

[Terms Of Service](#) | [Privacy Policy](#)

[illegible]

Prepare > Problem Solving

## Problem Solving

30 more points to get your first star!

Rank: 5930599 | Points: 0/30



Algorithms | Data Structures

### Solve Me First

Easy, Problem Solving (Basic), Max Score: 1, Success Rate: 97.64%

This is an easy challenge to help you start coding in your favorite language!



Solve Challenge

### Simple Array Sum

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 94.67%



Solve Challenge

### Compare the Triplets

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 95.96%



Solve Challenge

### A Very Big Sum

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 95.64%



Solve Challenge

#### STATUS

- ☐ Solved
- ☐ Unsolved

#### SKILLS

- ☐ Problem Solving (Intermediate)
- ☐ Problem Solving (Advanced)
- ☐ Problem Solving (Basic)

#### DIFFICULTY

- ☐ Easy
- ☐ Medium
- ☐ Hard

#### SUBDOMAINS

- ☐ Warmup

30 more points to get your first star!

Rank: 5930599 | Points: 0/30



Problem

An arcade game player wants to climb to the top of the leaderboard and track their ranking. The game uses [Dense Ranking](#), so its leaderboard works like this:

- The player with the highest score is ranked number 1 on the leaderboard.
- Players who have equal scores receive the same ranking number, and the next player(s) receive the immediately following ranking number.

#### Example

`ranked = [100, 90, 90, 80]`

`player = [70, 80, 105]`

The ranked players will have ranks 1, 2, 2, and 3, respectively. If the player's scores are 70, 80 and 105, their rankings after each game are 4<sup>th</sup>, 3<sup>rd</sup> and 1<sup>st</sup>. Return `[4, 3, 1]`.

#### Function Description

Complete the `climbingLeaderboard` function in the editor below.

`climbingLeaderboard` has the following parameter(s):

- `int ranked[n]`: the leaderboard scores
- `int player[m]`: the player's scores

#### Returns

- `int[m]`: the player's rank after each new score

#### Input Format

The first line contains an integer `n`, the number of players on the leaderboard.

Submissions

Leaderboard

ssions

Change Theme Language JavaScript (Node.js)

```
3 const fs = require('fs');
4
5 process.stdin.resume();
6 process.stdin.setEncoding('utf-8');
7
8 let inputString = '';
9 let currentLine = 0;
10
11 process.stdin.on('data', function(inputStdin) {
12   inputString += inputStdin;
13 });
14
15 process.stdin.on('end', function() {
16   inputString = inputString.split('\n');
17   main();
18 });
19
20 function readLine() {
21   return inputString[currentLine++];
22 }
23
24 function climbingLeaderboard(ranked, player) {
25   let uniqueRanks = [...new Set(ranked)];
```

Line: 53 Col: 1

Upload Code as File

☐ Test against custom input

Run Code

Submit Code

### Problem

The Utopian Tree goes through 2 cycles of growth every year. Each spring, it doubles in height. Each summer, its height increases by 1 meter.

For example, if the number of growth cycles is  $n = 5$ , the calculations are as follows:

### Function Description

utopianTree has the following parameter(s):

## Returns

### Input Format

```

1  'use strict';
2
3  process.stdin.resume();
4  process.stdin.setEncoding('utf-8');
5
6  let inputString = '';
7  let currentLine = 0;
8
9  process.stdin.on('data', function(inputStdin) {
10     inputString += inputStdin;
11 });
12
13 process.stdin.on('end', function() {
14     inputString = inputString.split('\n');
15     main();
16 });
17
18 function readLine() {
19     return inputString[currentLine++];
20 }
21
22 function utopianTree(n) {

```

Line: 43 Col: 1

```

1  'use strict';
2
3  const fs = require('fs');
4
5  process.stdin.resume();
6  process.stdin.setEncoding('utf-8');
7
8  let inputString = '';
9  let currentLine = 0;
10
11 process.stdin.on('data', function (inputStdin) {
12     inputString += inputStdin;
13 });
14
15 process.stdin.on('end', function () {
16     inputString = inputString.split('\n');
17     main();
18 });
19
20 function readLine() {
21     return inputString[currentLine++];
22 }
23
24 function sherlockAndAnagrams(s) {
25     let substrCount = new Map();
26     let totalPairs = 0;
27
28     for (let start = 0; start < s.length; start++) {
29         for (let end = start; end < s.length; end++) {
30             let subStr = s.substring(start, end + 1);
31             let sortedSubStr = subStr.split('').sort().join('');
32
33             substrCount.set(sortedSubStr, (substrCount.get(sortedSubStr) || 0) + 1);
34         }
35     }
36
37     for (let count of substrCount.values()) {

```

HackerRank

PrepareAlgorithmsStringsSherlock And Anagrams

Exit Full Screen View

ProblemSubmissionsLeaderboardDiscussionsEditorial

Two strings are **anagrams** of each other if the letters of one string can be rearranged to form the other string. Given a string, find the number of pairs of substrings of the string that are anagrams of each other.

