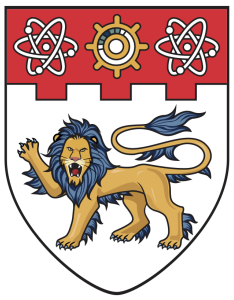


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Nanyang Programming Contest 2025

Stage 1: The First Contest

15 March 2025
2:00PM to 4:00PM (Contest Time)
Software Project Lab

This problem set should contain 8 problems on 17 numbered pages.

NPC2025

Problem A: Lucky Seven

Time Limit: 1 second
Memory Limit: 1.5 GB

Problem Description

Bob considers the number 7 lucky. As a result, he believes that the 7-th letter he sees on a day is his *lucky letter* of the day.

You are given a string s of length 10, denoting the first 10 letters Bob saw today. What is Bob *lucky letter*?

I/O Format

Input Format

The only line of input contains a string s, of length 10.

Output Format

Print a single character: Bob's lucky letter.

Function Format

Alternative to standard I/O, you may implement the function: `solve(s)`.

Function Parameters

- s (string of length 10)

Function Return

A character representing Bob's lucky letter

Constraint

- s has a length of 10
- s contains only lowercase Latin letters (i.e, the characters "a" to "z" (without the quotes))

Samples

Standard Input	Standard Output
`proceeding`	`d`
`outofsight`	`i`

Explanation for Sample 1:

The 7-th character of ``proceeding'' is 'd', and hence that is Bob's lucky letter.

Explanationa For Sample 2:

The 7-th character of ``outofsight'' is 'i', and hence that is Bob's lucky letter.

Remark

The problem **does not** have partial scoring.

Language	Instruction
Java	In submitting your solution, you may name your main class as anything

Problem B: Distance

Time Limit: 2 sec (C++, C), 3 sec (Pypy3), 4 sec (Java), 10 sec (Python3)
Memory Limit: 512 MB (C++, C, Pypy3, Python3), 2048 MB (Java)

Problem Description

There are n distincts positions in a straight line denoted as $x_1, x_2, \ldots, x_n$. We want to choose k positions out of the n positions such that the minimum distance between any two of the k positions is maximized. Print the largest minimum distance.

I/O Format

Input Format

The first line contains two space-separated integers, n and k. The next line contains n space-separated integers, where the i-th integer represents x_i.

```
n k
x1 x2 .  .  .  xn
```

Output Format

The only line contains an integer, which is the answer to the problem.

Function Format

Alternatively to standard I/O, you may implement the following function: `solve(n, k, x)`.

Function Parameters

- n (integer)
- k (integer)
- x (lists/array of integers): Array of n integers

Function Return

Integer: answer to the problem

Constraint

- $2 \leq n \leq 10^5$
- $0 \leq x_i \leq 10^9$
- $2 \leq k \leq n$

Samples

Standard Input	Standard Output
5 3 1 2 8 4 9	3

Explanation for Sample 1:

The optimal k position is {1,4,9}. The minimum distance between any two positions is $4-1 = 3$.

Remark

Language	Instruction
Java	In submitting your solution, please name your main class as **Solution**

Problem C: Graph

Time Limit: 2 seconds
Memory Limit: 1024 MB

Problem Description

Given a simple undirected graph with n vertices (vertices are numbered from 1 to n) and m edges (edge i is connecting vertex a_i and vertex b_i), k guards are placed on distinct vertices p_1, p_2, ..., p_K. Each guard i has stamina h_i. A vertex v is guarded if there exists at least one guard i such that the shortest path distance (the distance is calculated by the number of edges on the path) between v and p_i is at most h_i. Output all guarded vertices in ascending order.

I/O Format

Input Format

The first line contains three space-seperated integers, n, m, k
The next m lines contain two space-seperated integers, where the next i line contains a_i, b_i, which denote the vertices (1-based) the edge i connects with.
The next k lines contain two space-separated integers, where the next i line contains p_i, h_i, which denote the starting position of guard i and its stamina, respectively.

```
n k m
a₁ b₁
a₂ b₂
⋮
aₘ bₘ
p₁ h₁
p₂ h₂
⋮
pₘ hₘ
```

Output Format

The first line contains an integer g denoting the number of guarded vertices. The next line contains g space-separated integers where each integer is the vertex number (1-based) of guarded vertices in ascending orders.

```
g
v₁ v₂ ... vg
```

Function Format

Alternatively to standard I/O, you may implement the function: `solve(n, m, k, graph, guard)`.

