MANIPAL UNIVERSITY JAIPUR

DATA STRUCTURES PROJECT LIST

III SEMESTER- CCE-B

1. **String Compression**

   Mr. X has come up with a new string compression algorithm. Consider a string of length N which contains up to K distinct characters. The compression algorithm works as follows: Replace each maximal contiguous substring containing only one distinct character (repeated an arbitrary number of times) and replace it by 2 values: the character and the length of the substring. For example, the string "aabbaaa" will be compressed to "a, 2, b, 2, a, 3". Thus the length of the compressed string is 6. Since Mr. X is living in advanced times, the length of any integer is considered to be 1. For example, if a string is compressed to "a, 111, b, 13", then its length after compression is considered to be 4. To test his algorithm, he needs to know the expected length of the compressed string for given N and K if the input string is randomly uniformly chosen from all possibilities. He wants to run this experiment multiple times for different N, K and needs your help.

2. **Friends and their Popularity**

   Sam created a new Facebook account and accepted all the random friend requests. After a while Sam decides to remove some friends from his friend list using the following algorithm to delete a friend.

```
Algorithm Delete(Friend):
   DeleteFriend=false
   for i = 1 to Friend.length-1
      if (Friend[i].popularity < Friend[i+1].popularity)
        delete i th friend
        DeleteFriend=true
        break
   if(DeleteFriend == false)
     delete the last friend
```

NOTE: Order of friends after deleting exactly K friends should be maintained as given in input. Each test case contains N, the number of friends Sam currently has and K, the number of friends Sam decides to delete. Next lines contain popularity of his friends separated by space. For each test case print N-K numbers which represent popularity of Sam friend's after deleting K friends.

3. **List Editing**

   Implement List data structure using Arrays in C. It should have the ability to store double precision real numbers in it. The program accepts a string as a command line argument. The string may contain any printable characters other than a white space. Of these, characters {a,d,p,A,D,P} will be called action characters. Let i be a position in the string where an action character is found, and let k be the first position k > i where another action character or a NULL (\0) is found. Your program should take the following action for each action character found in the input.

   a. Append: When an action character a or A is found, then (1) check if the substring between i + 1 and k − 1 forms a valid real number, and (2) if so append the real number to the List.

   b. Delete: When an action character d or D is found, then (1) check if the substring between i+ 1 and k −1 forms a valid integer, and (2) if so delete the item in that integer position from the List.

   c. Print: When an action character p or P is found, print the current items in the list. If the parsing of the real or integer number fails for append or delete, that particular action should not be performed. If the integer does not correspond to a valid position in the List for delete, that action should not be taken. Characters that appear before the first action character can be ignored as well.

4. **Arithmetic operations on Complex Numbers**

   You are required to implement a data structure for handling arithmetic operations on complex numbers. Specifically, the list of operations that should be supported by this data structure is as follows: 1. Adding 2 complex numbers 2. Subtracting 2 complex numbers 3. Multiplying 2 complex numbers 4. Division of one complex number by another x + iy = a + ib c + id 5. Exponentiation of one complex number by another x + iy = (a + ib) (c+id) The initial complex number value present in the program is 0+i0, and math operations are performed sequentially to update this value by using the Complex number object.

5. **Performance Checker (BSTsort & Quicksort)**

   This assignment requires you to perform sorting of a large list of numbers using an inorder traversal of a Binary Search Tree (BST) and using Quick sort, and compare their performances for input lists of different sizes. Performs sanity checks on whether the sorted lists returned by BST and Quick sort are indeed sorted or not. Assume that the input file will only have distinct numbers. There will be no duplicates. Discuss the computational and storage complexities of these two sorting approaches. Further, you should also validate if these complexities match the actual sorting time taken by the two approaches. For this, you will run experiments using different appropriate input sizes (including the three input files already given). You should submit a table with three columns: the input size, the time taken by BSTsort and time taken by QUICKsort.

6. **List Update and Cost**

   Tom purchased an array *A* Having *N* integer values. After playing it for a while, he got bored of it and decided to update value of its element. In one second he can increase value of each array element by *1*. He wants each array element's value to become greater than or equal to *K*. Please help Tom to find out the minimum amount of time it will take for him to do so.

7. **Arithmetic Expression Checker**

   This assignment requires you to check if a given arithmetic expression is well-formed or not, as defined below. Note that we will use the general term “bracket” to mean parenthesis (), curly braces {} or square brackets [], unless noted otherwise. 1. The expression only contains the following characters: three types of brackets {, [, (,),], }, the digits 0-9, the four operators +, -, *, / or a whitespace ‘ ’ 2. The number of left and right brackets of each type should be equal. 3. For each right bracket in the expression, the closest preceding unmatched left bracket should be of the same type. For example, the following are well-formed expressions: 15/[(2+3)-{4+5}-1] [(1 * (2 + 3) - 5) + (3 + (4 - 5) * 3)] Whereas, the following are not: “3^5” [([2+3)-4+5].

8. **Monk and Cursed BST 1**

   Monk has *N* distinct integers and a Binary Search Tree which is initially empty. He inserts all the elements in the BST in the order given. But wait! The tree formed turns out to be cursed. Monk is having some weird experiences since he made that tree. So, now to stop all that, Monk has two options, to destroy the BST or to pray to God and ask for a solution. Now since Monk has to use this BST in a Code Monk Challenge, he cannot destroy it. So he prays to God. God answer his prayers and sends an angel named Micro. Now, Micro tells him two values, *X* and *Y*, present in the BST and ask him to delete both. Monk finds that X is having left subtree, right subtree and Y is a leaf node. Write a program which helps monk to delete X and Y.

9. **Monk and Cursed BST 2**

   Monk has *N* distinct integers and a Binary Search Tree which is initially empty. He inserts all the elements in the BST in the order given. But wait! The tree formed turns out to be cursed. Monk is having some weird experiences since he made that tree. So, now to stop all that, Monk has two options, to destroy the BST or to pray to God and ask for a solution. Now since Monk has to use this BST in a Code Monk Challenge, he cannot destroy it. So he prays to God. God answer his prayers and sends an angel named Macro. Now, Macro tells him two values, *X* and *Y*, present in the BST and ask him to find the successor of both and delete. Monk finds that X is not having right subtree (but it is not the biggest value in BST), Y is having right subtree. Write a program which helps the monk to delete the successors of X and Y.

10. **Monk’s Array and Tree**

    Once Monk was watching a fight between an array and a tree, of being better. Tree got frustrated and converted that array into a Binary Search Tree by inserting the elements as nodes in BST, processing elements in the given order in the array. Now Monk wants to know the height H of the created Binary Search Tree and the number comparison to find the value X, present in the created Binary Search Tree. Help Monk for the same.

11. **Traffic count (No of vehicles on Road):**

My friends Tushar , Hemel and Shovon recently built a software named “Traffic Count!. Few days ago I asked them “Tell me something about your software.”. Shovon replied smartly “Oh! It’s an android software. Any person using this software, will know how many vehicles are present in the road at a certain time of period!” Being astonished I asked them “How do you calculate this?”. This time Hemel answered me, “Our software was pre-loaded with 10^7+1 types of vehicles and they are numbered from 0,1,2,……,10^7 . Whenever a user connects our software to the internet, the software receives datas from our server. Our server holds the last t (t ≤ 10 ^ 5) seconds data. Each second, an integer is shown which indicates a type of vehicle enters into our server. User can give any number of q (q ≤ 10^5) queries. For each query user need to insert 4 integers t1 , t2 , type1 , type2. Our software returns how many vehicles from type1 to type2 are present from time t1 to time t2. Now the main question is how does our server work? "But I am not telling you about this. You don't need to know about the theory”. Now I want to build this type of software as quick as possible. I know how the server works. But I don’t know how the software answers the user queries quickly? I need your help to find this.

12. **Chefland using trees**

Everybody's heard about Chefland — the digital country where dreams come true, where nobody hears about crime and where everyone can get Accepted without submitting solutions. Of course, Chefland has the shape of a tree — there are N servers and N-1 bidirectional channels connecting pairs of servers in such a way that there is exactly one path between each pair of servers.

The citizens in Chefland are living online; each citizen lives on exactly one server at any point in time. Don't be surprised, it's a futuristic scenario! Each channel connecting two servers has a value called ping. Each citizen can move to a different server through channels; moving through a channel with ping 2w takes w nanoseconds. Initially, there are ci citizens living on the i-th server (for each 1 ≤ i ≤ N).

Recently, Chef has created an ICO for Chefcoin — the cryptocurrency of the future in Chefland. Everyone in Chefland should receive exactly 1 Chefcoin. How to organize this process? Chefcoins can only be received on certain servers containing cryptoexchanges. Unfortunately, Chef can only create cryptoexchanges on two servers, since they consume a lot of electric power. The citizens want to receive Chefcoin as fast as possible; therefore, each citizen chooses the closest server with a cryptoexchange and moves to that server. Receiving Chefcoin at a cryptoexchange takes zero time.

Chef wants to minimize the sum of the times each citizen needs to spend moving to receive Chefcoin. The two servers containing cryptoexchanges can be chosen arbitrarily. Compute the minimum total time it takes Chefland's citizens to receive Chefcoin!

13. Chef’s Tree

Chef has a rooted tree, consisting of N vertexes. Each tree vertex has a unique index (integer from 1 to N), and also has a color (black or white). The root of Chef's tree has index 1. Initially, all vertexes of the tree are colored white. Then Chef start playing with the tree. He writes out a sequence of M vertexes of the tree V1, V2, ..., VM. And now he wants to make M operations with the tree, each of them is:

a. During i-th operation Chef picks vertex Vi.

b. If vertex Vi is black, then he colors it white.

c. Chef finds the farthest white vertex from vertex Vi, in case of tie he picks the one with largest index. This vertex is the result of current operation.

d. If vertex Vi wasn't recolored on step 2 of current operation (therefore before the operation it is white), Chef colors it black.

Chef gives to you the tree and sequence V. Help him to make all the operations. So, print the index of resulting vertex for each operation.

14. Checking of balanced parenthesis

You are given a character parenthesis array and an integer array.

You need to find the maximum sum sub-array in the integer array such that the corresponding sub-array in the character array has balanced parenthesis.

Formally, a balanced parentheses is subset of { [,],{,},<,>,(,) }∗ defined recursively as follows:

- The empty string is balanced parentheses.
- If A is balanced parentheses, then so are the strings [A], {A} , <A>, (A).
- If A and B are balanced parenthesis, then so is the string AB.

15. Garden management

Daniil is a royal gardener. He takes care of a garden with NN plants numbered 11 through NN. For each ii (1≤i≤N1≤i≤N), the initial height of the ii-th plant is AiAi. Unfortunately, the Queen doesn't like the garden, so she asked Daniil to cut some plants — in order to satisfy the Queen's request, the ii-th plant should have height BiBi (for each 1≤i≤N1≤i≤N).

Daniil is allowed to perform the following operation an arbitrary number of times (including zero):

a. Let's denote the current heights of plants by H1,H2,…,HNH1,H2,…,HN.
b. Choose two indices LL and RR (1≤L≤R≤N1≤L≤R≤N) and a new height hh such that h≤Hih≤Hi for each ii between LL and RR inclusive.
c. Cut plants LL through RR down to height hh, i.e. change the height of plant ii to hh for each L≤i≤RL≤i≤R.

Some time ago, Daniil was one of the best competitive programmers. Therefore, he is interested in the minimum number of operations needed to satisfy the Queen's request. Can you help him?

16. Generate Distinct Pairs from given sequences

Chef has two integer sequences A1,A2,…,ANA1,A2,…,AN and B1,B2,…,BMB1,B2,…,BM. You should choose N+M−1N+M−1 pairs, each in the form (Ax,By)(Ax,By), such that the sums Ax+ByAx+By are all pairwise distinct.

It is guaranteed that under the given constraints, a solution always exists. If there are multiple solutions, you may find any one.

17. Employee Project performance Management

To maintain Project performance of employee, data structure Created:

Student Structure

a. E id.

b. Name

c. 3 projects and corresponding employee project Hr. of each project (Project id, total Working hr., Project Hr., %age).{Hint 2D array}

d. Average Project Working %

Write a menu driven program for above data structure with separate and features:

Menu:

a. Insert new employee
b. Update Project hr. of project id for given Employee id.
c. Update Project hr. of all project for all Employee
d. Display Employee Information

Features:

a. Maintain each Employee information in single node of doubly singly linked list.

b. Start pointer always point to Largest % of Average project Hr. Always.

c. Insert Employee in already sorted list with proper location

d. After updating Project Hr. of Employee all node will be in Sort on basis of Average Project Hr. (Decreasing order)

e. Insert min 5 Employee information using menu option 1.