**Example**

s = mnm

The list of all anagrammatic pairs is [m,n],[n,m],[mn, nm] at positions [[0], [2]], [[0], 1], [1, 2]] respectively.

**Function Description**

Complete the function sherlockAndAnagrams in the editor below.

sherlockAndAnagrams has the following parameter(s):

- string s: a string

**Returns**

- int: the number of unordered anagrammatic pairs of substrings in s

**Input Format**

The first line contains an integer q, the number of queries.

Each of the next q lines contains a string s to analyze.

**Constraints**

- $1 \leq q \leq 10$
- $2 \leq \text{length of } s \leq 100$
- s contains only lowercase letters in the range ascii[a-z].

**Sample Input 0**

```
2
abba
abcd
```

**Sample Output 0**

```
4
0
```

```
1 // return totalPairs;
2
3 }
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
93
```

HackerRank
Prepare Algorithms Implementation Lisa's Workbook
Exit Full Screen View

---

**Problem**

Lisa just got a new math workbook. A workbook contains exercise problems, grouped into chapters. Lisa believes a problem to be special if its index (within a chapter) is the same as the page number where it's located. The format of Lisa's book is as follows:

- There are  $n$  chapters in Lisa's workbook, numbered from 1 to  $n$ .
- The  $i^{\text{th}}$  chapter has  $\text{arr}[i]$  problems, numbered from 1 to  $\text{arr}[i]$ .
- Each page can hold up to  $k$  problems. Only a chapter's last page of exercises may contain fewer than  $k$  problems.
- Each new chapter starts on a new page, so a page will never contain problems from more than one chapter.
- The page number indexing starts at 1.

Given the details for Lisa's workbook, can you count its number of special problems?

**Example**

```
arr = [4, 2]
k = 3
```

Lisa's workbook contains  $\text{arr}[1] = 4$  problems for chapter 1, and  $\text{arr}[2] = 2$  problems for chapter 2. Each page can hold  $k = 3$  problems.

The first page will hold 3 problems for chapter 1. Problem 1 is on page 1, so it is special. Page 2 contains only Chapter 1. Problem 4, so no special problem is on page 2. Chapter 2 problems start on page 3 and there are 2 problems. Since there is no problem 3 on page 3, there is no special problem on that page either. There is 1 special problem in her workbook.

**Note:** See the diagram in the Explanations section for more details.

**Function Description**

Complete the workbook function in the editor below.

workbook has the following parameter(s):

- $\text{int } n$ : the number of chapters
- $\text{int } k$ : the maximum number of problems per page
- $\text{int arr}[]$ : the number of problems in each chapter

**Returns**

- $\text{int}$ : the number of special problems in the workbook

**Input Format**

The first line contains two integers  $n$  and  $k$ , the number of chapters and the maximum number of problems per page. The second line contains  $n$  space-separated integers  $\text{arr}[i]$  where  $\text{arr}[i]$  denotes the number of problems in the  $i^{\text{th}}$  chapter.

Change Theme Language JavaScript (Node.js)
🔍 ↻

```

1 'use strict';
2
3 const fs = require('fs');
4
5 process.stdin.resume();
6 process.stdin.setEncoding('utf-8');
7
8 let inputString = '';
9 let currentLine = 0;
10
11 ✓ process.stdin.on('data', function(inputStdin) {
12   |   inputString += inputStdin;
13   | });
14
15 ✓ process.stdin.on('end', function() {
16   |   inputString = inputString.split('\n');
17   |   main();
18   | });
19
20 ✓ function readline() {
21   |   return inputString[currentLine++];
22   | }
23
24 ✓ function workbook(n, k, arr) {
25   |   let specialProblems = 0;
26   |   let page = 1;
27
28   |   for (let i = 0; i < n; i++) {
29     |     let problems = arr[i];
30     |     for (let j = 1; j <= problems; j++) {
31       |       if (j === page) {
32         |         specialProblems++;
33         |       }
34       |       if (j % k === 0 || j === problems) {
35         |         page++;
36         |       }
37     |   }
          
```