Function Parameters

Each language may have a slightly different format. However, in general, the following parameters apply to all languages.

- n (Integer): the number of vertices
- m (Integer): the number of edges
- k (Integer): the number of guards
- *graph* (Adjacency List representation of the graph)

- *guard* (List of two integers): The first integer is p_i and the second integer is h_i

Function Return

A list of integers contains all the guarded vertices. (You do not need to return the number of guard vertices because our template can get the information from the list).

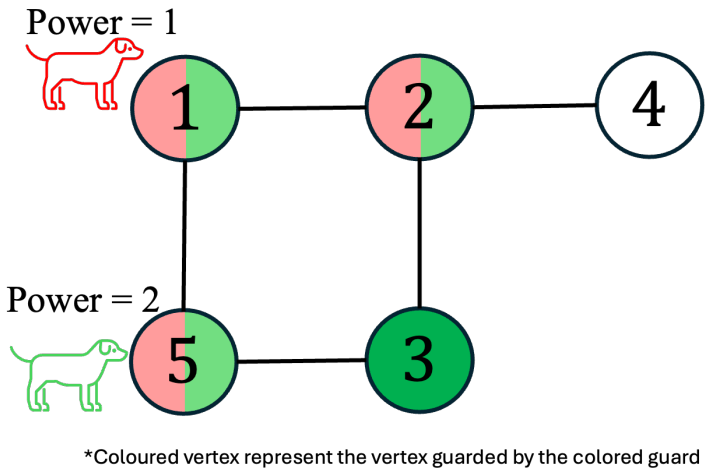
Constraint

- $1 \leq N \leq 2 \times 10^5$
- $0 \leq M \leq \min\left(\frac{N(N-1)}{2}, 2 \times 10^5\right)$
- $1 \leq K \leq N$
- $1 \leq a_i, b_i \leq N$
- The given graph is simple.
- $1 \leq p_i \leq N$
- All p_i are distinct.
- $1 \leq h_i \leq N$
- All input values are integers.

Samples

Standard Input	Standard Output
5 5 2 1 2 2 3 2 4 3 5 1 5 1 1 5 2	4 1 2 3 5
3 0 1 2 3	1 2
10 10 2 2 1 5 1 6 1 2 4 2 5 2 10 8 5 8 6 9 6 7 9 3 4 8 2	7 1 2 3 5 6 8 9

Explanation for Sample 1:



*Coloured vertex represent the vertex guarded by the colored guard

The guarded vertices are $1, 2, 3, 5$. These vertices are guarded because of the following reasons.

- The distance between vertex 1 and vertex $p_1 = 1$ is 0, which is not greater than $h_1 = 1$. Thus, vertex 1 is guarded.
- The distance between vertex 2 and vertex $p_1 = 1$ is 1, which is not greater than $h_1 = 1$. Thus, vertex 2 is guarded.
- The distance between vertex 3 and vertex $p_2 = 5$ is 1, which is not greater than $h_2 = 2$. Thus, vertex 3 is guarded.
- The distance between vertex 5 and vertex $p_1 = 1$ is 1, which is not greater than $h_1 = 1$. Thus, vertex 5 is guarded.

Explanationa For Sample 2:
The given graph may have no edges.

Remark

Language	Instruction
Java	In submitting your solution, please name your main class as **Main**

Problem D: Multiset

Time Limit: 2 sec (C++, C), 3 sec (Pypy3), 4 sec (Java), 10 sec (Python3)
Memory Limit: 512 MB (C++, C, Pypy3, Python3), 2048 MB (Java)

Problem Description

Given two multisets (i.e., unordered and possibly containing duplicate elements) that consist of n integers, $X = \{x_0, x_1, \ldots, x_{n-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{n-1}\}$. You can perform the following operation on set X:

- Choose two positions i, j, where $0 \leq i, j \leq n-1$ and $i \neq j$. Then, decrease x_i by 1 and increase x_j by 1.

Find the minimum number of operations you must perform so that X is equal to Y (i.e., both sets contain the same exact values, and the **order doesn't matter**); if it is not possible, print -1 instead.