18. Student attendance management

To maintain attendance information of students, data structure Created:

Student Structure

a. Enrollment no.

b. Name

c. Mobile no.

d. 3 courses and corresponding student attendance of each course (total, Present, %age).{Hint 2D array} 5. Average attendance %

Write a menu driven program for above data structure with separate and features:

Menu:

a. Insert Student

b. Update attendance of given Enrollment no.

c. Update attendance of all student (mark Attendance)

d. Display Student Information

Features:

a. Maintain each student information in single node of circular singly linked list.

b. Start pointer always point to smallest % of attendance.

c. Insert student in already sorted list with proper location

d. After updating attendance of students all node will be in Sort on basis of Average attendance (increasing order)

e. Insert min 5 student information using menu option 1.

19. Best four Card game

Four players is playing Card game with following features:

a. Total 52 cards arranged randomly in main queue.

b. At starting, Each player maintain own queue.

c. Each player fetches one card at a time from main queue and insert it into own queue in circular fashion till main queue empty.

Winning strategy:

In any instance, If all color and all type of one card is available in player queue implies to winner. Otherwise restart game.

Card structure:

Color : (red/ Black)

Type: (clubs (♧), *diamonds* (◊), *hearts* (♥) and spades (♤))

Number (1to 13)

Write a menu driven program for above data structure with separate and features:

20. Two Card Match Problem game

Four players is playing Card game with following features:

a. Total 52 cards arranged randomly in main queue.

b. At starting, Each player maintain own queue.

c. Each player fetches one card at a time from main queue and insert it into own queue in circular fashion till main queue empty.

d. Now every player has 13 cards, they will push one card in stack (P1 to p4 repeatedly) one by one till all cards push.

Winning strategy:

In any instance, If two cards at the top of stack are same color and type (number like both red card of 10) implies to winner. Otherwise restart game.

Card structure:

Color : (red/ Black)

Type: (clubs (♧), *diamonds* (◊), *hearts* (♥) and spades (♤))

Number (1to 13)

Write a menu driven program for above data structure with separate and features:

21. Best BST Game

Four players are playing game with following features:

a. Total 100 random no. from range 100-999 in main queue.

b. At starting, Each player maintain own Binary search tree.

c. Each player fetches one no. at a time from main queue and insert it into own BST in circular fashion (one by one) till main queue empty.

d. Now every player has BST of 25 no..

Winning strategy:

The player with min. height BST implies to winner. Otherwise restart game.

Write a menu driven program for above data structure with separate and features:

22. Best AVL Tree Game

Four players are playing game with following features:

a. Total 100 random no. from range 100-999 in main stack.

b. At starting, each player maintains own AVL tree.

c. Each player fetches one no. at a time from main stack and insert it into own AVL tree in circular fashion (one by one) till main stack is empty.

d. Now every player has AVL of 25 no.

Winning strategy:

The player with highest sum value of all nodes key value of AVL tree implies to winner. Otherwise restart game.

Write a menu driven program for above data structure with separate and features:

23. Delivery man processes

A Delivery man processes request for orders through an automated machine at a shop which process each order in 10 min. For each order processed, he gets Rs. 10 credited to his account. However, there may come orders with priorities where time taken is defined as ceil (50/priority no.) min and priorities range from 1-6 with amount credited as Rs. 10 * priority no. But to earn maximum, the delivery man has to be smart enough to counter the constraints given as:

- Cannot process more than 5 orders i.e. the first 5 orders then the next five. However, the five orders in a go can have any permutation. For eg. for processes 1-10, process 1-5 will be processed first and then 6-10. However, 1-5 can have any sequence and so is for 6-10.
- Cannot remain at counter for more than 100 seconds or 10 orders whichever meets first in a shot. As a consequence of previous constraint, only 50 seconds to process 5 orders.
- If less than 10 orders are processed, Rs. 20 is deducted for each unprocessed order.

You have to help him out by designing the solution to the problem so that he could estimate in a shorter span of time that what should be the sequence of orders, so that he earns maximum in a single shot.

24. Student project management

To maintain attendance information of students, data structure Created:

Student Structure

a. Enrollment no.

b. Name

c. Mobile no.

d. Project name

e. Faculty assigned

f. Marks (mid presentation, final presentation, guide marks)

Write a menu driven program for above data structure with separate and features:

Menu:

a. Insert Student

b. Update project info of given Enrollment no.

c. Update marks of all student (mark individually)

d. Display Student Information

Features:

a. Maintain each student information in single node of circular singly linked list.

b. Start pointer always point to smallest total marks.

c. After updating marks of students all node will be in Sort on basis of Average attendance (increasing order)

d. Insert min 5 student information using menu option 1.

25. Best king Card game

Four players is playing Card game with following features:

a. Total 52 cards arranged randomly in main queue.

b. At starting, Each player maintain own queue.

c. Each player fetches one card at a time from main queue and insert it into own queue in circular fashion till main queue empty.

Winning strategy:

In any instance, If all color and all type of king card is available in player queue implies to winner. Otherwise restart game.

Card structure:

Color : (red/ Black)

Type: (clubs (♧), *diamonds* (◊), *hearts* (♥) and spades (♤))

Number (1to 13)

Write a menu driven program for above data structure with separate and features:

26. Best Card Problem game

Four players is playing Card game with following features:

a. Total 52 cards arranged randomly in main queue.

b. At starting, Each player maintain own queue.

c. Each player fetches one card at a time from main queue and insert it into own queue in circular fashion till main queue empty.

d. Now every player has 13 cards, they will push one card in stack (P1 to p4 repeatedly) one by one till all cards push.

Winning strategy:

At last, Player with Best card at the top of stack implies to winner. Otherwise restart game.

Card structure:

Color : (red/ Black)

Type: (clubs (♧), *diamonds* (◊), *hearts* (♥) and spades (♤))

Number (1to 13)

Write a menu driven program for above data structure with separate and features:

27. Best MHT Game

Four players are playing game with following features:

a. Total 100 random no. from range 100-999 in main queue.

b. At starting, Each player maintain own Max heap tree (MHT).

c. Each player fetches one no. at a time from main queue and insert it into own MHT in circular fashion (one by one) till main queue empty.

d. Now every player has MHT of 25 no..

Winning strategy:

The player with max. value at top from others implies to winner. Otherwise restart game.

Write a menu driven program for above data structure with separate and features:

28. Best Min Heap Tree Game

Four players are playing game with following features:

a. Total 100 random no. from range 100-999 in main stack.

b. At starting, each player maintains own Min Heap Tree.

c. Each player fetches one no. at a time from main stack and insert it into own Min Heap Tree in circular fashion (one by one) till main stack is empty.

d. Now every player has Min Heap Tree of 25 no.

Winning strategy:

The player with min value of root nodes key value of Min Heap Tree implies to winner. Otherwise restart game.

Write a menu driven program for above data structure with separate and features:

29. Employee workload Management

To maintain Project performance of employee, data structure Created:

Student Structure

a. E id.

b. Name

c. 3 projects and corresponding employee project Hr. of each project (Project id, total Working hr., Project Hr., %age).{Hint 2D array}

Write a menu driven program for above data structure with separate and features:

Menu:

a. Insert new employee

b. Update Project hr. of project id for given Employee id.

c. Update Project hr. of all project for all Employee

d. Display Employee Information

e. total work done by Employee in every month and year

Features:

a. Maintain each Employee information in single node of doubly singly linked list.

b. Start pointer always point to Largest % of Average project Hr. Always.

c. Insert Employee in already sorted list with proper location

d. After updating Project Hr. of Employee all node will be in Sort on basis of Average Project Hr. (Decreasing order)

e. Insert min 5 Employee information using menu option 1.

30. B+ tree card arrangements

Four players is playing Card game with following features:

a. Total 52 cards arranged randomly in main queue.

b. At starting, Each player maintain own B+ tree according to card no.(from 1-13).

c. Each player fetches one card at a time from main queue and insert it into own B+ tree in circular fashion one by one till main queue empty.

d. Display B+ tree for each Player

Card structure:

Color : (red/ Black)

Type: (clubs (♧), *diamonds* (◊), *hearts* (♥) and spades (♤))

Number (1to 13)

Write a menu driven program for above data structure with separate and features:

31. B Tree Card Arrangements Problem

Four players is playing Card game with following features:

a. Total 52 cards arranged randomly in main queue.

b. At starting, Each player maintain own B tree according to card no.(from 1-13).

c. Each player fetches one card at a time from main queue and insert it into own B tree in circular fashion one by one till main queue empty.

d. Display B tree for each Player

Card structure:

Color : (red/ Black)

Type: (clubs (♧), *diamonds* (◊), *hearts* (♥) and spades (♤))

Number (1to 13)

Write a menu driven program for above data structure with separate and features:

32. Diagonal Elements Problem

Create a singly linked list containing diagonal elements of the 2-D Array as the information part. Further, extend the program and write a menu driven program which provides facility to sort, display and delete elements from the linked list.

33. Playing with Polynomials

Write a menu driven program which performs operation on two polynomials based on user choice. The program can perform addition, subtraction and multiplication on two polynomials.

34. Web History

Write a program to store URLs of web pages visited by a person in a doubly linked list with most recent at the top and with the restriction that the linked list can store not more than 50 URLs. In case the number of webpages visited exceeds the linked list capacity, the older will be deleted automatically. Further, the program shall have the facility to display the history and search a pre-visited URL.

35. Numbers Addition Using Linked List

Digits of two numbers are stored in two different linked list. Write a program to store the sum of the numbers into a third linked list.

36. Modified Stack

Write a program to implement a special type of stack, which apart from supporting normal stack operations: push(), isFull(), pop() and isEmpty(), supports additional operations: getMax(), getMin(), getMiddle().

37. Level Order Traversing

Write a program to create a binary search tree. Further, extend the program and write a menu driven program which provides the following facilities: Inserting a Node, Deleting a Node, traversing a tree in level order beginning from root.

38. Postorder Traversal

Write a program to create a binary search tree. Further, extend the program and write a menu driven program which provides the following facilities: Insert a Node, Deleting a Node, traversing a tree in postorder (non-recursive).

39. Single Array, Two Stacks

Write a program to implement two stacks using single array. First stack shall be able to store values less than equal to N and will grow from left go right, whereas the second stack shall be able to store values greater than N and shall grow from right to left. Do not divide the array equally among the two stacks.

40. Leaf Nodes Paths

Write a program to create a binary search tree. Further, extend the program and write a menu driven program which provides the following facilities: Number of Leaf Nodes, Number of Non-Leaf Nodes, Print all Paths from Root to Leaf Nodes.

41. Mirror Image of BST

Write a program to convert a binary search tree to its mirror.

42. Playing with AVL

Write a program to create AVL tree. Further extend the program further, extend the program and write a menu driven program which provides the following facilities: Inserting a Node and Deleting a Node from the AVL Tree.

43. BST to Doubly Linked List

Given a binary tree, convert it into a doubly linked list following the spiral order. The conversion should be done in such a way that the left child pointer of a binary tree node should act as a previous pointer for doubly linked list node and the right child pointer should act as a next pointer for doubly linked list node

44. Implement parenthesis check that it is balance or imbalance.

[{ ( ) }] ← BALANCE

[{ (})] ← IMBALANCE

45. You are given a binary tree rooted at 1. You have to find the mirror image of any node q_i about node 1. If it doesn't exist then print -1.

Input:

First line of input is N and Q.

Next N-1 line consists of two integers and one character first of whose is parent node , second is child node and character "L" representing Left child and "R" representing right child.

Next Q lines represents q_i.

Output:

For each q_i print it mirror node if it exists else print -1.

NOTE: 1 is mirror image of itself.

Constraints:

1 <= N <= 10^3

1<= Q <= 10^3

SAMPLE INPUT

```
10 8
1 2 R
1 3 L
2 4 R
2 5 L
3 6 R
3 7 L
5 8 R
5 9 L
7 10 R
2
5
3
6
1
10
9
4
```

SAMPLE OUTPUT

```
3
6
2
5
1
```

```
-1
-1
7
```

46. You are given a tree of N nodes and N−1 edges. Now you need to select two nodes a and b in the tree such that the cycle that will be formed after adding an edge between the two nodes a and b, its length should be maximum. If there are more than one possible answer, you can output any of them.

Input

The first line contains an integer N as input. Next N−1 lines contain a pair of integers (a,b) that denote there is an edge between the two nodes a and b in the tree.

Output

In the output, you need to print two integers separated by space which denote the nodes between which you can add the edge so as to maximize the length of the cycle in the tree.

Constraints

1≤N≤105

SAMPLE INPUT

```
7
1 2
1 3
2 4
2 5
3 6
3 7
```

SAMPLE OUTPUT

```
4 6
```

47. DISK TOWER (QUEUES)

Your task is to construct a tower in N days by following these conditions:

Every day you are provided with one disk of distinct size.

The disk with larger sizes should be placed at the bottom of the tower.

The disk with smaller sizes should be placed at the top of the tower.

The order in which tower must be constructed is as follows:

You cannot put a new disk on the top of the tower until all the larger disks that are given to you get placed.

Print N lines denoting the disk sizes that can be put on the tower on the ith day.

Input format

First line: N denoting the total number of disks that are given to you in the N subsequent days

Second line: N integers in which the ith integers denote the size of the disks that are given to you on the ith day

Note: All the disk sizes are distinct integers in the range of 1 to N.