⬇ Upload Codes File
☐ Test against custom input

Run Code
Submit Code

Line: 57 Col: 1

•  $1 \leq n, k, arr[i] \leq 100$

**Sample Input**

```
STDIN      Function
-----
5 3        n = 5, k = 3
4 2 6 1 10 arr = [4, 2, 6, 1, 10]
```

**Sample Output**

```
4
```

**Explanation**

The diagram below depicts Liza's workbook with  $n = 5$  chapters and a maximum of  $k = 3$  problems per page. Special problems are outlined in red, and page numbers are in yellow squares.

There are 4 special problems and thus we print the number 4 on a new line.

```
22 }
23
24 function workbook(n, k, arr) {
25     let specialProblems = 0;
26     let page = 1;
27
28     for (let i = 0; i < n; i++) {
29         let problems = arr[i];
30         for (let j = 1; j <= problems; j++) {
31             if (j === page) {
32                 specialProblems++;
33             }
34             if (j % k === 0 || j === problems) {
35                 page++;
36             }
37         }
38     }
39
40     return specialProblems;
41 }
42
43 function main() {
44     const ws = fs.createWriteStream(process.env.OUTPUT_PATH);
45
46     const firstMultipleInput = readLine().replace(/\s+$/g, '').split(' ');
47
48     const n = parseInt(firstMultipleInput[0], 10);
49     const k = parseInt(firstMultipleInput[1], 10);
```

Line: 57 Col: 1

You have earned 25.00 points!  
You are now 93.3 points away from the gold level for your problem solving badge.

75% 756.7850

**HackerRank** | Prepare | Algorithms | Implementation | Marissa and Stones

**Problem**

•  $int$ : all possible values of the last stone, sorted ascending

**Input Format**

The first line contains an integer  $T$ , the number of test cases.

Each test case contains 3 lines:

- The first line contains  $n$ , the number of non-zero stones found.
- The second line contains  $a$ , one possible difference.
- The third line contains  $b$ , the other possible difference.

**Constraints**

- $1 \leq T \leq 10$
- $1 \leq n, a, b \leq 10^3$

**Sample Input**

```
STDIN      Function
-----
2          T = 2 (test cases)
3          n = 3 (test case 1)
1          a = 1
2          b = 2
4          n = 4 (test case 2)
10         a = 10
100        b = 100
```

**Sample Output**

```
2 3 4
30 120 210 300
```

**Explanation**

With differences 1 and 2, all possible series for the first test case are given below:

- 0, 1, 2
- 0, 1, 3
- 0, 2, 3
- 0, 2, 4

**Congrats!**

You have earned your 5th star.

**Congratulations**

You solved this challenge. Would you like to challenge your friends?

Problem

Submissions

Leaderboard

Discussions

Editorial

Your local library needs your help! Given the expected and actual return dates for a library book, create a program that calculates the fine (if any). The fee structure is as follows:

1. If the book is returned on or before the expected return date, no fine will be charged (i.e.:  $fine = 0$ ).
2. If the book is returned after the expected return day but still within the same calendar month and year as the expected return date,  $fine = 15 \text{ Hackos} \times$  (the number of days late).
3. If the book is returned after the expected return month but still within the same calendar year as the expected return date, the  $fine = 500 \text{ Hackos} \times$  (the number of months late).
4. If the book is returned after the calendar year in which it was expected, there is a fixed fine of 10000 Hackos.

Charges are based only on the least precise measure of lateness. For example, whether a book is due January 1, 2017 or December 31, 2017, if it is returned January 1, 2018, that is a year late and the fine would be 10,000 Hackos.

**Example**

$d1, m1, y1 = 14, 7, 2018$   
 $d2, m2, y2 = 5, 7, 2018$

The first values are the return date and the second are the due date. The years are the same and the months are the same. The book is 14 - 5 = 9 days late. Return  $9 \times 15 = 135$ .

