor as our answer.

Sample Input 1

10

5

Sample Output 1

0

Explanation 1

Factoring n = 10 we get {1, 2, 5, 10}. There are only 4 factors and p = 5. We return 0 as our answer.

Sample Input 2

1

1

Sample Output 2

1

Explanation 2

Factoring $n = 1$ we get $\{1\}$. We then return the $p = 7$ st factor as our answer.

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main(){
3     long n,p,count=0,result=0;
4     scanf("%ld%ld",&n,&p);
5     for(long i=1;i<=n;i++){
6         if(n%i==0){
7             count++;
8             if(count==p){
9                 result=i;
10                break;
11            }
12        }
13    }
14    if(count<p){
15        result=0;
16    }
17    printf("%ld\n",result);
18    return 0;
19 }
```

	Input	Expected	Got	
✓	10 3	5	5	✓
✓	10 5	0	0	✓
✓	1 1	1	1	✓

Passed all tests! ✓

Problem Statement 4:

Given an array A of sorted integers and another non negative integer k , nd if there exists 2 indices i and j such that $A[i] - A[j] = k$, $i \neq j$.

Input Format

- 1. First line is number of test cases T . Following T lines contain:**
- 2. N , followed by N integers of the array**
- 3. The non-negative integer k**

Output format

Print 1 if such a pair exists and 0 if it doesn't.

Sample Input:

1

3 1 3 5

4

Sample Output:

1

Sample Input 1:

1

3 1 3 5

99

Sample Output 1:

0

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int t;
5     scanf("%d",&t);
6     while(t-->0)
7     {
8         int n;
9         scanf("%d",&n);
10        int a[n];
11        for(int i=0;i<n;i++){
12            scanf("%d",&a[i]);
13        }
14        int k;
15        scanf("%d",&k);
16        int temp=0;
17        for(int i=0;i<n;i++){
18            for(int j=i+1;j<n;j++){
19                if(a[i]-a[j]==k||a[j]-a[i]==k){
20                    temp=1;
21                    break;
22                }
23            }
24        }
25        if(temp)break;
26        printf("%d\n",temp);
27    }
28 }
29
```

	Input	Expected	Got	
✓	1 3 1 3 5 4	1	1	✓
✓	1 3 1 3 5 99	0	0	✓

Passed all tests! ✓

Problem Statement 5:

Sam loves chocolates and starts buying them on the 1st day of the year. Each day of the year, x , is numbered from 1 to Y . On days when x is odd, Sam will buy x chocolates; on days when x is even, Sam will not purchase any chocolates.

Complete the code in the editor so that for each day N_i (where $1 \leq x \leq N \leq Y$) in array `arr`, the number of chocolates Sam purchased (during days 1 through N) is printed on a new line. This is a function-only challenge, so input is handled for you by the locked stub code in the editor.

Input Format

The program takes an array of integers as a parameter

The locked code in the editor handles reading the following input from `stdin`, assembling it into an array of integers (`arr`), and calling `calculate(arr)`.

The first line of input contains an integer, T (the number of test cases). Each line i of the T subsequent lines describes the i th test case as an integer, N_i (the number of days).

Constraints

$$1 \leq T \leq 2 \times 10^5$$

$$1 \leq N \leq 2 \times 10^6$$

$$1 \leq x \leq N \leq Y$$

Output Format

For each test case, T_i in `arr`, your `calculate` method should print the total number of chocolates Sam purchased by day N_i on a new line.

Sample Input 0

3

1

2

3

Sample Output 0

1

1

4

Explanation

Test Case 0: N = 1

Sam buys 1 chocolate on day 1, giving us a total of 1 chocolate. Thus, we print 1 on a new line.

Test Case 1: N = 2

Sam buys 1 chocolate on day 1 and 0 on day 2. This gives us a total of 1 chocolate. Thus, we print 1 on a new line.

Test Case 2: N = 3

Sam buys 1 chocolate on day 1, 0 on day 2, and 3 on day 3. This gives us a total of 4 chocolates. Thus, we print 4 on a new line.

