A. Requirements

Code (100%)

You can write your code in Java, Python, C, or C++. The *time limit* may vary among different languages, depending on the performance of the language. Your code must be a complete excutable program instead of only a function. We guarantee test data strictly compliance with the requirements in the description, and you do not need to deal with cases where the input data is invalid.

No AI Assistance or Plagiarism: All code must be your own. The use of AI tools (e.g., ChatGPT, GitHub Copilot) or copying from external sources or peers is **strictly forbidden**.

Violations of the plagiarism rules will result in 0 points or even **failure** of this course.

Libraries in this assignment:

- For C/C++, you can only include standard library.
- For Java, you can only `import java.util.*`
- For Python, you can only import standard library. In other words, you cannot import libraries such as `numpy`.

We provide an example problem to illustrate the information above better.

B. Example Problem: A + B Problem

Description

Given 2 integers A and B, compute and print A + B

Input

Two integers in one line: A, and B

Output

One integer: A + B

Sample Input 1

```
1 2
```

Sample Output 1

```
3
```

Problem Scale & Subtasks

For 100% of the test cases, $0 \leq A, B \leq 10^6$

Solutions

Java

```java
import java.util.*;

public class Example {
    public static void main(String[] args) {
        int a, b;
        Scanner scanner = new Scanner(System.in);
        a = scanner.nextInt();
        b = scanner.nextInt();
        scanner.close();
```

```
        System.out.println(a + b);
    }
}
```

Python

```
AB = input().split()
A, B = int(AB[0]), int(AB[1])
print(A + B)
```

C

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  int A, B;
  scanf("%d%d", &A, &B);
  printf("%d\n", A + B);
  return 0;
}
```

C++

```
#include <iostream>

int main(int argc, char *argv[])
{
  int A, B;
  std::cin>> A >> B;
  std::cout<< A + B << std::endl;
  return 0;
}
```

C. Submission

After finishing this assignment, you are required to submit your code to the Online Judge System (OJ), and upload your `.zip` package of your code files to BlackBoard.

C.1 Online Judge

Once you have completed one problem, you can submit your code on the page on the Online Judge platform (`oj.cuhk.edu.cn`, campus only) to gain marks for the code part. You can submit your solution of one problem for **no more than 80 times**.

After you have submitted your program, OJ will test your program on all test cases and give you a grade. The grade of your latest submission will be regarded as the final grade of the corresponding problem. Each problem is tested on multiple test cases of different difficulty. You will get a part of the score even if your algorithm is not the best.

Note: The program running time may vary on different machines. Please refer to the result of the online judge system. OJ will show the time and memory limits for different languages on the corresponding problem page.

If you have other questions about the online judge system, please refer to OJ wiki (campus network only). If this cannot help you, feel free to contact us.

C.2 BlackBoard

You are required to upload your **source codes** to the BlackBoard platform. You need to name your files according to the following rules and compress them into `A3_<Student ID>.zip` :

```
A3_<Student ID>.zip
|-- A3_P1_<Student ID>.java/py/c/cpp
|-- A3_P2_<Student ID>.java/py/c/cpp
|-- A3_P3_<Student ID>.java/py/c/cpp
```

For Java users, **you don't need to consider the consistency of class name and file name**.

For example, suppose your ID is 123456789, and your problem 1 and 2 is written in `Python`, problem 3 is written in `Java` then the following contents should be included in your submitted `A3_123456789.zip`:

