

AVAILABLE
LESSONS:*Lesson 1*
Iterations*Lesson 2*
Arrays*Lesson 3*
Time Complexity*Lesson 4*
Counting
Elements*Lesson 5*
Prefix Sums*Lesson 6*
Sorting*Lesson 7*
Stacks and
Queues*Lesson 8*
Leader*Lesson 9*
Maximum slice
problem*Lesson 10*
Prime and
composite
numbers*Lesson 11*

RESPECTABLE

MinMaxDivision

START

Divide array A into K blocks and minimize the largest sum of any block.

Programming language: C++

You are given integers K, M and a non-empty zero-indexed array A consisting of N integers. Every element of the array is not greater than M.

You should divide this array into K blocks of consecutive elements. The size of the block is any integer between 0 and N. Every element of the array should belong to some block.

The sum of the block from X to Y equals $A[X] + A[X + 1] + \dots + A[Y]$. The sum of empty block equals 0.

The *large sum* is the maximal sum of any block.

For example, you are given integers K = 3, M = 5 and array A such that:

A[0] = 2
A[1] = 1
A[2] = 5
A[3] = 1
A[4] = 2
A[5] = 2
A[6] = 2

The array can be divided, for example, into the following blocks:

- [2, 1, 5, 1, 2, 2, 2], [], [] with a large sum of 15;
- [2], [1, 5, 1, 2], [2, 2] with a large sum of 9;
- [2, 1, 5], [], [1, 2, 2, 2] with a large sum of 8;
- [2, 1], [5, 1], [2, 2, 2] with a large sum of 6.

The goal is to minimize the large sum. In the above example, 6 is the minimal large sum.

Write a function:

Sieve of
Eratosthenes

Lesson 12

Euclidean
algorithm

Lesson 13

Fibonacci
numbers

Lesson 14

**Binary search
algorithm**

Lesson 15

Caterpillar
method

Lesson 16

Greedy
algorithms

Lesson 17

Dynamic
programming

Lesson 90

Tasks from
Indeed Prime
2015 challenge

Lesson 91

Tasks from
Indeed Prime
2016 challenge

Lesson 92

Tasks from
Indeed Prime
2016 College
Coders
challenge

Lesson 99

```
int solution(int K, int M, vector<int> &A);
```

that, given integers K, M and a non-empty zero-indexed array A consisting of N integers, returns the minimal large sum.

For example, given K = 3, M = 5 and array A such that:

A[0] = 2

A[1] = 1

A[2] = 5

A[3] = 1

A[4] = 2

A[5] = 2

A[6] = 2

the function should return 6, as explained above.

Assume that:

- N and K are integers within the range [1..100,000];
- M is an integer within the range [0..10,000];
- each element of array A is an integer within the range [0..M].

Complexity:

- expected worst-case time complexity is $O(N \cdot \log(N+M))$;
- expected worst-case space complexity is $O(1)$, beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Copyright 2009–2016 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

**For
programmers**

Lessons Challenges
Terms FAQ

For companies

Tour About
Pricing us
Blog Jobs
Privacy Terms
Cookies
API

**Sign up for our
newsletter:**

Information about
upcoming challenges,
solutions and lessons
directly in your inbox.

Social:

[f](#) [t](#) [in](#)

Contact us:

For customer support
queries:

UK +44 (0) 208 970
78 68

US 1-415-466-8085
support@codility.com

For sales queries:

UK +44 (0) 208 970
78 67

US 1-415-466-8085
sales@codility.com

© 2009–2016 Codility Ltd., registered in England and Wales (No. 7048726). VAT ID
GB981191408. Registered office: 107 Cheapside, London EC2V 6DN