Birla Institute of Technology and Science, Pilani

Second Semester 2017-2018, DSA (CS F211)

Lab Assignment #7

1. Venu joined an Analytics Company as an Intern. Since he had no prior work experience he was assigned a boring job. Every day he will be given a number and he has to calculate the median of all numbers he was given, till the present date. Venu being an ambitious guy doesn't want the company to think low of him, so he wanted to automate this process by writing a C Code which works efficiently. Imagine you are Venu and prove yourself by writing a C Code for the above question.

   Input: N (Total Number of days)
   N space separated integers denoting the value given to Venu on that date

   Output: N integers each on a single line denoting the median of all the numbers given to him till the ith date (i is the line number, *Example*: On the 3rd day you have to print the median of all the numbers given to you till the 3rd day (3rd day inclusive)). Print the median as a single double-precision number scaled to 1 decimal place (i.e., 15.7 format).

   Sample Input:
   6
   12
   4
   5
   3
   8
   7

   Sample Output:
   12.0
   8.0
   5.0
   4.5
   5.0
   6.0

   Explanation:

   1. $list = \{12\}, median = 12.0$
   2. $list = \{12, 4\} \rightarrow \{4, 12\}, median = \frac{12+4}{2} = 8.0$
   3. $list = \{12, 4, 5\} \rightarrow \{4, 5, 12\}, median = 5.0$
   4. $list = \{12, 4, 5, 3\} \rightarrow \{3, 4, 5, 12\}, median = \frac{4+5}{2} = 4.5$
   5. $list = \{12, 4, 5, 3, 8\} \rightarrow \{3, 4, 5, 8, 12\}, median = 5.0$
   6. $list = \{12, 4, 5, 3, 8, 7\} \rightarrow \{3, 4, 5, 7, 8, 12\}, median = \frac{5+7}{2} = 6.0$

Constrains: 1 <= N <= 10^5, 0 <= a_i(Value given on the ith day) <= 10^5
Time Limit: 2 seconds.

2. One of the standard problems in Operating Systems is of scheduling processes, such that their average waiting time is minimal. There are N processes and the ith process arrives at time ti and takes approximately di time to complete. You can think of scheduling problem as follows. Imagine you are the person responsible for doing KYC for PayTM and there are N people waiting in a line. The ith person arrives at time Ti and takes approximately Di time to get his KYC done. At any given time, you have the choice of picking any person (from all people who arrived) from the line and get his KYC done. You have to perform this action so as to make the average waiting time as minimal as possible. Waiting time is defined as follows:
Waiting Time = (Time at which the task is done) – (Time at which the person arrived)
Example:
A person joined the line at time t = 3 and left the line after getting his KYC done at t = 8
Then Waiting time = 8-3 = 5
Input:
N
In the Next N lines, the ith line denotes the arrival time of the person(Ti) and the time it takes to get his KYC done(Di)

Output:

Minimum Waiting Time(Integer Part, *Ex: 8.89 Ans: 8*)

Constraints:

0 <= Ti <= 10^9

1 <= Di <= 10^9
1 <= n <= 100000

Sample Input:

3

0 3

1 9

2 5

Sample Output:

8

Explanation:

Let's call the person arriving at time = 0 as *A*, time = 1 as *B* and time = 2 as *C*. By getting the KYC done in the order *A*, *C* and *B* we get the minimum average wait time to be (3+6+16)/3 = 25/3 = 8

3. Assume you are assigned with the job of controlling traffic on EarthV2.0. Unlike this planet EarthV2.0 is flat. So, traffic only flows from one side to another. Let's say there are N lanes and all lanes are well separated such that cars from one lane can't move to another lane and each car is associated with a number. Cars in a lane travel in such a manner that given any car with a number say n, is ahead of all cars with number greater than n. Let's say there is a junction such that all these N lanes join into 1 lane. So, you are assigned with job of controlling traffic or move cars which arrive at the junction to that single lane such that the above property holds in the 1 lane road (given any car with a number say n, is ahead of all cars with number greater than n). Also, each lane is such that only one car fits along its width.
   Input:
   N (No of Lanes), K (Cars in a lane)
   N lines, each line contains K numbers space separated, corresponding to each car arriving at the junction and which belongs to ith lane (numbers are arranged according to the property or are sorted).
   Output: N*K space separated integers, such that the ith number denotes the number corresponding to the ith car which moved into the single lane
   Sample Input:
   3 3
   10 15 21
   5 13 19
   20 25 35
   Sample Output:
   5 10 13 15 19 20 21 25 35
   Time Complexity: O(NK*LogN)