Output format

Print N lines. In the ith line, print the size of disks that can be placed on the top of the tower in descending order of the disk sizes.

If on the ith day no disks can be placed, then leave that line empty.

Constraints

1≤N≤106

1≤size of a disk≤N

SAMPLE INPUT

5

4 5 1 2 3

SAMPLE OUTPUT

5 4

3 2 1

48. The Monk is trying to explain to its users that even a single unit of time can be extremely important and to demonstrate this particular fact he gives them a challenging task.

    There are **N** processes to be completed by you, the chosen one, since you're Monk's favorite student. All the processes have a unique number assigned to them from **1 to N**.

    Now, you are given two things:

    The **calling** order in which all the processes are called.

    The **ideal** order in which all the processes should have been executed.

    Now, let us demonstrate this by an example. Let's say that there are **3 processes**, the calling order of the processes is: **3 - 2 - 1.** The ideal order is: **1 - 3 - 2,** i.e., process number 3 will only be executed after process number 1 has been completed; process number 2 will only be executed after process number 3 has been executed.

Iteration #1: Since the ideal order has process #1 to be executed firstly, the calling ordered is changed, i.e., the first element has to be pushed to the last place. Changing the position of the element takes 1 unit of time. The new calling order is: 2 - 1 - 3. Time taken in step #1: 1.

Iteration #2: Since the ideal order has process #1 to be executed firstly, the calling ordered has to be changed again, i.e., the first element has to be pushed to the last place. The new calling order is: 1 - 3 - 2. Time taken in step #2: 1.

Iteration #3: Since the first element of the calling order is same as the ideal order, that process will be executed. And it will be thus popped out. Time taken in step #3: 1.

Iteration #4: Since the new first element of the calling order is same as the ideal order, that process will be executed. Time taken in step #4: 1.

Iteration #5: Since the last element of the calling order is same as the ideal order, that process will be executed. Time taken in step #5: 1.

Total time taken: 5 units.

PS: Executing a process takes 1 unit of time. Changing the position takes 1 unit of time.

Input format

The first line a number **N**, denoting the number of processes. The second line contains the calling order of the processes. The third line contains the ideal order of the processes.

Output Format

Print the total time taken for the entire queue of processes to be executed.

Constraints:

1<=**N**<=100

SAMPLE INPUT

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

SAMPLE OUTPUT

```
5
```

49. **Stack and Queue <Nissan>(STACK)**

You are given a stack of **N** integers such that the first element represents the top of the stack and the last element represents the bottom of the stack. You need to pop at least one element from the stack. At any one moment, you can convert stack into a queue. The bottom of the stack represents the front of the queue. You cannot convert the queue back into a stack. Your task is to remove exactly **K** elements such that the sum of the **K** removed elements is maximised.

Input format :

The first line consists of two space-separated integers **N** and **K**.

The second line consists of **N** space-separated integers denoting the elements of the stack.

Output format :

Print the maximum possible sum of the **K** removed elements

Constraints :

1≤*N*≤105
1≤*K*≤*N*
1≤*Ai*≤109
SAMPLE INPUT
10 5
10 9 1 2 3 4 5 6 7 8
SAMPLE OUTPUT
40

50. **Perfect Pair(HASH TABLE)**

Rajiv and Nitish had a fight because Rajiv was annoying Nitish with his question. Rajiv being a genius in arrays gave Nitish an array of natural numbers ***A*** of length ***N*** with elements ***A_1, A_2, . . ., A_N***. Nitish has to find the total amount of perfect pairs in the array. A perfect pair ***(A_i, A_j)*** is a pair where ***($A_i + A_j$)*** is a perfect square or a perfect cube and **i ≠ j**. Since Rajiv and Nitish are not talking with each other after the fight you have been given the question to solve and inturn make both of them a perfect pair again. ***NOTE :-*** A pair ***(A_i, A_j)*** and ***(A_j, A_i)*** are same and not to be counted twice.

Input

The first line on the input contains the a single integer ***T*** denoting the number of test cases. The first line of each test case contains a single integer ***N***. The second line contains ***N*** space-separated integers ***A_1, A_2, . . ., A_N***.

Output

For each test case, print a single line containing a single integer denoting the total number of perfect pairs.

Constraints

1 ≤ ***T*** ≤ 10
1 ≤ ***N*** ≤ 10^5
1 ≤ ***A***$_i$ ≤ 10^3

SAMPLE INPUT

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

SAMPLE OUTPUT

```
3
2
```

51. **Kth Character(HASH TABLE)**

Given a string S of length N, you have to answer Q queries. Each query includes a range (L and R) and a number K. For every query to have to print the lexicographically Kth smallest alphabet in the given range.

INPUT

The first line includes two integers N and Q. N representing the length of the string and Q representing the number of queries.

Second line contains the string S.

Next Q lines contain 3 integers L, R and K

OUTPUT

The output should contain q lines, each line representing the Kth character in the given range of the query.

CONSTRAINT

1 <= N <= 1000000
1 <= Q <= 1000000
1 <= L <= R <= 1000000
1 <= K <= R - L + 1
The string consists of small english alphabets

SAMPLE INPUT

```
10 5
```

```
ababcdabcd
1 4 2
1 4 3
1 5 5
1 10 8
7 10 3
```

SAMPLE OUTPUT

```
a
b
c
c
c
```

52. **Monk and BST (BINARY SEARCH TREE)**

Monk as always has brought a new task for Fredo. He asks Fredo to create a Binary Search Tree (BST) with *L* levels and 2L−1 nodes. Let the sum of all node values in the BST be *M*. He further adds that *M* should be smallest possible integer greater than *S*, where *S* is an integer given by Monk. He states the rules of creating BST as follows:

The left sub-tree contains only nodes with values less than or equal to the parent node; the right sub-tree contains only nodes with values greater than the parent node.

If a node has level *i*, then the subtree rooted at that node should have exactly 2L−i number of distinct values in the subtree. Note that it is the number of distinct values in the subtree and not the number of nodes in the subtree.

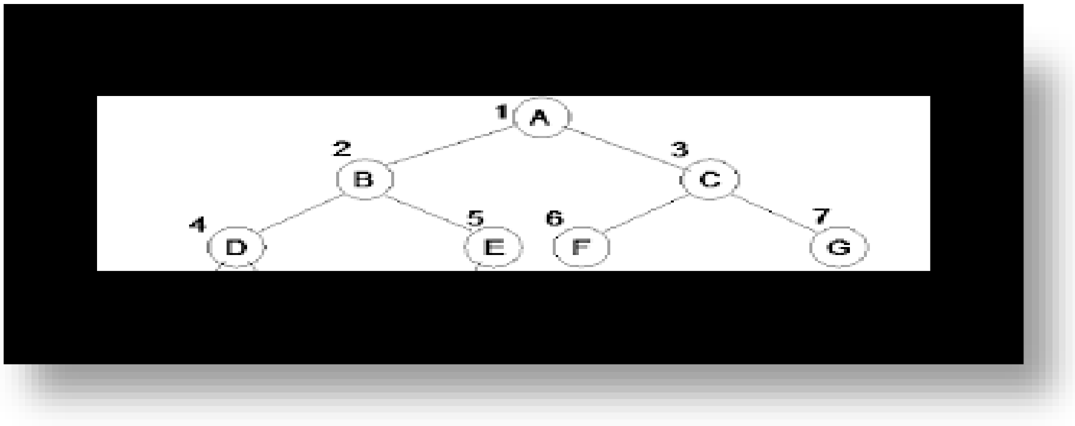
If *a* is the smallest value in the tree and *b* is the largest value, then all values from *a* to *b* must come atleast once in the tree.

Level of root is 1, next level is 2 and so on.

Now, he will ask two type of queries to Fredo.

Type *0*: Find the closest node to root whose value is equal to val and print path to that node from the root. If root has value equal to val, print "root". Else print "*l*" when we visit left child of any node and "*r*" when we visit right child of any node.

Type *1*: Tell the value of kth node in the tree. The nodes are numbered as:



Here 1,2 and so on are node numbers and A,B etc. are values of nodes.

Finally, he will ask Fredo *Q* queries, each query belonging to one of the types mentioned above. Since, Fredo is new to this concept, help him complete this task.

Input format

First line consists of two integers *L* and *S* as described in the question.
Second line consists of *Q*, denoting the number of queries.
Each of the following *Q* lines consists of two integers. The first integer denoting the type of query and second denoting either val or *k* as described in the queries.

Output format

For each query, print the required answer in a separate line.

Input Constraints:

1≤L≤30
1≤S≤1018
1≤Q≤105
0≤type≤1
1≤k≤2L−1
val is in between minimum and maximum value in tree(inclusive).
Values of *S*, val and *k* are such that answer would always exist for them.
Node values will be non-negative for all inputs.
M will never exceed 2×1018

SAMPLE INPUT

```
3 16
5
0 3
0 4
1 4
1 7
0 5
```

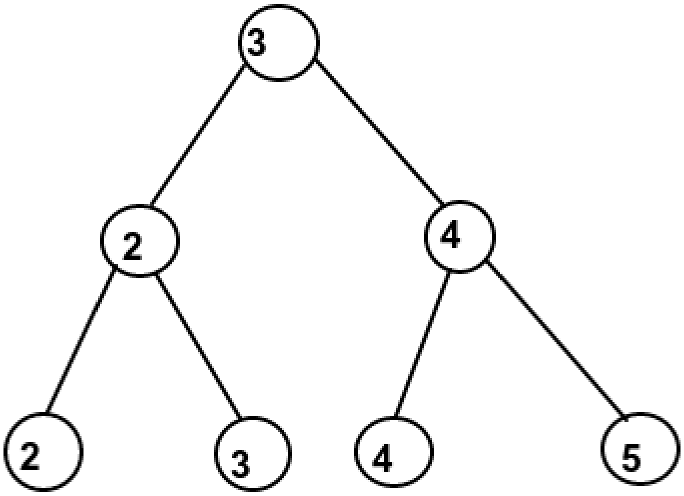
SAMPLE OUTPUT

```
root
r
```

2
5
rr

Explanation

The tree would look like:



Sum = 3+2+4+2+3+4+5=23 which is smallest possible sum which we can get which is greater than 16.
Query 1: Node with value 3 which is closest to root is root itself.
Query 2: Node with value 4 which is closest to root is right child of root. So, we print "r".
Query 3: Value of node at number 4 is 2.
Query 4: Value of node at number 7 is 5.
Query 5: Node with value 5 which is closest to root is grandchild of root. So, we print "rr" which is path to node from root.

53. Eerie Planet(QUEUE)

You own a club on eerie planet. The day on this planet comprises of H hours. You appointed C crew members to handle the huge crowd that you get, being the best club on the planet. Each member of the crew has fixed number of duty hours to work. There can be multiple or no crew members at work at any given hour of the day.
Being on weird planet, the rules of this club cannot be normal. Each member of the crew only allows people who are taller than him to enter the club when he is at work.
Given the schedule of work and heights of the crew members, you have to answer Q queries. Each query specifies the time of entry and height of a person who is visiting the club. You have to answer if the person will be allowed to enter the club or not.

Input:

First line of the input contains 3 integers, H,C,Q. Representing number of hours in a day, number of crew members and number of queries respectively.
Next C lines follow, where each line contains 3 integers, hi,Si,Ei, representing height of the crew member and start and end hour of his/her work schedule. He/she works for hours [Si,Ei], both inclusive.
Next Q lines follow, each containing 2 integers, hi,ti, representing height and time (in hour) of the person trying to enter the club.

Output:

Q lines, each line containing "YES" or "NO", without the quotes, answering if the person will be allowed to enter the club or not.

Constraints:

1≤H≤109
1≤C≤105
1≤Q≤105
1≤Si≤Ei≤H
1≤ti≤H
1≤hi≤107

SAMPLE INPUT

```
10 1 5
50 2 6
10 1
10 2
50 5
51 6
100 10
```

SAMPLE OUTPUT

```
YES
NO
NO
YES
YES
```

54. Little Monk and ABD(tree)

Little Monk meets his another favorite cricketer this time: A-B-D. Little Monk says that he is the biggest fan of ABD. ABD does not believe the Monk at all, and asks him to prove how much does he know about ABD's career.

So, ABD tells the Monk that given his latest **N** innings, he is going to ask him **Q** number of questions about his career which would involve questions of two types:

Find the **k**th smallest score of his career - denoted by a query of type: "**k S**", where **k** is an integer and **S** denotes *smallest*.

Find the **k**th largest score of his career - denoted by a query of type: "**k L**", where **k** is an integer and **L** denotes *largest*.

Help Little Monk answer as many queries as possible!

Input Format:-

The first line contains an integer **N**, which denotes the number of innings played by ABD which have to be dealt by The Monk. The next line contains **N** space separated integers denoting the number of scores made by ABD. The next line contains an integer **Q** denoting the number of questions ABD is going to be asking. After that, the next **Q** lines will contain a query like the ones mentioned above.

Output format

Print the required answer for each query on a newline.