I/O Format

Input Format

The first line contains a single integer, n.
The second line contains n space-separated integers describing the respective values of set X.
The third line contains n space-separated integers describing the respective values of set Y.

```
n
x₁ x₂ ... xₙ
y₁ y₂ ... yₙ
```

Output Format

The only line contains a single integer denoting the minimum number of operations required to make set X equal to set Y; if no number of operations will ever make the two sets equal, print -1 instead.

Function Format

Alternatively to standard I/O, you may implement the function: `solve(n, x, y)`.

Function Parameters

- n (Integer): The number of elements in the set
- x (list/array of integers): The multiset X
- y (list/array of integers): The multiset Y

Function Return

Integer (Answer to the problem)

Constraint

- $1 \leq n \leq 10^5$
- $-10^9 \leq x_i, y_i \leq 10^9$, where $0 \leq i < n$

Samples

Standard Input	Standard Output
3 1 2 3 -1 4 3	2
3 1 2 3 2 3 2	-1

Explanation for Sample 1:
In this example, we perform two operations:

1. $\mathbf{1,2}, 3 \rightarrow \mathbf{0,3}, 3$
2. $\mathbf{0,3}, 3 \rightarrow \mathbf{-1,4}, 3$

Explanationa For Sample 2:
In this example, set X and set Y can never be the same with any amount of operations.

Remark

Please use 64-bit integer representation for C++, C or Java to avoid integer overflow.

Language	**Instruction**
Java	In submitting your solution, please name your main class as **Solution**

Problem E: Operations

Time Limit: 2 sec (C++, C), 3 sec (Pypy3), 4 sec (Java), 10 sec (Python3)
Memory Limit: 512 MB (C++, C, Pypy3, Python3), 2048 MB (Java)

Problem Description

You are given a number x, initially $x = 0$. You can perform the following three operations on x:

1. $x \leftarrow x + a$, where a is a given integer
2. $x \leftarrow x + b$, where b is a given integer
3. $x \leftarrow \lfloor \frac{x}{2} \rfloor$, where $\lfloor x \rfloor$ is the floor function, which represents the greatest integer y such that $y \leq x$.

Operations 1 and 2 can be performed any number of times (including zero times), while operation 3 can be performed at most once. You can execute these operations in any order. However, **at any point** x cannot exceeds a certain limit, t.

Your task is to maximize x after the execution of operations. Print the maximum x.

I/O Format

Input Format

The single line contains three space-separated integers: t, a, and b.

Output Format

A single integer, representing the maximum x you can achieve.

Function Format

Alternative to standard I/O, you may implement the function: `solve(t, a, b)`.

Function Parameters

- t (integer), the maximum limit of x
- a (integer)
- b (integer)

Function Return

Integer: Answer to the problem

Constraint

- $t \leq 5 \times 10^6$
- $1 \leq a, b \leq t$

Samples

Standard Input	Standard Output
8 5 6	8
2 3 4	0

Explanation for Sample 1:

In this example, we perform the following operations:

1. Operation 2: $x \leftarrow 0 + 6 = 6$
2. Operation 3: $x \leftarrow \lfloor \frac{6}{2} \rfloor = 3$
3. Operation 1: $x \leftarrow 3 + 5 = 8$

Explanationa For Sample 2:
You cannot perform any of the operations 1 and 2 because these operation will make x to be greater than t, which is 2. Operation 3 do not increase the x. Thus, the maximum achieveable x is 0.

Remark

Language	Instruction
Java	In submitting your solution, please name your main class as **Solution**

Problem F: Subseqeunce

Time Limit: 2 seconds
Memory Limit: 1024 MB

Problem Description

Given an array consisting of n integers. Your task is to find a subsequence of length k with the smallest difference between the maximum and minimum values in the subsequence. Print the value of the minimal difference in the subsequence.

Remark: A subsequence of an array a is constructed by removing some elements (possibly none) in a without changing the order of the rest of the elements.

I/O Format

Input Format

The first line of the input contains two space-seperated integers, n and k. The next n line contains one integer, where the next i-th line contains the the value a_i.

```
n k
a_1
a_2
⋮
a_n
```

Output Format

Print only a single line which contains the answer to the problem.

Function Format

Alternatively to standard I/O, you may implement the function: `solve(n, k, a)`.

Function Parameters

- n (integer): number of elements in the array a
- k (integer): length of the subsequence.
- a_i (Array/List of n integers): The array a

Function Return

Integer: Answer to the problem.