```
A3_123456789.zip
|-- A3_P1_123456789.py
|-- A3_P3_123456789.py
|-- A3_P3_123456789.java
```

C.3 Late Submissions

Submissions after Apr.21 2025 23:59:00(UTC+8) would be considered as LATE.

The LATE submission page will open after deadline on OJ.

Submisson time = max{latest submisson time for every problem, BlackBoard submisson time}

There will be penalties for late submission:

- 0–24 hours after deadline: final score = your score×0.8
- 24–72 hours after deadline: final score = your score×0.5
- 72+ hours after deadline: final score = your score×0

FAQs

Q: My program passes samples on my computer, but not get AC on OJ.

A: Refer to OJ Wiki Q&A

Authors

If you have questions for the problems below, please contact:

- Q1: Yihan Wang: 123090588@link.cuhk.edu.cn
- Q2: Hanjun Zheng: hanjunzheng@link.cuhk.edu.cn
- Q3: Wangmeiyu Zhang: wangmeiyuzhang@link.cuhk.edu.cn

CSC3100 Data Structures Spring 2025

Programming Assignment 3

Yihan Wang: 123090588@link.cuhk.edu.cn
Hanjun Zheng: hanjunzheng@link.cuhk.edu.cn
Wangmeiyu Zhang: wangmeiyuzhang@link.cuhk.edu.cn

Due: Apr.21 2025 23:59:00

Assignment Link: https://oj.cuhk.edu.cn/d/csc3100_2025_spring/homework/67eb58a2c07c7678dc21526e

Problem 1: Maximizing Customer Satisfaction (30% of this assignment)

Description

You are a product manager at a company responsible for the product line. Each product has two key attributes: **performance score** and **reliability rating**. You have two arrays of length n, `performance` $= [p_0, p_1, \ldots, p_{n-1}]$ and `reliability` $= [r_0, r_1, \ldots, r_{n-1}]$, representing the performance score and reliability rating of all the company's products, respectively.

You have received a series of customer inquiries. Each customer submits a request specifying their required **minimum performance score** and **minimum reliability rating**. These customer requests are given as a 2D array `requests`, where `requests`[i] = [min_performance$_i$, min_reliability$_i$].

For each customer request, you need to identify all products from the company's product line that meet the following criteria:

- The product's performance score is not less than the customer's minimum performance requirement.
- The product's reliability rating is not less than the customer's minimum reliability requirement.

Among all the products that satisfy these conditions, you need to find the one that maximizes the customer **satisfaction**. We define customer satisfaction as the sum of the product's performance score and reliability rating.

For each customer request, return the maximum customer satisfaction achievable by a suitable product. If no product can simultaneously meet both the performance and reliability requirements of a customer, return -1.

Return an array `answers`, where `answers`[i] is the answer to the i^{th} customer request.

Input Format

The first line contains two interger n,m representing the number of product and the number of customer. The following two lines represent `performance` and `reliability`. The next m lines represents m customer request with the format `min_performance`, `min_reliability` in each line.

Output Format

One array, where the i-th element represents the maximum customer satisfaction for the i-th request, or -1 if no product meets the requirements.

Sample Input and Output

Sample Input 1

```
4 3
4 3 1 2
2 4 9 5
4 1
1 3
2 5
```

Sample Output 1

```
6 10 7
```

Sample Input 2

```
3 3
3 2 5
2 3 4
4 4
3 2
1 1
```

Sample Output 2

```
9 9 9
```

Constraints

- `performance.length` == `reliability.length`
- `requests`[i].length == 2
- For all i, $1 \leq$ `performance`$[i] \leq 10^9$
- For all i, $1 \leq$ `reliability`$[i] \leq 10^9$
- For all i, $1 \leq \text{min\_performance}_i \leq 10^9$ (where `requests`[i] = [min_performance_i, min_reliability_i])
- For all i, $1 \leq \text{min\_reliability}_i \leq 10^9$ (where `requests`[i] = [min_performance_i, min_reliability_i])

Problem Scale & Subtasks

Test Case No.	Constraint
1-3	$1 \leq n \leq 10^2, 2 \leq m \leq 10^2$
4-6	$1 \leq n \leq 10^4, 2 \leq m \leq 10^4$
7-10	$1 \leq n \leq 10^5, 2 \leq m \leq 10^5$

Hint

You may use stack to solve this problem.

Problem 2: Trip Planning (30% of this assignment)

Description

Nadeshiko is planning a camping trip in Yamanashi. The region has N distinct campsites connected by $(N-1)$ bidirectional roads, each with a specific length. Among these campsites, D are designated as key campsites, while the rest are ordinary. Nadeshiko wants to know: for **every** campsite, determine the farthest distance to any **key** campsite. Write a program to help her.

Input Format

The first line contains two integers N and D. N represents the total number of campsites. D represents the number of key campsites. The campsites are denoted by numbers from 1 to N.

The second line contains D integers, represents all the key campsites.

Next $(N-1)$ lines: The i-th line of these $(N-1)$ lines contains two integers, u_i, w_i, represents that campsite u_i and campsite $(i+1)$ are connected by a road with length w_i.

Output Format

N lines, each line contains only one integer. The integer in the i-th line represents the farthest distance to any key campsites from campsite i.

Sample Input 1