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main(){
3     int t;
4     scanf("%d",&t);
5     while(t--){
6         int n,c=0;
7         scanf("%d",&n);
8         for(int i=0;i<=n;i++){
9             if(i%2!=0)
10                c=c+i;
11         }
12         printf("%d\n",c);
13     }
14 }
```

	Input	Expected	Got	
✓	3	1	1	✓
	1	1	1	
	2	4	4	
	3			
✓	10	1296	1296	✓
	71	2500	2500	
	100	1849	1849	
	86	729	729	
	54	400	400	
	40	25	25	
	9	1521	1521	
	77	25	25	
	9	49	49	
	13	2401	2401	
	98			

Passed all tests! ✓

Problem Statement 6:

The number of goals achieved by two football teams in matches in a league is given in the form of two lists. Consider:

- Football team A, has played three matches, and has scored { 1 , 2 , 3 } goals in each match respectively.
- Football team B, has played two matches, and has scored { 2, 4 } goals in each match respectively.
- Your task is to compute, for each match of team B, the total number of matches of team A, where team A has scored less than or equal to the number of goals scored by team B in that match.
- In the above case:

For 2 goals scored by team B in its first match, team A has 2 matches with scores 1 and 2.

For 4 goals scored by team B in its second match, team A has 3 matches with scores 1, 2 and 3.

Hence, the answer: {2, 3}.

Complete the code in the editor below. The program must return an array of m positive integers, one for each $\text{maxes}[i]$ representing the total number of elements $\text{nums}[j]$ satisfying $\text{nums}[j] \leq \text{maxes}[i]$ where $0 \leq j < n$ and $0 \leq i < m$, in the given order.

It has the following:

$\text{nums}[\text{nums}[0], \dots, \text{nums}[n-1]]$: first array of positive integers

$\text{maxes}[\text{maxes}[0], \dots, \text{maxes}[m-1]]$: second array of positive integers

Constraints

- $2 \leq n, m \leq 105$
- $1 \leq \text{nums}[j] \leq 109$, where $0 \leq j < n$.
- $1 \leq \text{maxes}[i] \leq 109$, where $0 \leq i < m$.

Input Format For Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer n , the number of elements in nums . The next n lines each contain an integer describing $\text{nums}[j]$

where $0 \leq j < n$. The next line contains an integer m , the number of elements in `maxes`. The next m lines each contain an integer describing `maxes[i]` where $0 \leq i < m$.

Sample Input 0

4

1

4

2

4

2

3

5

Sample Output 0

2

4

Explanation 0

We are given $n = 4$, `nums` = [1, 4, 2, 4], $m = 2$, and `maxes` = [3, 5].

1. For `maxes[0]` = 3, we have 2 elements in `nums` (`nums[0]` = 1 and `nums[2]` = 2) that are \leq `maxes[0]`.

2. For `maxes[1]` = 5, we have 4 elements in `nums` (`nums[0]` = 1, `nums[1]` = 4, `nums[2]` = 2, and `nums[3]` = 4) that are \leq `maxes[1]`.

Thus, the function returns the array [2, 4] as the answer.

Sample Input 1

5

2

10

5

4

8

4

3

1

7

8

Sample Output 1

1

0

3

4

Explanation 1

We are given, $n = 5$, $nums = [2, 10, 5, 4, 8]$, $m = 4$, and $maxes = [3, 1, 7, 8]$.

1. For $maxes[0] = 3$, we have 1 element in $nums$ ($nums[0] = 2$) that is $\leq maxes[0]$.

2. For $maxes[1] = 1$, there are 0 elements in $nums$ that are $\leq maxes[1]$.

3. For $maxes[2] = 7$, we have 3 elements in $nums$ ($nums[0] = 2$, $nums[2] = 5$, and $nums[3] = 4$) that are $\leq maxes[2]$.

4. For $maxes[3] = 8$, we have 4 elements in $nums$ ($nums[0] = 2$, $nums[2] = 5$, $nums[3] = 4$, and $nums[4] = 8$) that are $\leq maxes[3]$. 61

Thus, the function returns the array $[1, 0, 3, 4]$ as the answer.

Answer: (penalty regime: 0 %)

1

#include<stdio.h>

2

int main()

3

{

4

int s1,s2,ans;

5

scanf("%d",&s1);

6

int ta[s1];

7

for(int i=0;i<s1;i++)

8

scanf("%d",&ta[i]);

9

scanf("%d",&s2);

10

int tb[s2];

11

for(int i=0;i<s2;i++)

12

scanf("%d",&tb[i]);

13

for(int j=0;j<s2;j++)

14

{

15

ans=0;

16

for(int i=0;i<s1;i++){

17

if (tb[j]>=ta[i])

18

ans++;

19

}printf("%d\n",ans);

20

}

21

}

22

}

	Input	Expected	Got	
✓	4	2	2	✓
	1	4	4	
	4			
	2			
	4			
	2			
	3			
	5			
✓	5	1	1	✓
	2	0	0	
	10	3	3	
	5	4	4	
	4			
	8			
	4			
	3			
	1			
	7			
	8			

Passed all tests! ✓