Constraints:

1 ≤ **N** ≤ 106

1 ≤ **Q** ≤ 106

1 ≤ **K** ≤ 105

1 ≤ $\mathbf{N_i}$ ≤ 109

SAMPLE INPUT

```
5
1 2 3 4 5
3
3 L
3 S
1 L
```

SAMPLE OUTPUT

```
3
3
5
```

Explanation

3rd largest score is 3.
3rd smallest score is 3.
1st largest score is 5.

55. **Number Recovery(QUEUE)**

A **positive** integer X has been stolen. But luckily, N hints are available, each described by two integers ai and di, meaning that |X−ai|=di. The hints are numbered 1 through N. While some of those hints are helpful, some might be just a lie. Therefore, we are going to investigate the number X under different possible scenarios.

Initially, we neither trust nor distrust any hint. That is, each hint may be either true or false.

Then, in each of the Q stages, we will either:

```
1 id
```

a. Entrust the id-th hint (1≤id≤N). That is, from now on, the id-th hint must be true, unless declared otherwise in the future.

```
2 id
```

b. Distrust the id-th hint (1≤id≤N). That is, from now on, the id-th hint must be false, unless declared otherwise in the future.

```
3 id
```

c. Neutralize the id-th hint (1≤id≤N). That is, from now on, the id-th hint may be either true or false, unless declared otherwise in the future.

After each stage, you should determine the number of possible positive values X and report such values in an increasing order. If there are infinitely many such values, print −1 instead.

Input

The first line contains two space-separated integers N and Q.
The i-th of the following N lines contains two space-separated integers ai and di, describing the i-th hint. It is guaranteed that no two hints are identical. That is, for every two different i, j, it is guaranteed that ai≠aj or di≠dj.
Then, Q lines follow, each containing two integers t and id — the type of an update and the index of an affected hint.

Output

After each stage, print the number of possible values of X (in case there are infinitely many of them, print −1). If the number of possible values is finite and non-zero, in the same line, continue to print those values in an increasing order.

Constraints

1≤N,Q≤200000
0≤ai,di≤109
1≤t≤3 for every stage (update).
1≤id≤N for every stage.
In tests worth 74 points in total, ai,di≤500000.

Note that the expected output feature for custom input is disabled for this contest.

SAMPLE INPUT

```
3 10
3 0
0 3
6 3
1 1
3 1
1 2
3 2
1 3
3 3
1 1
1 2
2 1
1 3
```

SAMPLE OUTPUT

```
1 3
-1
1 3
-1
2 3 9
-1
1 3
1 3
0
0
```

Explanation

In the sample test, we are given N=3 hints and Q=10 stages.
The first stage is described by a pair "1 1", which represents entrusting hint 1.
After this stage, |X−3|=0 must be true, so X must be equal to 3. We report 1 possible value: 3.
Then, the information that |X−3|=0 is neutralized at stage 2. At this point, X could be any positive integer, so we print −1 in the second line.

56. **Bag Of Numbers(STACKS)**

Vikas is given a bag which consists of numbers (integers) blocks,Vikas has to organize the numbers again in the same order as he has inserted it into the bag, **i.e. the first number inserted into the bag by Vikas should be picked up first followed by other numbers in series.** Help Vikas to complete this work in O(n) time complexity with the condition to use one extra bag to complete the work **(assume that the bags are compact and is in the form**

of a stack structure and has the same width as that of the number blocks and is large enough to fill the bag to the top and the number taken from bag is in reverse order).

Hint: You may use the concept of Stacks.

SAMPLE INPUT

input: 15 21 39 390 392 380.

SAMPLE OUTPUT

output: 15 21 39 390 392 380.

57. **Bob and String(hashing table)**

Bob and Khatu both love the string. Bob has a string **S** and Khatu has a string **T**. They want to make both string **S** and **T** to anagrams of each other. Khatu can apply two operations to convert string **T** to anagram of string **S** which are given below:
1.) Delete one character from the string **T**.
2.) Add one character from the string **S**.

Khatu can apply above both operation as many times he want. Find the minimum number of operations required to convert string **T** so that both **T** and **S** will become anagram of each other.

Input:

First line of input contains number of test cases **T**. Each test case contains two lines. First line contains string **S** and second line contains string **T**.

Output:

For each test case print the minimum number of operations required to convert string **T** to anagram of string **S**.

Constraints:

1 ≤ **T** ≤ 10
1 ≤ **|S|,|T|** ≤ 10^5

SAMPLE INPUT

```
4
abc
cba
abd
acb
talentpad
talepdapd
code
road
```

SAMPLE OUTPUT

```
0
2
4
4
```

58. **Remove Friends (STACK)**

After getting her PhD, Christie has become a celebrity at her university, and her facebook profile is full of friend requests. Being the nice girl she is, Christie has accepted all the requests.

Now Kuldeep is jealous of all the attention she is getting from other guys, so he asks her to delete some of the guys from her friend list.

To avoid a 'scene', Christie decides to remove some friends from her friend list, since she knows the popularity of each of the friend she has, she uses the following algorithm to delete a friend.

Algorithm Delete(Friend):

```
DeleteFriend=false
for i = 1 to Friend.length-1
    if (Friend[i].popularity < Friend[i+1].popularity)
        delete i th friend
        DeleteFriend=true
        break
if(DeleteFriend == false)
    delete the last friend
```

Input:

First line contains **T** number of test cases. First line of each test case contains **N**, the number of friends Christie currently has and **K** ,the number of friends Christie decides to delete. Next lines contains **popularity** of her friends separated by space.

Output:

For each test case print **N-K** numbers which represent popularity of Christie friend's after deleting **K** friends.

Constraints

1<=**T**<=1000

1<=**N**<=100000

0<=**K**< N
0<=**popularity_of_friend**<=100
NOTE:
Order of friends after deleting exactly **K** friends should be maintained as given in input.
SAMPLE INPUT
3
3 1
3 100 1
5 2
19 12 3 4 17
5 3
23 45 11 77 18

59. Due to the demonetization move, there is a long queue of people in front of ATMs. Due to withdrawal limit per person per day, people come in groups to withdraw money. Groups come one by one and line up behind the already present queue. The groups have a strange way of arranging themselves. In a particular group, the group members arrange themselves in increasing order of their height(not necessarily strictly increasing).

Swapy observes a long queue standing in front of the ATM near his house. Being a curious kid, he wants to count the total number of groups present in the queue waiting to withdraw money. Since groups are standing behind each other, one cannot differentiate between different groups and the exact count cannot be given. Can you tell him the minimum number of groups that can be observed in the queue?

Input format:
The first line of input contains one positive integer *N*. The second line contains *N* space-separated integers H[i] denoting the height of i-th person. Each group has group members standing in increasing order of their height.
Output format:
Print the minimum number of groups that are at least present in the queue?
Constraints:
1≤N≤1,000,000
1≤H[i]≤1,000,000
SAMPLE INPUT
4
1 2 3 4
SAMPLE OUTPUT
1

60. You are given an image, that can be represented with a *2*-d *n* by *m* grid of pixels. Each pixel of the image is either on or off, denoted by the character '*0*' or '*1*', respectively.

You would like to compress this image. You want to choose an integer k>1 and split the image into *k* by *k* blocks. If *n* and *m* are not divisible by *k*, the image to be padded with 0's to the right and/or to the bottom. Each pixel in each individual block must have the same value.

The given image may not be compressible in its current state. Find the minimum number of pixels you need to toggle(padded pixels also can be toggled) in order for the image to be compressible for some *k*.

More specifically, the steps are to first choose *k*, then the image is padded with 0s, then, we can toggle the pixels so it is compressible for some *k*. The image must be compressible in that state.

Input Format

The first line of input will contain two integers n,m, the dimensions of the image.

The next *n* lines of input will contain a binary string with exactly *m* characters, representing the image.

Output Format

Print a single integer, the minimum number of pixels needed to toggle to make the image compressible.

Constraints

2≤n,m≤2500

SAMPLE INPUT

```
3 5
00100
10110
11001
```

SAMPLE OUTPUT

```
5
```

61. There are N chairs arranged in a row. K people come in a line and start occupying the chairs. Each person wants to be as far as possible from every other person. So, every person arriving looks for the largest empty **continuous** sequence of unoccupied chairs and occupies the middle position. They have a preference indicating whether they would choose the left or the right chair if there are two chairs at the middle to chose from (else the preference does not matter, since there is only 1 chair at the center). If there are multiple largest empty sequences, then the person chooses the sequence which appears first from left side.

You are asked to answer Q queries. Determine which person has occupied the queried position.

Input Format

The first line of every test file are 2 integers N and K.
The next line contains a string S of length K. The ith character would be 'L' or 'R' indicating the preference of the ith person - the left or the right seat respectively.
Next line contains an integer Q - the number of queries.
Next Q lines contain an integer Qi - the queried position.

Output Format

For each query, output the persons' entry number if it is occupied, else print '−1' without quotes in a new line.

Constraints

1≤N≤1018
1≤K≤min(N, 105)
1≤Q≤105
1≤Qi≤N
length(S)=K

Sample Inputs

Input	**Output**
3 3 RRL 3 1 3 2	2 3 1

SAMPLE INPUT

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

SAMPLE OUTPUT

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

62. **Stevie :** " The second half of a programming contest is just as important as the second half of a football match, it's the decisive part ". So, here you are presumably having got over the first half. Also, let's not forget this guy who has given us some real memories to cherish :

You are given *N* segments [L,R], where L≤R . Now, you need to answer some queries based on these segments.

Overall, you need to answer *Q* queries. In each query you shall be given *2* integers *K* and *X*. You need to find the size of the Kth smallest segment that contains point *X*.

If no *K* segments contain point *X* , print *1* instead as the answer to that query.

A segment [L,R] is said to contain a point *X*, if L≤X≤R . When we speak about the Kth smallest segment, we refer to the one having the Kth smallest size. We define the size of a segment to be the number of integral points it contains, i.e. R+1−L

Input Format :
The first line contains a single integer *N*,
Each of the next *N* lines contains *2* space separated integers, where the *2* integers on the ith line denote the start and end points of the ith segment given to you.
The next line contains a single integer *Q*.
Each of the next *Q* lines contains *2* space separated integers *K* and *X*.
Output Format :
Print the ansnwer to each query on a new line.
Constraints :
1≤N,Q≤2×105
1≤Li≤Ri≤109 ,where 1≤i≤N

1≤K≤N

1≤X≤109

SAMPLE INPUT

```
5
1 2
1 3
2 4
4 8
1 9
2
2 4
4 2
```

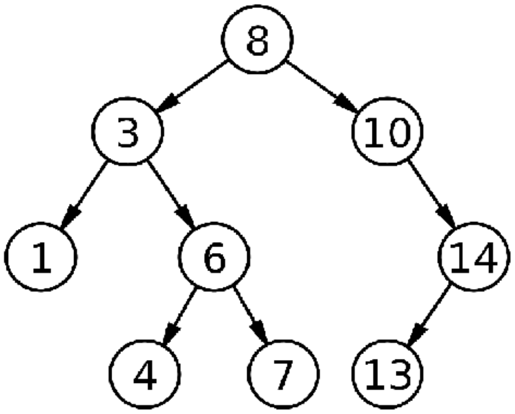
SAMPLE OUTPUT

```
5
9
```

63. Yatin is playing PUBG and he has reached a place with a large staircase in front of him. And there is an enemy at each landing of the staircase.

This is not the actual staircase. He is looking at the staircase from sideways.

The staircase is analogous to a binary tree with each of its nodes as a landing of the staircase and each of its edges as stairs from one landing to another.



Yatin wants to kill the maximum possible number of enemies. He can kill every person he can see from his position with his suppressed sniper gun. But he can see only the persons at the leftmost standing at each level and cannot see the rest.

Before starting shooting them, he wants to know how many persons he can kill. He is busy keeping an eye on the enemies. So he wants you to find out the maximum number of people he can kill from that location by providing you with the analogous a binary search tree.

[Note: Players do not change their position after one player has died, i.e. the leftmost node remains the same even after player on that node has died. Or we can say that the nodes are not removed after the player on that node has died.]

Input

The first line of input contains a number t denoting the number of test cases.

The first line of each test case contains n, the number of nodes in the tree.

The second line of each test case contains n space separated integers (unique) a_i denoting the value at each node of the BST.

Output

Print the answer for each test case on a new line.

Constraints

1 <= t <= 20

1 <= n <= 1000

1 <= a_i <= 1000000000

SAMPLE INPUT

```
1
9
8 3 10 1 6 14 4 7 13
```

SAMPLE OUTPUT

4

64. Given an undirected graph of **N** nodes and **N** weighted edges, it doesn't contain parallel edges nor self loops, it's guaranteed the graph will always remain connected.

you have to support **Q** queries, in each query you will be given one edge to delete and one edge to add in such a way to graph will remain connected, after each query you have to tell the cost of minimum spanning tree.

you have to provide an online solution (i.e. you can't see future queries until you answer the current one)

Input:

First line contains two integers **N** and **Q**.

Each of the next **N** lines describe an edge by 3 integers **u, v, c**, it means there's an edge connecting nodes **u** and **v** and has weight equal to **c**.