Constraint

- $2 \leq k \leq n \leq 10^5$
- $1 \leq a_i \leq 10^9$
- a_i is an integer.

Samples

Standard Input	Standard Output
5 3 11 15 10 14 12	2
5 3 5 7 5 7 7	0

Explanation for Sample 1:
The optimal subsequence with the first, third and fifth integers, $max = 12, min = 10$, so the minimum difference is 2.

Explanationa For Sample 2:
The optimal subsequence consists of the three 7. Therefore the minimum difference is 0.

Remark

Language	Instruction
Java	In submitting your solution, please name your main class as **Main**

Problem G: Tree

Time Limit: 2 sec (C++, C), 3 sec (Pypy3), 4 sec (Java), 10 sec (Python3)
Memory Limit: 512 MB (C++, C, Pypy3, Python3), 2048 MB (Java)

Problem Description

Given a tree with n vertices rooted at vertex 1. For each $k \in [0, n)$, print the number of vertices with exactly k children.

Remark: A rooted tree is a tree with a designated root node, where each node (except the root) has a unique parent. The edges define a parent-child relationship.

I/O Format

Input Format

The first line contains a single integer, n. The second line contains $n-1$ space-separated integers, where the i-th integer describe the parent of vertex $i+1$.

```
n
p₂ p₃ ⋯ pₙ
```

where p_i is the parent of node i.

Output Format

The only line contains n space-seperated integers, where the i-th integer is the answer for $k = i - 1$.

```
ans₀ ans₁ ... ansₙ₋₁
```

where ans_k is the answer to k.

Function Format

Alternatively to standard I/O, you may implement the function: `solve(n, parent)`.

Function Parameters

- n (integer): The number of nodes
- *parent* (array/list of $n-1$ integers): Array with length $n-1$, where the i-th integer describe the parent of vertex $i+1$.

Function Return

List/Array of n integers: The i-th integer contains the answer for $k = i - 1$.

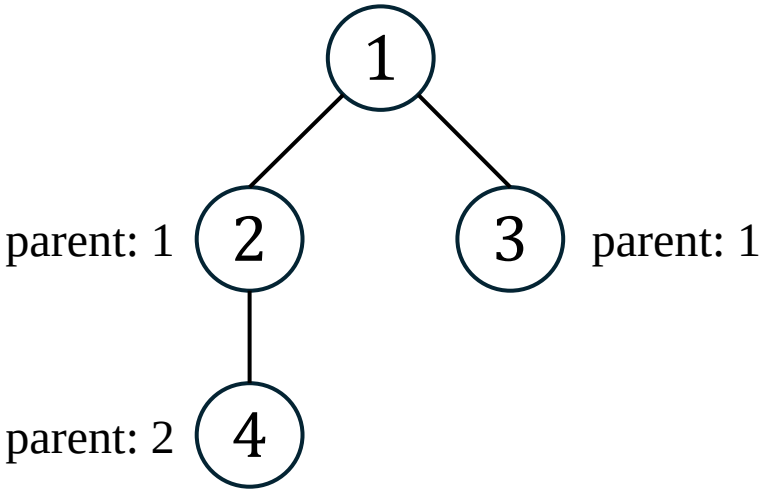
Constraint

- $2 \leq n \leq 10^6$

Samples

Standard Input	Standard Output
4 1 1 2	2 1 1 0

Explanation for Sample 1:



Node 3 and Node 4 have zero children, Node 2 have 1 child and node 1 have 2 child.

Remark

Language	Instruction
Java	In submitting your solution, please name your main class as **Solution**

Problem H: Bye Bye

Time Limit: 2 seconds
Memory Limit: 256 MB

Problem Description

There are n apartments along a number line, numbered 1 through n. Apartment i is located at coordinate x_i. Moreover, the office of NTU is located at coordinate s. Every employee of NTU lives in one of the n apartments. There are p_i employees who are living in Apartment i. All employees of NTU are now leaving the office all together. Since they are tired from work, they would like to get home by the NTU's shuttle bus. NTU owns only one bus, but it can accommodate all the employees. This bus will leave coordinate s with all the employees and move according to the following rule:

- Everyone on the bus casts a vote on which direction the bus should proceed, positive or negative. (The bus is autonomous and has no driver.) Each person has one vote, and abstaining from voting is not allowed. Then, the bus moves a distance of 1 in the direction with the greater number of votes. If a tie occurs, the bus moves in the negative direction. If there is an apartment at the coordinate of the bus after the move, all the employees who live there get off.
- Repeat the operation above as long as there is one or more employees on the bus.