**Function Description**

Complete the libraryFine function in the editor below.

libraryFine has the following parameter(s):

- $d1, m1, y1$ : returned date day, month and year, each an integer
- $d2, m2, y2$ : due date day, month and year, each an integer

**Returns**

- int: the amount of the fine or 0 if there is none

**Input Format**

The first line contains 3 space-separated integers,  $d1, m1, y1$ , denoting the respective *day, month, and year* on which the book was returned.

The second line contains 3 space-separated integers,  $d2, m2, y2$ , denoting the respective *day, month, and year* on which the book was due to be returned.

**Constraints**

- $1 \leq d1, d2 \leq 31$
- $1 \leq m1, m2 \leq 12$
- $1 \leq y1, y2 \leq 10^4$

Change Theme Language JavaScript (Node.js)

```
1 'use strict';
2
3 const fs = require('fs');
4
5 process.stdin.resume();
6 process.stdin.setEncoding('utf-8');
7
8 let inputString = '';
9 let currentLine = 0;
10
11
12 process.stdin.on('data', function(inputStdin) {
13     inputString += inputStdin;
14 });
15
16 process.stdin.on('end', function() {
17     inputString = inputString.split('\n');
18     main();
19 });
20
21 function readline() {
22     return inputString[currentLine++];
23 }
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Line: 48 Col: 2

Upload Code as File Test against custom input Run Code Submit Code

HackerRank | Prepare Algorithms Implementation Library Fine

Exit Full Screen View

Problem

Submissions

Leaderboard

Discussions

Editorial

Your local library needs your help! Given the expected and actual return dates for a library book, create a program that calculates the fine (if any). The fee structure is as follows:

1. If the book is returned on or before the expected return date, no fine will be charged (i.e.:  $fine = 0$ ).
2. If the book is returned after the expected return day but still within the same calendar month and year as the expected return date,  $fine = 15 \text{ Hackos} \times$  (the number of days late).
3. If the book is returned after the expected return month but still within the same calendar year as the expected return date, the  $fine = 500 \text{ Hackos} \times$  (the number of months late).
4. If the book is returned after the calendar year in which it was expected, there is a fixed fine of 10000 Hackos.

Charges are based only on the least precise measure of lateness. For example, whether a book is due January 1, 2017 or December 31, 2017, if it is returned January 1, 2018, that is a year late and the fine would be 10,000 Hackos.

**Example**

$d1, m1, y1 = 14, 7, 2018$   
 $d2, m2, y2 = 5, 7, 2018$

The first values are the return date and the second are the due date. The years are the same and the months are the same. The book is 14 - 5 = 9 days late. Return  $9 \times 15 = 135$ .

**Function Description**

Complete the libraryFine function in the editor below.

libraryFine has the following parameter(s):

- $d1, m1, y1$ : returned date day, month and year, each an integer
- $d2, m2, y2$ : due date day, month and year, each an integer

**Returns**

- int: the amount of the fine or 0 if there is none

**Input Format**

The first line contains 3 space-separated integers,  $d1, m1, y1$ , denoting the respective *day, month, and year* on which the book was returned.

The second line contains 3 space-separated integers,  $d2, m2, y2$ , denoting the respective *day, month, and year* on which the book was due to be returned.

**Constraints**

- $1 \leq d1, d2 \leq 31$
- $1 < m1, m2 < 12$
- $1 \leq y1, y2 \leq 10^4$

38 function libraryFine(d1, m1, y1, d2, m2, y2) {
39 if (y1 < y2 || (y1 === y2 && m1 < m2) || (y1 === y2 && m1 === m2 && d1 < d2)) {
40 return 0;
41 } else if (y1 > y2) {
42 return 10000;
43 } else if (m1 > m2) {
44 return 500 \* (m1 - m2);
45 } else {
46 return 15 \* (d1 - d2);
47 }
48 }
49
50
51 function main() {
52 const ws = fs.createWriteStream(process.env.OUTPUT\_PATH);
53
54 const firstMultipleInput = readline().replace(/ /g, '').split(' ');
55
56 const d1 = parseInt(firstMultipleInput[0], 10);
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

Line: 48 Col: 2

Upload Code as File Test against custom input Run Code Submit Code

5 Star Hacker Rating

You have earned 15,000 points!  
You are now 1888.3 points away from the 6th star for your problem solving badge.

12% 101,372,200

Congratulations  
You solved this challenge. Would you like to challenge your friends?

Next Challenge

Test case 0 Compiler Message

Problem

Submissions

Leaderboard

Discussions

Editorial

Function Description

Complete the libraryFine function in the editor below.

libraryFine has the following parameter(s):

- $d1, m1, y1$ : returned date day, month and year, each an integer
- $d2, m2, y2$ : due date day, month and year, each an integer

Returns

- int: the amount of the fine or 0 if there is none

Input Format

The first line contains 3 space-separated integers,  $d1, m1, y1$ , denoting the respective *day*, *month*, and *year* on which the book was returned.

The second line contains 3 space-separated integers,  $d2, m2, y2$ , denoting the respective *day*, *month*, and *year* on which the book was due to be returned.

Constraints

- $1 \leq d1, d2 \leq 31$
- $1 \leq m1, m2 \leq 12$
- $1 \leq y1, y2 \leq 3000$
- It is guaranteed that the dates will be valid Gregorian calendar dates.

Sample Input

9 8 2015  
6 6 2015

Sample Output

45

Explanation

Given the following dates:  
Returned  $d1 = 9, m1 = 8, y1 = 2015$   
Due:  $d2 = 6, m2 = 6, y2 = 2015$   
Because  $y2 \geq y1$ , we know it's less than a year late.  
Because  $m2 \geq m1$ , we know it's less than a month late.  
Because  $d2 < d1$ , we know that it was returned late (but still within the same month and year).  
For the library's fee structure, we know that our fine will be  $15 \text{ Hackos} \times (\# \text{ days late})$ . We then print the result of  $15 \times (d1 - d2) = 15 \times (9 - 6) = 45$  as our output.

Change Theme

Language

JavaScript (Node.js)

Exit Full Screen View