Each of the next **Q** lines describe a query by 5 integers **x1, x2, x3, x4, x5**, let **ans** be the answer to previous query (if this is first query then **ans** = 0). now let **a = x1 + (ans mod 100)**, **b = x2 + (ans mod 100)**, **u = x3 + (ans mod 100)**, **v = x4 + (ans mod 100)**, **c = x5 + (ans mod 100)**, it means to delete edge connecting nodes **a** and **b** and add edge connecting **u** and **v** and has weight equal to **c**.

Output:

Output the answer to each query in new line, the answer is one integer equal to sum of weights of minimum spanning tree of the graph.

Constraints:

3 <= **N** <= 300,000

1 <= **Q** <= 300,000

Graph will always remain connected without self loops nor parallel edges

1 <= weight of edges <= 1,000,000

SAMPLE INPUT

```
6 2
1 2 2
1 3 3
1 4 4
1 5 5
1 6 6
5 6 7
5 6 2 3 7
-18 -17 -17 -16 -13
```

SAMPLE OUTPUT

```
20
20
```

65. **Balanced Brackets**

A bracket is considered to be any one of the following characters: (,) , { , } , [, or] .

Two brackets are considered to be a matched pair if the an opening bracket (i.e., (, [, or {) occurs to the

left of a closing bracket (i.e.,) ,] , or }) of the exact same type . There are three types of matched pairs of brackets: [] , {} , and () .

A matching pair of brackets is not balanced if the set of brackets it encloses are not matched. For

example, {[(])} is not balanced because the contents in between { and } are not balanced. The pair of square brackets encloses a single, unbalanced opening bracket, (, and the pair of parentheses encloses a single, unbalanced closing square bracket,] .

By this logic, we say a sequence of brackets is balanced if the following conditions are met:

It contains no unmatched brackets.

The subset of brackets enclosed within the confines of a matched pair of brackets is also a matched pair of brackets.

Given strings of brackets, determine whether each sequence of brackets is balanced. If a string is balanced, return YES . Otherwise, return NO .

Function Description

Complete the function isBalanced in the editor below. It must return a string: YES if the sequence is balanced or NO if it is not.

isBalanced has the following parameter(s):

s: a string of brackets

Input Format
The first line contains a single integer , the number of strings.
Each of the next lines contains a single string , a sequence of brackets.
Constraints
, where is the length of the sequence.
All chracters in the sequences ∈ { **{**, **}**, **(**, **)**, **[**, **]** }.
Output Format
For each string, return YES or NO .
Sample Input
3{
[()]}
{[(])}

{{[[(())]]}}

Sample Output

YES
NO
YES

Explanation

The string {[()]} meets both criteria for being a balanced string, so we print YES on a new line.

The string {[(])} is not balanced because the brackets enclosed by the matched pair { and } are
not balanced: [(]) .

The string {{[[(())]]}} meets both criteria for being a balanced string, so we print YES on a new line.

66. Once again, we have a lost guy who loves to play racing games. In the game, there are a total of *N* cities (numbered from 1 to *N*), and these are connected by *M* bi-directional roads of some certain lengths. The odd thing is, laws of physics don't hold here. For each road, the racer can only drive at a certain constant speed.

Now, he wants to know the minimum time it'll take for him to reach from city *u* to city *v* , or it is not possible to reach at all.

Input:
First line contains a single integer *T* denoting the number of test cases.
First line of each test case contains four integers *N*, *M*, *u* and *v* denoting the number of cities, number of roads, start city and destination city.
Next *M* lines contain four integers *x*, *y*, *len* and *sp* denoting that there is a road between city *x* and city *y* with length *len* and speed allowed of exactly *sp*.
Output:
For each test case, output in a new line the minimum time needed to reach from *u*
to *v*, or −1 if it is not possible to reach from *u* to *v* .
Your answer will be considered right, if it is within 10^{-2} of the correct answer.
Constraints
$1 \le T \le 3$
$1 \le N,M \le 10^5$
$1 \le u,v,x,y \le N$
$1 \le len \le 10^9$
$1 \le sp \le 100$

Sub-tasks
Sub-task 1 (10 points): ***sp*=1**
for all roads.
Sub-task 2 (15 points): ***len*=1**
for all roads.

Sub-task 3 (15 points): *N*,*M*≤1000
Sub-task 4 (60 points): Original Constraints.
Sample Input:

```
1
5 6 1 5
1 2 7 4
1 4 10 2
2 3 5 2
4 3 4 3
3 5 2 1
4 5 8 5
```

Sample Output:

```
6.25000000
```

EXPLANATION:
The optimal path is 1→2→3→5.

67. TDKPRIME - Finding the Kth Prime

The problem statement is really simple. There are some queries. You are to give the answers.

Input

An integer stating the number of queries Q(equal to 50000), and Q lines follow, each containing one integer K between 1 and 5000000 inclusive.

Output
Q lines with the answer of each query: the Kth prime number.
Example
Input:

```
7
1
10
100
1000
10000
100000
1000000
```

Output:

```
2
29
541
7919
104729
1299709
```

15485863

68. Game of Two Stacks

Alexa has two stacks of non-negative integers, stack and stack where index denotes the top of the stack. Alexa challenges Nick to play the following game:

In each move, Nick can remove one integer from the top of either stack or stack.

Nick keeps a running sum of the integers he removes from the two stacks.

Nick is disqualified from the game if, at any point, his running sum becomes greater than some

integer given at the beginning of the game.

Nick's final score is the total number of integers he has removed from the two stacks.

Given, and for games, find the maximum possible score Nick can achieve (i.e., the maximum number of integers he can remove without being disqualified) during each game and print it on a new line.

Input Format

The first line contains an integer, (the number of games). The subsequent lines describe each
game in the following format:
The first line contains three space-separated integers describing the respective values of (the number of integers in stack), (the number of integers in stack), and (the number that the sum of the integers removed from the two stacks cannot exceed).
The second line contains space-separated integers describing the respective values of
The third line contains space-separated integers describing the respective values of

Constraints

Subtasks

for of the maximum score.

Output Format

For each of the games, print an integer on a new line denoting the maximum possible score Nick can
achieve without being disqualified.

Sample Input 1

```
15
4 10
4 2 4 6 1
2 1 8 5
```

Sample Output 1

```
4
```

Explanation 1

The two stacks initially look like this:

The image below depicts the integers Nick should choose to remove from the stacks. We print as our answer, because that is the maximum number of integers that can be removed from the two stacks without the sum exceeding.

(There can be multiple ways to remove the integers from the stack, the image shows just one of them.)

69. In a far away Galaxy of Tilky Way, there was a planet Tarth where the sport of Competitive Coding was very popular. According to legends, there lived a setter who loved giving questions on DP+Graphs.

You are given *N* nodes, which are numbered from 1 to *N*. There are no edges between them initially. You have to handle *Q* queries of form-

1 *u v* - Add a bidirectional edge between Node *u* and Node *v*

2 *u v* - Tell if *u* and *v* are in same connected component or not.

Note: - Two nodes are said to belong to the same connected component, if you can reach from one node to the other, using some of the edges present at that time.

Input:

The first line has 2 integers, *N* and *Q* , denoting the number of nodes, and the number of queries respectively.

Next *Q* lines have queries as described above.

Output:

For every query of type 2, print Yes or No depending on whether *u* and *v* belong to the same connected component, or not.

Constraints

$1 \le N \le 10^5$

$1 \le Q \le 10^6$

$1 \le u,v \le N$

$u \ne v$

Subtasks

30% points – $N \le 1000$ and $Q \le 5000$

70% points - Original Constraints

Sample Input:

```
3 6
2 1 2
1 1 2
2 1 2
2 1 3
1 2 3
2 1 3
```

Sample Output:

```
No
```

Yes
No
Yes
EXPLANATION:
Initially all nodes are disconnected, hence answer for the first query of type 2 is No
An edge is added between 1 and 2..
Now 1 and 2 are in the same connected component, and hence answer is Yes
1 and 3 are NOT in the same connected component. Hence, answer is No. Recall all nodes are disconnected initially.
An edge is added between 2 and 3.
Now since 1 is connected to 2 which in turn is connected to 3, 1 and 3 are in the same connected component, and hence the answer is Yes.

70. In a far away Galaxy of Tilky Way, there was a planet Tarth where the sport of Tompetitive Toding was very popular. According to legends, there lived a setter who loved super linear sieve from the research papers he had read.

You are given two numbers *A* and *B*. Now, for every number *N* between *A* and *B*, both inclusive, we define "score" of the number *N* as follows -

Let the number of divisors of *N* be *K*. You are allowed to divide *N* by **one** of its **prime factors, ONLY ONCE**. Let the new number be *N′* and its divisors be *K′*. *N*'s score is (*K*−*K′*).

You have to find the maximum score of each number between *A* and *B* (each of them independently), and then output the sum of all those scores.

Input:
The first line has an integer *T*, the number of test cases per file.
The first line of each test case has 2 integers, *A* and *B* as described in the problem statement.
Output:
For every test case, print the **sum** of maximum scores for all numbers *N*
in range [*A*,*B*]
Constraints
$1≤T≤3$
$2≤A≤B≤10^{12}$
$B−A≤10^6$
Subtasks
10% points- B≤10^3
20% points- B≤10^6
70% points - Original Constraints
Sample Input:
2
12 12

8 10

Sample Output:

3

4

EXPLANATION:

In the first subtask, the only *N*

that we have to consider is 12. Its divisors are {1,2,3,4,6,12}. So *K*=6

.

Now, if we divide *N* by 3 (which is one of its prime factors), we get *N*′=4. Its divisors are {1,2,4}. So, *K*′=3. So, *K*−*K*′=6−3=3. You can check that this is the best that we can do. Hence the answer is 3.

71. Nestor was doing the work of his math class about three days but he is tired of make operations a lot and he should deliver his task tomorrow. His math's teacher gives him two numbers a and b. The problem consist of finding the last digit of the potency of base a and index b. Help Nestor with his problem. You are given two integer numbers: the base a (0 <= a <= 20) and the index b (0 <= b <= 2,147,483,000), a and b both are not 0. You have to find the last digit of a^b.

    Input

    The first line of input contains an integer t, the number of test cases (t <= 30). t test cases follow. For each test case will appear a and b separated by space.

    Output

    For each test case output an integer per line representing the result.

    Example

    Input:

    2

    3 10

    6 2

    Output:

    9

    6

72. You are given a sequence A[1], A[2], ..., A[N] . (|A[i]| ≤ 15007 , 1 ≤ N ≤ 50000). A query is defined as follows:

Query(x,y) = Max { a[i]+a[i+1]+...+a[j] ; x ≤ i ≤ j ≤ y }.

Given M queries, your program must output the results of these queries.

Input

The first line of the input file contains the integer N.
In the second line, N numbers follow.
The third line contains the integer M.
M lines follow, where line i contains 2 numbers xi and yi.
Output
Your program should output the results of the M queries, one query per line.
Example
Input:
3
-1 2 3
1
1 2

Output:
2

73. Farmer John has built a new long barn, with N (2 <= N <= 100,000) stalls. The stalls are located along a straight line at positions x1,...,xN (0 <= xi <= 1,000,000,000). His C (2 <= C <= N) cows don't like this barn layout and become aggressive towards each other once put into a stall. To prevent the cows from hurting each other, FJ wants to assign the cows to the stalls, such that the minimum distance between any two of them is as large as possible. What is the largest minimum distance?

Input
t – the number of test cases, then *t* test cases follows.
* Line 1: Two space-separated integers: N and C
* Lines 2..N+1: Line i+1 contains an integer stall location, xi
Output
For each test case output one integer: the largest minimum distance.
Example
Input:
1
5 3
1
2
8
4
9

Output:
3
Output details:
FJ can put his 3 cows in the stalls at positions 1, 4 and 8,
resulting in a minimum distance of 3.

74. One thing is clear from other comments here, the greedy solution will NOT work.

None of the explanations in the comment section are good at explaining how/why the given c++ solution by vkreddy21 works. I have tried explaining it below:

So far only 1 thing is clear in the solution: When elements from 2nd stack are added, and (more recently added/last) elements from 1st stack are popped off, the solution ensures that the max count does not decrease and sum <= x (given in input).

Graphically:

A much better way to understand this is to think about what the final solution to the problem will look like.

For example, a possible solution could look like:

[s1 s2 s1 s2 s2]

where s1, s2 refer to some element from stack 1 and 2 respectively. It may be noted that all s1 elements are contiguous (since problem requires us to only pick from top of stack). Same for s2 elements.

Now, since these elements are summed (which should be <= x), this means *they can be arranged in any order. So you could group all s1 accesses and then s2 accesses * (could have also done vice-versa). this is obviously correct because for "+" operator (using which sum is being calculated) does not care about order of operands.

Hence, our previous solution can be re-arranged to look like any of the following:

[s2 s2 s2 s1 s1]
(OR)
[s1 s1 s2 s2 s2]
What the second case ^ suggests is that you can create a solution like:
[s1 s1 s1 s1 s1]
and then add (or replace s1 elements if sum is exceeding "x") s2 elements one by one, eventually leading to
[s1 s1 s2 s2 s2] = [s1 s2 s1 s2 s2]
Note that ^ this final solution is same as the ^ solution (simply created as an example for this explanation) in the earlier part of this explanation.