For a specific example, see Sample Input 1. The bus takes one seconds to travel a distance of 1. The time required to vote and get off the bus is ignorable. Every employee will vote so that he himself/she herself can get off the bus at the earliest possible time. Strictly speaking, when a vote is taking place, each employee see which direction results in the earlier arrival at his/her apartment, assuming that all the employees follow the same strategy in the future. Based on this information, each employee makes the optimal choice, but if either direction results in the arrival at the same time, he/she casts a vote to the negative direction. Find the time the bus will take from departure to arrival at the last employees' apartment. It can be proved that, given the positions of the apartments, the numbers of employees living in each apartment and the initial position of the bus, the future movement of the bus is uniquely determined, and the process will end in a finite time.

I/O Format

Input Format

Input is given from Standard Input in the following format:

```
n s
x_1 p_1
x_2 p_2
⋮
x_n p_n
```

Output Format

Print the number of seconds the bus will take from departure to arrival at the last employees' apartment.

Function Format

Alternatively to standard I/O, you may implement the function: `solve(n, x, p)`.

Function Parameters

- n (integer): number of apartments
- x (list/array of n integers): position of the apartments
- p (list/array of n integers): number of employees stay at the apartments

Function Return

Integer: Answer to the problem.

Constraint

- $1 \leq n \leq 10^5$
- $1 \leq s \leq 10^9$
- $1 \leq x_1 \leq x_2 \leq \cdots \leq x_n \leq 10^9$
- $x_i \neq s$, where $1 \leq i \leq n$
- $1 \leq p_i \leq 10^9$, where $1 \leq i \leq n$
- All values in input are integers

Samples

Standard Input	Standard Output
3 2 1 3 3 4 4 2	4
6 4 1 10 2 1000 3 100000 5 1000000 6 10000 7 100	21
15 409904902 94198000 15017 117995501 7656764 275583856 313263626 284496300 356635175 324233841 607 360631781 148 472103717 5224 497641071 34695 522945827 816241 554305668 32 623788284 22832 667409501 124410641 876731548 12078 904557302 291749534 918215789 5	2397671583

Explanation for Sample 1:
Assume that the bus moves in the negative direction first. Then, the coordinate of the bus changes from 2 to 1, and the employees living in Apartment 1 get off. The movement of the bus after that is obvious: it just continues moving in the positive direction. Thus, if the bus moves in the negative direction first, the coordinate of the bus changes as $2 \to 1 \to 2 \to 3 \to 4$ from departure. The time it takes to get home for the employees living in Apartment 1, 2, 3 are 1, 3, 4 seconds, respectively.
Next, assume that the bus moves in the positive direction first. Then, the coordinate of the bus changes from 2 to 3, and the employees living in Apartment 2 get off. Afterwards, the bus starts heading to Apartment 1, because there are more employees living in Apartment 1 than in Apartment 3. Then, after arriving at Apartment 1, the bus heads to Apartment 3. Thus, if the bus moves in the positive direction first, the coordinate of the bus changes as $2 \to 3 \to 2 \to 1 \to 2 \to 3 \to 4$ from departure. The time it takes to get home for the employees living in Apartment 1, 2, 3 are 3, 1, 6 seconds, respectively.
Therefore, in the beginning, the employees living in Apartment 1 or 3 will try to move the bus in the negative direction. On the other hand, the employees living in Apartment 2 will try to move the bus in the positive direction in the beginning. There are a total of $3+2 = 5$ employees living in Apartment 1 and 3 combined, which is more than those living in Apartment 2, which is 4. Thus, the bus will move in the negative direction first, and the coordinate of the bus will change as $2 \to 1 \to 2 \to 3 \to 4$ from departure.

Explanationa For Sample 2:
Since the numbers of employees living in different apartments are literally off by a digit, the bus consistently head to the apartment where the most number of employees on the bus lives.

Remark

Please use 64-bit integer representation for C++, C or Java to avoid integer overflow.

Language	**Instruction**
Java	In submitting your solution, please name your main class as **Main**