```
1 // Complete the libraryFine function in the editor below.
2 // libraryFine has the following parameter(s):
3 // 1. INTEGER d1
4 // 2. INTEGER m1
5 // 3. INTEGER y1
6 // 4. INTEGER d2
7 // 5. INTEGER m2
8 // 6. INTEGER y2
9 //
10 // Returns
11 // int: the amount of the fine or 0 if there is none
12
13 function libraryFine(d1, m1, y1, d2, m2, y2) {
14     if (y1 < y2 || (y1 === y2 && m1 < m2) || (y1 === y2 && m1 === m2 && d1 < d2)) {
15         return 0;
16     } else if (y1 > y2) {
17         return 10000;
18     } else if (m1 > m2) {
19         return 500 * (m1 - m2);
20     } else {
21         return 15 * (d1 - d2);
22     }
23 }
24
25 function main() {
26     const ws = fs.createWriteStream(process.env.OUTPUT_PATH);
27
28     const firstMultipleInput = readLine().replace(/\s+$/, '').split(' ');
29     const d1 = parseInt(firstMultipleInput[0], 10);
30     const m1 = parseInt(firstMultipleInput[1], 10);
31     const y1 = parseInt(firstMultipleInput[2], 10);
32
33     const secondMultipleInput = readLine().replace(/\s+$/, '').split(' ');
34     const d2 = parseInt(secondMultipleInput[0], 10);
35     const m2 = parseInt(secondMultipleInput[1], 10);
36     const y2 = parseInt(secondMultipleInput[2], 10);
37
38     const result = libraryFine(d1, m1, y1, d2, m2, y2);
39
40     ws.write(result + '\n');
41     ws.end();
42 }
```

Line: 48 Col: 2

Upload Solution File

Test against custom input

Run Code

Submit Code

Prepare

Certify

Compete

Apply

Search

793.3 more points to get your next star!

Rank: 130353 | Points: 1226.7/2200

Algorithms

Data Structures

Compare the Triplets

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 95.96%

Solve Challenge

Solve Me First

Easy, Problem Solving (Basic), Max Score: 1, Success Rate: 97.64%

Solved

Simple Array Sum

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 94.67%

Solved

A Very Big Sum

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 95.94%

Solved

Diagonal Difference

Easy, Problem Solving (Basic), Max Score: 10, Success Rate: 95.17%

Solved

STATUS

☐ Solved

☐ Unsolved

SKILLS

☐ Problem Solving (Intermediate)

☐ Problem Solving (Advanced)

☐ Problem Solving (Basic)

DIFFICULTY

☐ Easy

☐ Medium

☐ Hard

SUBDOMAINS

☐ Warmup

☐ Implementation

☐ Strings

☐ Sorting

973.3 more points to get your next star!

Rank: 130353 | Points: 1226.7/2200