75. Peter wants to generate some prime numbers for his cryptosystem. Help him! Your task is to generate all prime numbers between two given numbers!

Input

The input begins with the number t of test cases in a single line (t<=10). In each of the next t lines there are two numbers m and n (1 <= m <= n <= 1000000000, n-m<=100000) separated by a space.

Output

For every test case print all prime numbers p such that m <= p <= n, one number per line, test cases separated by an empty line.

Example

Input:

```
2
1 10
3 5
```

Output:

```
2
3
5
7
3
5
```

76. Its time of Raksha Bandhan so its compulsory for every brother to give a gift to his sister. Brother Jon is planning to give a string **S** of length **L** to his sister Arya.

After receiving the gift from her brother, Arya decided to share the string with her brother Jon.

String **S** contains characters from index **0** to index **L - 1**.

She will select a pivot called P ie any index **P(1 ≤ P ≤ L - 1)** and divides the original string into two parts. First part will contain characters from index **0** to index **P - 1** and second part will contain characters from index **P** to **L - 1**.

Jon will keep the first part and Arya will keep the second part.

Arya is great fan of LCP (Longest Common Prefix) so she decided that she will select such an index as pivot such that LCP of the two part is high.

Though she like LCP but she does not know how to calculate it. Help her to solve the problem.

Arya will ask you **Q** questions. Each question is represented by a index **P** and you have to answer the LCP (Longest Common Prefix) of the two parts.

Input

First line of input contains a string **S**.

Second line of input contains a positive number **Q** representing the number of question Arya will ask you.

Q lines will follow each containing a positive number **P**

Output
Output **Q** numbers(the required answer). Each number on a separate line.
Constraints
1 ≤ Length of the string(or L) ≤ 5 * 10^6
1 ≤ P ≤ L - 1
1 ≤ Q(number of queries) ≤ 10^6
All characters in the string will be small case english alphabets (a-z)
Example
Input:

```
abababa
4
1
2
3
6
```

Output:

```
0
2
0
1
```

Explanation
For the first query, first part is **a**.and second part is **bababa**, hence LCP is 0.
For the second query, first part is **ab**.and second part is **ababa**, hence LCP is 2.
For the third query, first part is **aba**.and second part is **baba**, hence LCP is 0.
For the fourth query, first part is **ababab**.and second part is **a**, hence LCP is 1.

77. In a far away Galaxy of Tilky Way, there was a planet Tarth where the sport of Tompetitive Toding was very popular. According to legends, there lived a setter known for his deep mathematical DP problems.

You are given a bit string *S* (string containing only 0's and 1's) of length *N* with some numbers missing. Each missing number/bit is replaced by a "_". Find number of ways of filling the blanks with 0 or 1 such that the resulting number (which is in binary form) is divisible by *M*

Input:

The first line has a single integer *T*, denoting number of test cases per file.
The first line of each testcase contains 2 integers, *N* and *M.*
The next line of input contains the string *S.*
Output:
For each test case, in a new line, print a single integer, denoting number of ways of filling the blanks such that resulting number is divisible by *M*

Constraints
$1 \le T \le 10$
$1 \le N, M \le 100$
S consists only of 0 , 1 and _
Number of _ in a single string is ≤15.
Subtasks
40% points - $N \le 20$
60% points - Original Constraints
Sample Input:
2
3 5
10_
3 2
1_0
Sample Output:
1
2
EXPLANATION:
For first case, we can fill the blank with 1
to get (101)2≡5 which is divisible by 5
(recall that the number is in binary!)
For second test case, we can fill the blank with 0 or 1 and get 4 and 6 respectively, both of which are divisible by 2.

78. You have an empty sequence, and you will be given queries. Each query is one of these three types:

1 x -Push the element x into the stack.

2 -Delete the element present at the top of the stack.

3 -Print the maximum element in the stack.

Input Format

The first line of input contains an integer, . The next lines each contain an above mentioned query. (It
is guaranteed that each query is valid.)
Constraints
Output Format
For each type query, print the maximum element in the stack on a new line.
Sample Input
10
1 97
21

```
20
21
26
1 20
231
91
3
```

Sample Output

```
26
91
```

79. DevG likes too much fun to do with numbers. Once his friend Arya came and gave him a challenge, he gave DevG an array of digits which is forming a number currently (will be called as given number). DevG was challanged to find the just next greater number which can be formed using digits of given number. Now DevG needs your help to find that just next greater number and win the challenge.

Input

The first line have t number of test cases (1 <= t <= 100). In next 2*t lines for each test case first there is number n (1 <= n <= 1000000) which denotes the number of digits in given number and next line contains n digits of given number separated by space.

Output

Print the just next greater number if possible else print -1 in one line for each test case.

Note : There will be no test case which contains zero in starting digits of any given number.

Example

Input:

```
2
5
1 5 4 8 3
10
1 4 7 4 5 8 4 1 2 6
```

Output:

```
15834
  1474584162
```

80. For sure, the love mobiles will roll again on this summer's street parade. Each year, the organisers decide on a fixed order for the decorated trucks. Experience taught them to keep free a side street to be able to bring the trucks into order.

The side street is so narrow that no two cars can pass each other. Thus, the love mobile that enters the side street last must necessarily leave the side street first. Because the trucks and the ravers move up closely, a truck cannot drive back and re-enter the side street or the approach street.

You are given the order in which the love mobiles arrive. Write a program that decides if the love mobiles can be brought into the order that the organisers want them to be.

Input
There are several test cases. The first line of each test case contains a single number n, the number of love mobiles. The second line contains the numbers 1 to n in an arbitrary order. All the numbers are separated by single spaces. These numbers indicate the order in which the trucks arrive in the approach street. No more than 1000 love mobiles participate in the street parade. Input ends with number 0.

Output
For each test case your program has to output a line containing a single word "yes" if the love mobiles can be re-ordered with the help of the side street, and a single word "no" in the opposite case.

Example

Sample input:

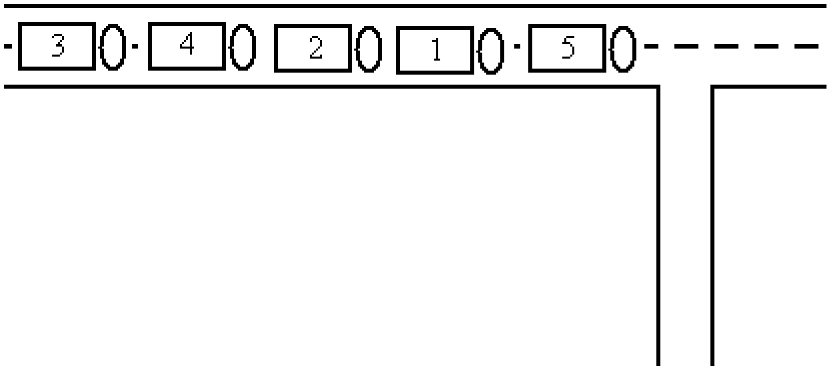
```
5
5 1 2 4 3
0
```

Sample output:

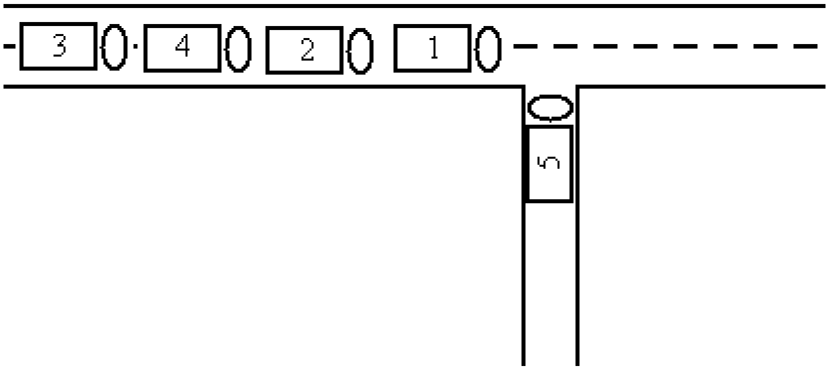
```
yes
```

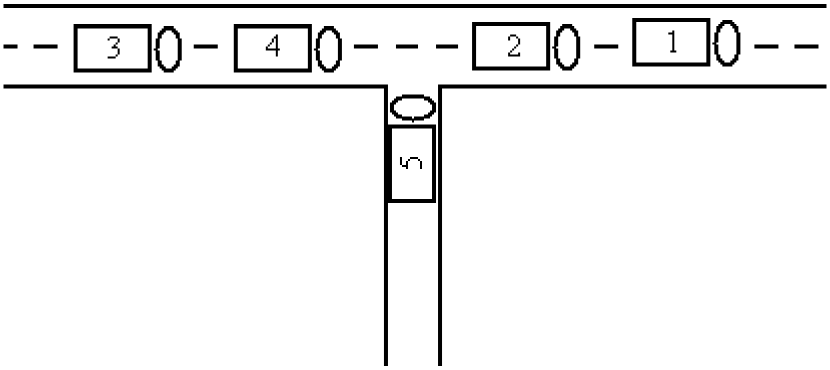
Illustration

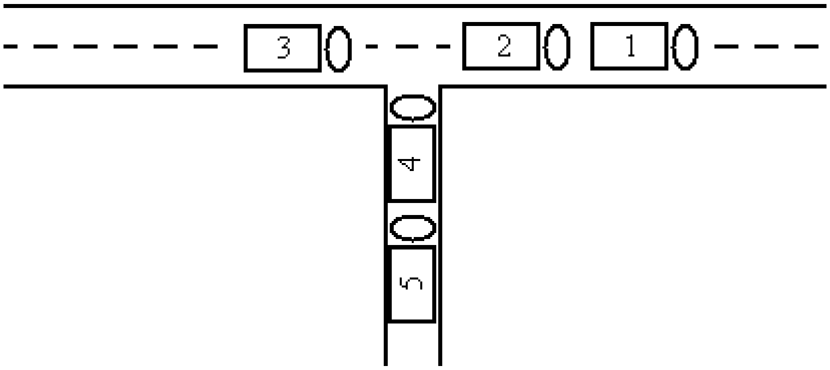
The sample input reflects the following situation:

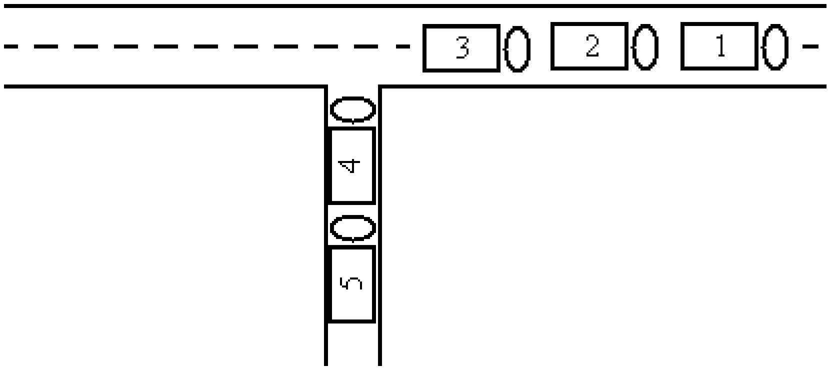


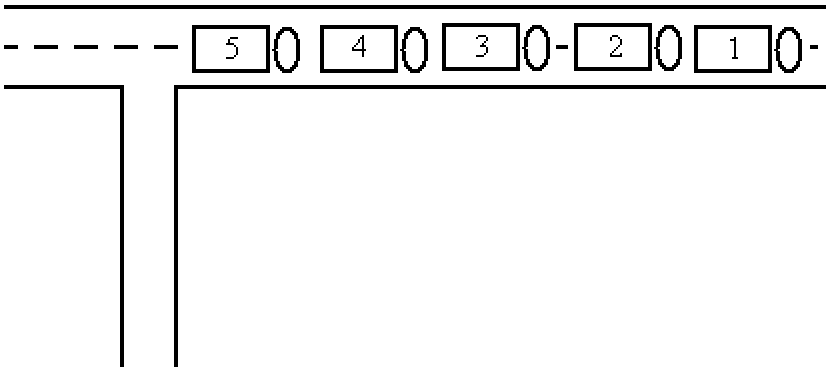
The five trucks can be re-ordered in the following way:











81. Transform the algebraic expression with brackets into RPN form (Reverse Polish Notation).

Two-argument operators: +, -, *, /, ^ (priority from the lowest to the highest), brackets ( ).

Operands: only letters: a,b,...,z. Assume that there is only one RPN form (no expressions like a*b*c).