4. Write a recursive function convertTreeToList(Node root) that takes a Binary Search Tree and rearranges the internal pointers so as to make a circular doubly linked list out of the tree nodes. The "previous" pointers of the linked list should be stored in the "small" field (left pointer) and the "next" pointers should be stored in the "large" field (right pointer) of the linked list or the "small" pointer should play the role of "previous" and the "large" pointer should play the role of "next". The list should be arranged so that the nodes are in increasing order. Return the head pointer to the new list. The operation can be done in O(n) time -- essentially operating on each node once. Basically, take figure-1 as input and rearrange the pointers to make figure-2.
   Note: You have to hard-code a Binary Search Tree and convert it into Doubly linked list

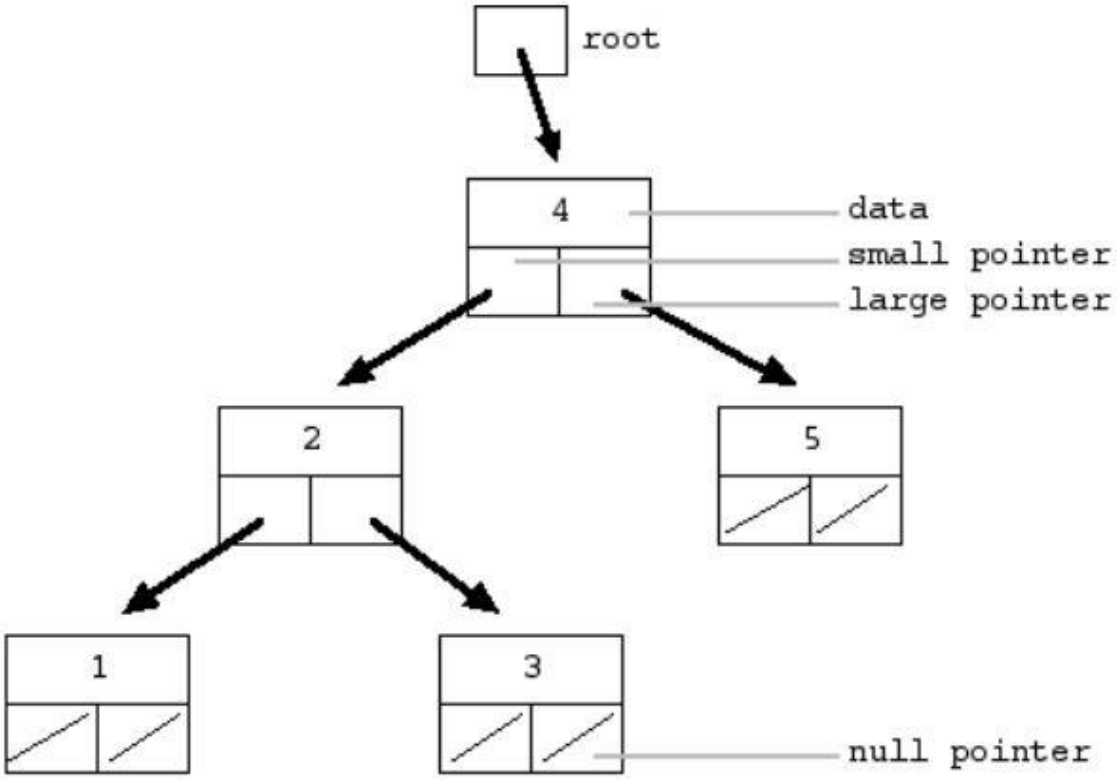


Figure -1

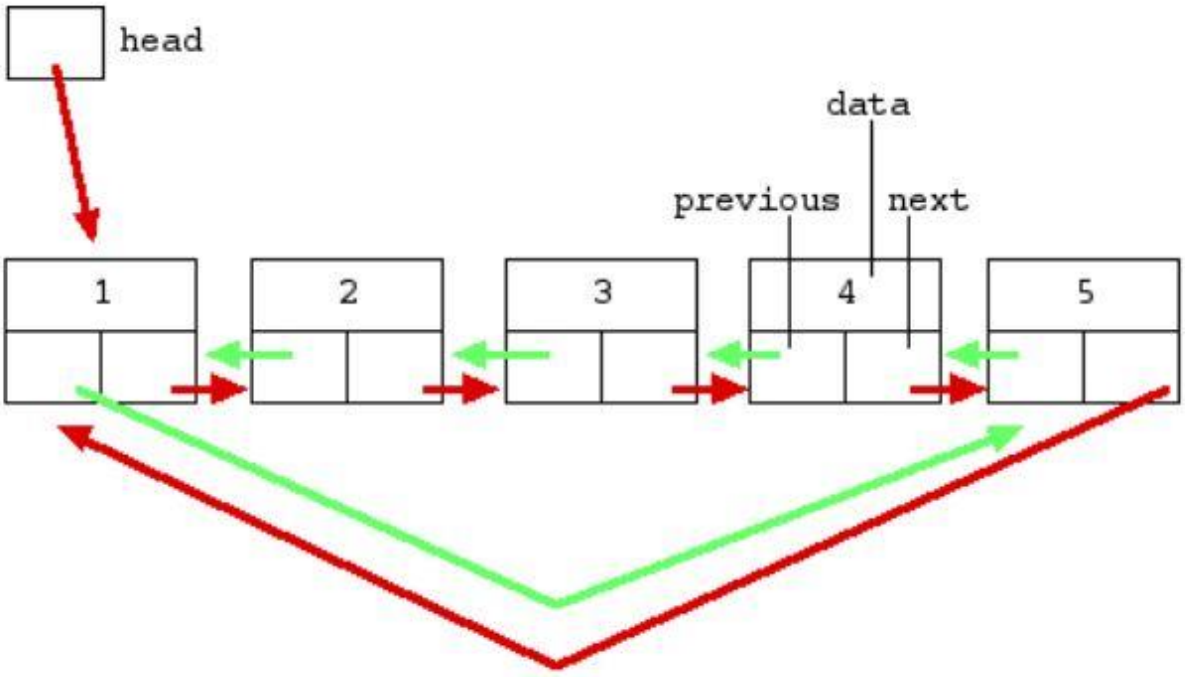
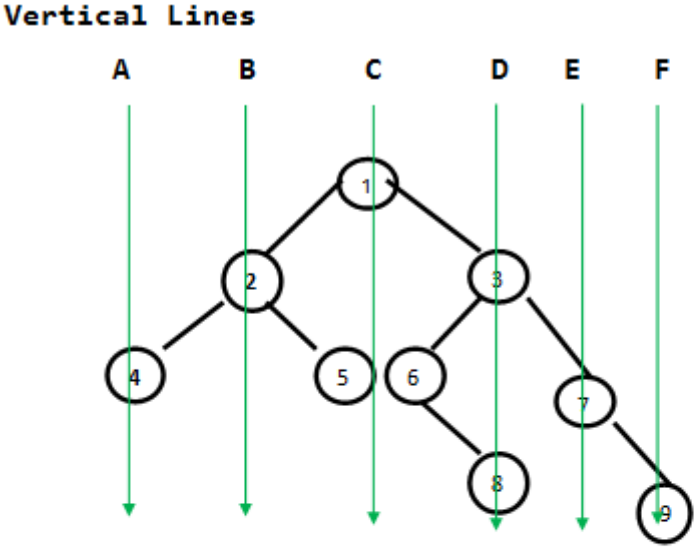


Figure-2

5. Merge Sort with Linked Lists: For sorting a Linked List, merge sort is always preferred because algorithms like quick sort are slow due to high access time of the list.
   Given a sequence of numbers, take them into a linked list and sort them.
   Input:
   N – number of integers to be sorted
   Sequence of n integers
   Output:
   Sorted sequence
   Constraints:
   $N < 10^5$
   Time Complexity: O(NlogN)
   Sample Input:
   5
   2 3 4 5 1
   Sample Output:
   1 2 3 4 5

6. You are given a binary tree. Print the key value of nodes in vertical order traversal (Explained in the figure below).