```
5 2
2 4
1 2
1 3
1 3
2 4
```

Sample Output 1

```
3
5
6
5
9
```

Sample Input 2

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

Sample Output 2

```
5
4
3
3
4
5
5
4
```

Problem Scale & Subtasks

Test Case No.	Constraints
1-5	$1 \leq D \leq N \leq 1000$
6-8	$1 \leq D \leq 20 \leq N \leq 2 \times 10^5$
9-10	$D = N = 8 \times 10^5$
11-12	$1 \leq D \leq N = 9 \times 10^5, u_i = i$
13-20	$1 \leq D \leq N = 10^6$

For all test cases, there are $1 \leq D \leq N \leq 10^6$, $1 \leq u_i \leq i$ and $1 \leq w_i \leq 100$.

Problem 3: Mysterious BST (40% of this assignment)

Description

Lazy Xiaoming has built a Binary Search Tree (BST) in a mysterious world. Each node on that tree has an ID and a *value*, and their IDs and *value*s are unique. ID is an integer between 1 and n (assuming the BST has n nodes). To avoid ambiguity, we have discussed with Xiaoming and made a deal to use IDs to represent nodes.

Xiaoming stays in this mysterious world for Q days. Each day, lazy Xiaoming will:

- Option 1: forget to water the tree, which means one leaf node on that tree falls off;
- Option 2: water the tree, which means one node that has dropped before grows from its original position. The grown node is guaranteed to be a leaf node of the new tree.

At the end of each day, lazy Xiaoming will ask us the subtree size of the Kth **largest** (based on the nodes' *value*s) node on the tree.

Oh! What does the tree look like before the first day? Lazy Xiaoming is too lazy, so he can only tell us how to build the tree: Access each node from smallest to largest by ID. For the node with ID i, if:

- the current BST is empty, then set node i as the root;
- the current BST is not empty, then traverse the BST starting from the root.

  Specifically, assuming that we are at node x of BST,

  - if the *value* of node i is less than the *value* of node x, then go to x's left subtree;
  - otherwise, go to x's right subtree;

  Repeat the above steps until we find an empty position. Insert the node i into that empty position.

Input

The first line: an integer n, represents the number of nodes before the first day; and an integer Q, representing the number of days.

The second line: n integers, where the ith integer represents the *value* of the node with ID i. We guarantee that these n integers are **distinct** and were **randomly generated** from a uniform distribution over the range $[1, 10^{18}]$

Next q lines: each line contains 3 integers, op, id, and K. $op \in \{1, 2\}$, represents Xiaoming's option on that day. $1 \leq id \leq n$, represents the ID of the node that falls ($op = 1$) or grows($op = 2$). $1 \leq K \leq size$, represents Xiaoming's question, where $size$ is the size of BST on that day.

Output

Q lines, each line contains one integer, representing the answers to each question of the day.

Sample Input 1

```
5 7
16991 28739 5393 30404 1229
1 5 1
2 5 4
1 5 4
1 3 3
2 3 4
2 5 4
1 4 1
```

Sample Output 1

```
1
2
1
3
1
2
1
```

Sample Input 2

```
6 6
15969 17873 3371 3475 3017 10553
1 6 4
1 5 4
1 4 1
1 2 2
1 3 1
2 3 2
```

Sample Output 2

```
3
2
1
1
1
1
```

Problem Scale & Subtasks

Test Case No.	Constraint
1-3	$1 \leq Q \leq 2 \times 10^3, 2 \leq n \leq 2 \times 10^3$
4-5	$Q = 1, 2 \leq n \leq 5 \times 10^5$
6-7	$K = 1, 1 \leq Q \leq 5 \times 10^5, 2 \leq n \leq 5 \times 10^5$
8-10	$1 \leq Q \leq 5 \times 10^5, 2 \leq n \leq 5 \times 10^5$