Input
t [the number of expressions <= 100]
expression [length <= 400]
[other expressions]
Text grouped in [ ] does not appear in the input file.
Output
The expressions in RPN form, one per line.
Example
Input:

```
3
(a+(b*c))
((a+b)*(z+x))
((a+t)*((b+(a+c))^(c+d)))
```

Output:

```
abc*+
ab+zx+*
  at+bac++cd+^*
```

82. Forgotten languages (also known as extinct languages) are languages that are no longer in use. Such languages were, probably, widely used before and no one could have ever imagined that they will become extinct at some point. Unfortunately, that is what happened to them. On the happy side of things, a language may be dead, but some of its words may continue to be used in other languages.

Using something called as the Internet, you have acquired a dictionary of N words of a forgotten language. Meanwhile, you also know K phrases used in modern languages. For each of the words of the forgotten language, your task is to determine whether the word is still in use in any of these K modern phrases or not.

Input

The first line of the input contains an integer T denoting the number of test cases. The description of T test cases follows.

The first line of a test case description contains two space separated positive integers N and K.

The second line of the description contains N strings denoting a dictionary of the forgotten language.

Each of the next K lines of the description starts with one positive integer L denoting the number of words in the corresponding phrase in modern languages. The integer is followed by L strings (not necessarily distinct) denoting the phrase.

Output

For each test case, output a single line containing N tokens (space-separated): if the ith word of the dictionary exists in at least one phrase in modern languages, then you should output YES as the ith token, otherwise NO.

Constraints

1 ≤ T ≤ 20

1 ≤ N ≤ 100

1 ≤ K, L ≤ 50

1 ≤ length of any string in the input ≤ 5

Example

Input:

```
2
3 2
piygu ezyfo rzotm
1 piygu
6 tefwz tefwz piygu ezyfo tefwz piygu
```

```
4 1
kssdy tjzhy ljzym kegqz
4 kegqz kegqz kegqz vxvyj
```

Output:

```
YES YES NO
  NO NO NO YES
```

83. There are 100 houses located on a straight line. The first house is numbered 1 and the last one is numbered 100. Some M houses out of these 100 are occupied by cops.

    Thief Devu has just stolen PeePee's bag and is looking for a house to hide in.

    PeePee uses fast 4G Internet and sends the message to all the cops that a thief named Devu has just stolen her bag and ran into some house.

    Devu knows that the cops run at a maximum speed of x houses per minute in a straight line and they will search for a maximum of y minutes. Devu wants to know how many houses are safe for him to escape from the cops. Help him in getting this information.

Input

First line contains T, the number of test cases to follow.
First line of each test case contains 3 space separated integers: M, x and y.
For each test case, the second line contains M space separated integers which represent the house numbers where the cops are residing.

Output

For each test case, output a single line containing the number of houses which are safe to hide from cops.

Constraints

1 ≤ T ≤ 104
1 ≤ x, y, M ≤ 10

Example

Input:

```
3
4 7 8
12 52 56 8
2 10 2
21 75
2 5 8
10 51
```

Output:

```
0
18
9
```

Explanation

Example 1 : Cops in house 12 can cover houses 1 to 68, and cops in house 52 can cover the rest of the houses. So, there is no safe house.
Example 2 : Cops in house 21 can cover houses 1 to 41, and cops in house 75 can cover houses 55 to 95, leaving houses numbered 42 to 54, and 96 to 100 safe. So, in total 18 houses are safe.

84. Lapindrome is defined as a string which when split in the middle, gives two halves having the same characters and same frequency of each character. If there are odd number of characters in the string, we ignore the middle character and check for lapindrome. For example gaga is a lapindrome, since the two halves ga and ga have the same characters with same frequency. Also, abccab, rotor and xyzxy are a few examples of lapindromes. Note that abbaab is NOT a lapindrome. The two halves contain the same characters but their frequencies do not match.

Your task is simple. Given a string, you need to tell if it is a lapindrome.

Input:

First line of input contains a single integer T, the number of test cases.
Each test is a single line containing a string S composed of only lowercase English alphabet.

Output:

For each test case, output on a separate line: "YES" if the string is a lapindrome and "NO" if it is not.

Constraints:
1 ≤ T ≤ 100
2 ≤ |S| ≤ 1000, where |S| denotes the length of S

Example:

Input:

```
6
gaga
abcde
rotor
xyzxy
abbaab
ababc
```

Output:

```
YES
NO
YES
YES
NO
 NO
```

85. Given a string **S** consisting of only **1s** and **0s**, find the number of substrings which start and end both in **1**.

    In this problem, a substring is defined as a sequence of continuous characters $S_i, S_{i+1}, ...,$ S_j where $1 \leq i \leq j \leq N$.

    Input
    First line contains **T**, the number of testcases. Each testcase consists of **N**(the length of string) in one line and string in second line.
    Output
    For each testcase, print the required answer in one line.
    Constraints
    $1 \leq T \leq 10^5$
    $1 \leq N \leq 10^5$
    Sum of **N** over all testcases ≤ 10^5
    Example
    Input:
    2
    4
    1111
    5
    10001
    Output:
    10
    3
    Explanation
    #test1: All substrings satisfy.
    #test2: Three substrings S[1,1], S[5,5] and S[1,5] satisfy.

86. You are given **N** integers. In each step you can choose some **K** of the remaining numbers and delete them, if the following condition holds: Let the **K** numbers you've chosen be $a_1, a_2, a_3, ..., a_K$ in sorted order. Then, for each i ≤ **K** - 1, a_{i+1} must be greater than or equal to $a_i * C$.

    You are asked to calculate the maximum number of steps you can possibly make.
    Input
    The first line of the input contains an integer **T**, denoting the number of test cases. The description of each testcase follows.
    The first line of each testcase contains three integers: **N**, **K**, and **C**
    The second line of each testcase contains the **N** initial numbers
    Output
    For each test case output the answer in a new line.

Subtasks

Subtask #1 (40 points):

1 ≤ **N** ≤ 10^3

1 ≤ Sum of **N** over all test cases ≤ 10^3

Subtask #2 (60 points):

Original constraints

Constraints

1 ≤ **T** ≤ 100

1 ≤ **N** ≤ 3 * 10^5

1 ≤ **K** ≤ 64

2 ≤ **C** ≤ 50

1 ≤ $\mathbf{a_i}$ ≤ 10^{18}

1 ≤ Sum of **N** over all test cases ≤ 3 * 10^5

Example

Input:

```
2
6 3 2
4 1 2 2 3 1
6 3 2
1 2 2 1 4 4
```

Output:

```
1
2
```

Explanation

Testcase 1: You can make one step by choosing **{1, 2, 4}**.

Testcase 2: You can make one step by choosing **{1, 2, 4}** and another by choosing **{1, 2, 4}**.

87. You are given a string **S** of length **N** consisting only of **0**s and **1**s. You are also given an integer **K**.

You have to answer **Q** queries. In the **i**th query, two integers $\mathbf{L_i}$ and $\mathbf{R_i}$ are given. Then you should print the number of substrings of **S[L, R]** which contain at most **K 0**s and at most **K 1**s where **S[L, R]** denotes the substring from **L**th to **R**th characters of the string **S**.

In other words, you have to count number of pairs **(i, j)** of integers such that **L ≤ i ≤ j ≤ R** such that no character in substring **S[i, j]** occurs more than **K** times.

Input

The first line of input contains an integer **T**, denoting the number of test cases. Then **T** test cases follow.

The first line of each test case contains three space-separated integers **N**, **K** and **Q** as described in the problem. The second line contains a string **S** of length **N**. Then the next **Q** lines describe the query, where the **i**th line of them contains two space-separated integers **L_i** and **R_i**.

Output

For each query, print the required answer in a single line.

Constraints and Subtasks

$1 \le T \le 10^5$

$1 \le K \le N \le 10^5$

$1 \le Q \le 10^5$

$1 \le L_i \le R_i \le N$

Sum of **N** over all test cases in one test file does not exceed **10^5**

Sum of **Q** over all test cases in one test file does not exceed **10^5**

S consists only of **0**s and **1**s.

Subtask 1: 10 points

Sum of **N** over all test cases in one test file does not exceed **100**

Subtask 2: 10 points

Q = 1

$1 \le K \le \min(5, N)$

Subtask 3: 20 points

$1 \le Q \le 10$

Subtask 4: 60 points

Original constraints.

Example

Input:

```
1
8 2 3
01110000
1 4
2 4
5 8
```

Output:

```
8
5
7
```

Explanation

Query 1: Consider substring **P = S[1, 4] = "0111"**.

Out of **10** total substrings of **P**, substrings **P[1, 4]** and **P[2, 4]** are not valid because both contain more than two **1**s.

Other substrings contains at most two **0**s and at most two **1**s, thus the answer is **8**.

Query 2: Consider substring **P = S[2, 4] = "111"**.

Out of **6** total substrings of **P**, substrings **P[1, 3]** is not valid because it contains more than two 1s.

Query 3: Consider substring **P = S[5, 8] = "0000"**.
Out of **10** total substrings of **P**, substrings **P[1, 3]**, **P[1, 4]** and **P[2, 4]** are not valid because all contain more than two **0**s.

88. What is the maximum number of squares of size **2x2** that can be fit in a right angled

isosceles triangle of base **B**.

One side of the square must be parallel to the base of the isosceles triangle.

Base is the shortest side of the triangle

Input
First line contains **T**, the number of test cases.
Each of the following **T** lines contains 1 integer **B**.

Output
Output exactly T lines, each line containing the required answer.
Constraints
$1 \leq \mathbf{T} \leq 10^3$
$1 \leq \mathbf{B} \leq 10^4$
Sample Input

```
11
1
2
3
4
5
6
7
8
9
10
11
```

Sample Output

```
0
0
0
1
1
3
3
6
6
10
```

10

89. To protect people from evil, a long and tall wall was constructed a few years ago. But just a wall is not safe, there should also be soldiers on it, always keeping vigil. The wall is very long and connects the left and the right towers. There are exactly **N** spots (numbered 1 to **N**) on the wall for soldiers. The K^{th} spot is *K* miles far from the left tower and (**N**+1-*K*) miles from the right tower.

Given a permutation of spots *P* of {1, 2, ..., **N**}, soldiers occupy the **N** spots in that order. The $P[i]^{th}$ spot is occupied before the $P[i+1]^{th}$ spot. When a soldier occupies a spot, he is connected to his nearest soldier already placed to his left. If there is no soldier to his left, he is connected to the left tower. The same is the case with right side. A connection between two spots requires a wire of length equal to the distance between the two.

The realm has already purchased a wire of **M** miles long from Nokia, possibly the wire will be cut into smaller length wires. As we can observe, the total length of the used wire depends on the permutation of the spots *P*. Help the realm in minimizing the length of the unused wire. If there is not enough wire, output -1.

Input

First line contains an integer **T** (number of test cases, 1 ≤ **T** ≤ 10). Each of the next **T** lines contains two integers **N M**, as explained in the problem statement (1 ≤ **N** ≤ 30 , 1 ≤ **M** ≤ 1000).

Output

For each test case, output the minimum length of the unused wire, or -1 if the the wire is not sufficient.

Example

Input:

```
4
3 8
3 9
2 4
5 25
```

Output:

```
0
0
-1
5
12
```

90. I’m out of stories. For years I’ve been writing stories, some rather silly, just to make simple problems look difficult and complex problems look easy. But, alas, not for this one.

    You’re given a non empty string made in its entirety from opening and closing braces. Your task is to find the minimum number of “operations” needed to make the string stable. The definition for being stable is as follows:
    An empty string is stable.
    If S is stable, then {S} is also stable.
    If S and T are both stable, then ST (the concatenation of the two) is also stable.
    All of these strings are stable: {}, {}{}, and {{}{}}; But none of these: }{, {{}{, nor {}{.
    The only operation allowed on the string is to replace an opening brace with a closing brace, or visa-versa.

    Input
    Your program will be tested on one or more data sets. Each data set is described on a single line. The line is a non-empty string of opening and closing braces and nothing else. No string has more than 2000 braces. All sequences are of even length.
    The last line of the input is made of one or more ’-’ (minus signs.)

    Output
    For each test case, print the following line:
    k. N
    Where k is the test case number (starting at one,) and N is the minimum number of operations needed to convert the given string into a balanced one.

    Example
    Input:

    ```
    }{
    {}{}{}
    {{{}
    ---
    ```

    Output:

    ```
    1. 2
    2. 0
    3. 1
    ```

91. Bike loves looking for the second maximum element in the sequence. The second maximum element in the sequence of distinct numbers x_1, x_2, ..., x_k ($k > 1$) is such maximum element x_j, that the following inequality holds: .

    The lucky number of the sequence of distinct positive integers x_1, x_2, ..., x_k ($k > 1$) is the number that is equal to the bitwise excluding OR of the maximum element of the sequence and the second maximum element of the sequence.

You've got a sequence of distinct positive integers $s_1, s_2, ..., s_n$ ($n > 1$). Let's denote sequence $s_l, s_{l+1}, ..., s_r$ as $s[l..r]$ ($1 \le l < r \le n$). Your task is to find the maximum number among all lucky numbers of sequences $s[l..r]$.

Note that as all numbers in sequence s are distinct, all the given definitions make sence.

Input

The first line contains integer n ($1 < n \le 10^5$). The second line contains n distinct integers $s_1, s_2, ..., s_n$ ($1 \le s_i \le 10^9$).

Output

Print a single integer — the maximum lucky number among all lucky numbers of sequences $s[l..r]$.

Examples

input

Copy

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

output

Copy

```
7
```

input

Copy

```
5
9 8 3 5 7
```

output

Copy

```
15
```

Note

For the first sample you can choose $s[4..5] = \{4, 3\}$ and its lucky number is (4 *xor* 3) = 7. You can also choose $s[1..2]$.

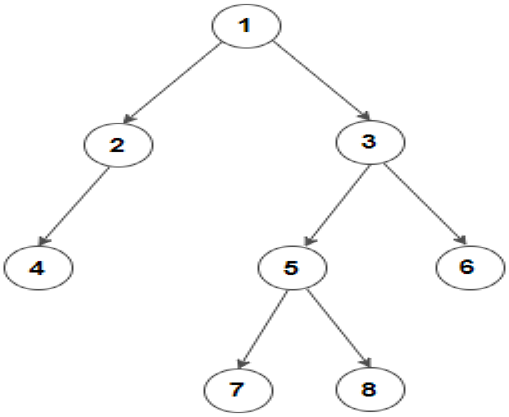
For the second sample you must choose $s[2..5] = \{8, 3, 5, 7\}$.

92. If A = (a1, ...,an) and B = (b1, ...,bm) are ordered lists, then A < B if ai = bi for and aj < bj or if ai =bi for and n < m. Write a program which returns -1,0, + 1 depending upon whether A < B, A = B or A > B. Assume you can compare atoms ai and bj.
93. Assume that n lists, n > 1, are being represented sequentially in the one dimensional array SPACE (1: m). Let FRONT(i) be one less than the position of the first element in the ith list and let REAR(i) point to the last element of the ith list, 1 i n. Further assume that REAR(i) <= FRONT(i + 1), 1 <=i<= n with FRONT(n + 1) = m. The functions to be performed on these lists are insertion and deletion.
94. Another kind of sparse matrix that arises often in numerical analysis is the tridiagonal matrix. In this square matrix, all elements other than those on the major diagonal and on the diagonals immediately above and below this one are zero. If the elements in the band formed by these three diagonals are represented row-wise in an array, B, with A (1,1) being

stored at B(1), Write a program to implement the sparse matrix described above using arrays and using linked list.

95. **Difference of Sum of Nodes at Odd and Even Levels**: Given a binary tree, find the difference between sum of all nodes present at odd levels and sum of all nodes present at even level.

    For example, consider below binary tree. The required difference is
    (1 + 4 + 5 + 6) – (2 + 3 + 7 + 8) = -4

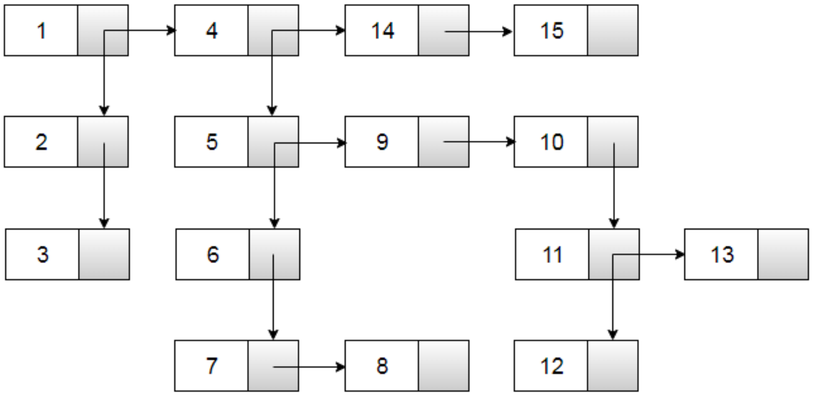


96. **Merge two BSTs into a Doubly Linked List** : Given two Binary Search Trees, merge them into a doubly linked list in sorted order... The idea is to convert each binary search tree into a doubly linked list first in sorted order and then merge both lists into a single doubly linked list in sorted order.

97. Multilevel Linked List

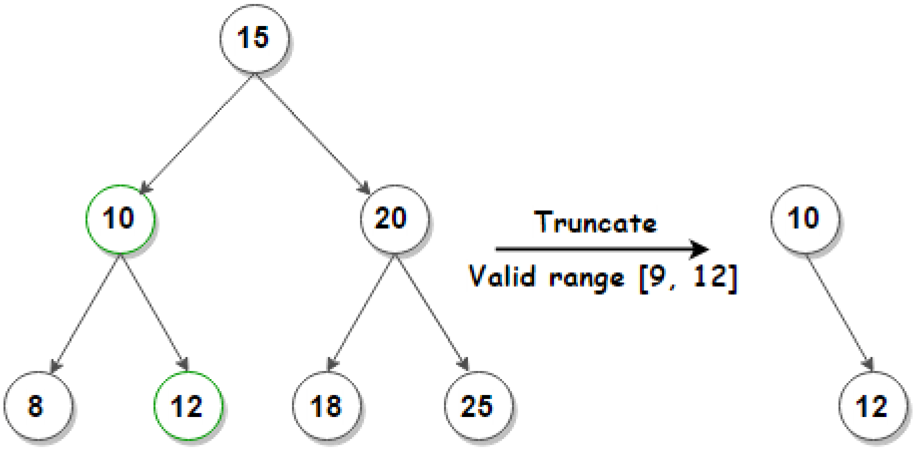
Given a list which can grow in both horizontal and vertical directions (right and down), flatten it into a singly linked list. The conversion should be in such a way that down node is processed before the next node for any node.
This list is similar to the standard linked list except that it has one extra field which points to a vertical list. The vertical list can have horizontal list attached to it and vice versa.

For example, consider below linked list



The Flattened list would be
1->2->3->4->5->6->7->8->9 ->10 -> 11 -> 12 ->13 -> 14 ->15->null

98. **Nodes Removal - Keys Outside the Valid Range :** Given a BST and a valid range of keys, remove nodes from BST that have keys outside the valid range. For example, consider BST shown on the left below.



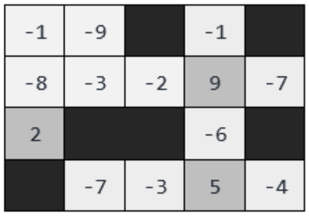
The idea is very simple, we traverse the tree in bottom-up fashion and truncate left and right subtree first before processing a node. Since we are doing a postorder traversal, it is possible that subtree rooted at current node is truncated and current node becomes a leaf node now. So, for each node we check:

- if its key fall within the valid range, nothing needs to be done.
- if root's key is smaller than the minimum allowed, we remove it and set root to root's right child
- if root's key is larger than the maximum allowed, we remove it and set root to root's left child

99. **Negative to Positive Values Conversion**: Given a M x N matrix of integers whose each cell can contain a negative, zero or positive value, determine the minimum number of passes required to convert all negative values in the matrix to positive.

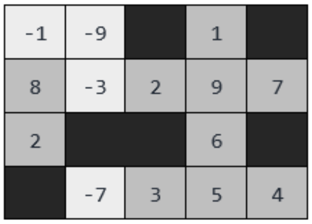
Only a non-zero positive value at cell (i, j) can convert negative values present at its adjacent cells (i-1, j), (i+1, j), (i, j-1),and (i, j+1) i.e. up, down, left and right.

For example, below matrix needs 3 passes as demonstrated below:



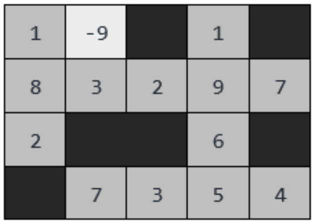
-1	-9	■	-1	■
-8	-3	-2	9	-7
2	■	■	-6	■
■	-7	-3	5	-4

Input Matrix



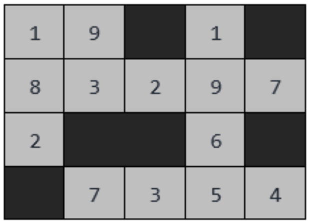
-1	-9	■	1	■
8	-3	2	9	7
2	■	■	6	■
■	-7	3	5	4

After end of Pass 1



1	-9	■	1	■
8	3	2	9	7
2	■	■	6	■
■	7	3	5	4

After end of Pass 2

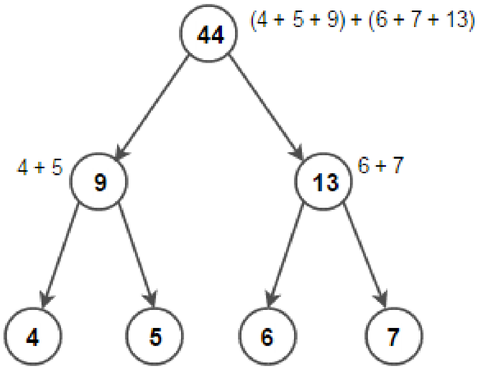


1	9	■	1	■
8	3	2	9	7
2	■	■	6	■
■	7	3	5	4

After end of Pass 3

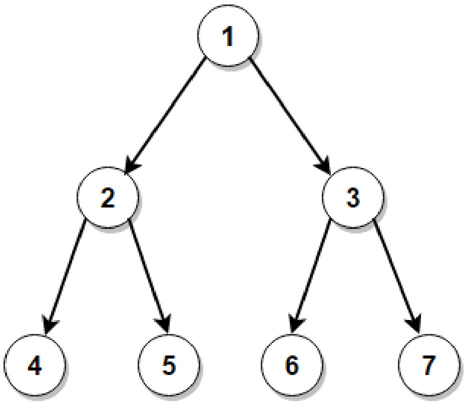
100. **Binary Tree to Sum Tree:** Given a binary tree, in-place convert it to its sum tree. In a sum tree, value at each node is equal to the sum of all elements present in its left and right subtree. The value of an empty node is considered as 0.

For example,



101. **Cousins Printing of Binary Tree**: Given a binary tree, print cousins of given node. Two nodes of binary tree are cousins of each other only if they have different parents but they have same level.

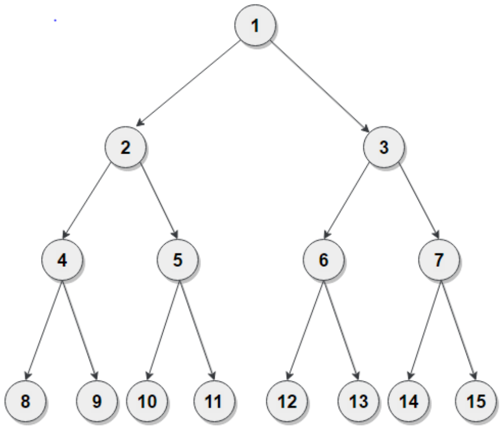
For Example, Consider tree below:



6 and 7 are cousins of 4 and 5
4 and 5 are cousins of 6 and 7

102. **Traversal in Alternate Fashion - Complete Binary Tree**: Given a perfect binary tree, write an efficient algorithm to print all nodes of it in specific order. We need to print nodes of every level in alternating

For example, there are two ways to print below tree



Variation 1: Print Top-Down : 1, 2, 3, 4, 7, 5, 6, 8, 15, 9, 14, 10, 13, 11, 12
Variation 2: Print Bottom : 8, 15, 9, 14, 10, 13, 11, 12, 4, 7, 5, 6, 2, 3, 1

103. **Multiple Linked Lists Merging:** Merge four linked list in such a way that the final linked list contains nodes from each of the linked lists alternatively.

For example:

Linked-List-1 : 1->5->9->13->NULL
Linked-List-2 : 2->6->10->14->18->NULL
Linked-List-3 : 3->7->11->15->19->23->NULL
Linked-List-4 : 4->8->NULL

Final Linked List : 1->2->3->4->5->6->7->8->9->10->11->13->14->15->18->19->23->NULL

104. **Group of K Nodes Reversal – Linked List**: Given a linked list, reverse every alternate group of k nodes in it where k is given positive integer. The idea is to traverse the linked list and consider every group of 2*k nodes at a time.

For example,

Input List: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> null

For k = 2

Output List: 2 -> 1 -> 3 -> 4 -> 6 -> 5 -> 7 -> 8 -> 10 -> 9 -> null

For k = 3

Output List: 3 -> 2 -> 1 -> 4 -> 5 -> 6 -> 9 -> 8 -> 7 -> 10 -> null

For k = 1

Output List: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> null

For k >= 10

Output List: 10 -> 9 -> 8 -> 7 -> 6 -> 5 -> 4 -> 3 -> 2 -> 1 -> null

105. **BSTs to Linked List:** Create a single linked list by using the elements of four binary search trees (BST) in level order fashion: First node of the linked list will be root of first BST, second node will be root of second BST and so on. Thereafter, the nodes of the linked list contain nodes at level-1 of the first BST, then nodes from level-1 of second BST and so on.

106. **Modified Linked List :** Create a linked list in such a way that, all values less than a threshold value appears on the left side of that and all values greater than the threshold shall appear on the right side of that. Extend the program and the application shall have the provision of deleting group of K nodes at a time.