```
Vertical order traversal is:
A- 4
B- 2
C- 1 5 6
D- 3 8
E- 7
F- 9
```

Input: m – number of edges. And the value of root node.
Followed by m lines which contain 'a b L/R' L – left child and R – right child

Sample input: The above example will be given as an input in the form

8 1

1 2 L

2 4 L

2 5 R

1 3 R

3 6 L

6 8 R

3 7 R

7 9 R

Sample Output: In the figure

7. Mr. Manchali likes climbing trees. He encountered a very strange tree in which at each node, it may branch into at most two branches. The strength of each node is a given by a value between 1 to 100 (All have unique values). He wants to determine if climbing that tree is easy or not. He determines that from a few conditions:

a. Each node branches into nodes that are equal or lesser in strength.
b. All branches are as far left as possible.
c. If maximum height of the tree is h, all nodes till h-2 height must have 2 branches.

If these three conditions are satisfied, the tree is easy to climb.

Input:

m – number of edges. And the value of root node.

Followed by m lines which contain 'a b L/R' L – left branch and R – right branch.

Output:

"Easy" or "Difficult"

Time Complexity: O(m)

Sample Input:

5 10

10 9 L

10 8 R

9 7 L

9 6 R

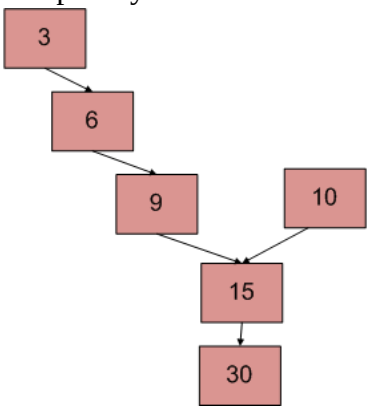
8 5 L

Sample Output:

Easy

8. A railway contractor is assigned the job to make a single long stretch railway line by riveting multiple lines of given length. Cost of riveting two lines is the sum of lengths of each line. Due to obvious reasons, the funds are limited and the contractor needs this job to be done in minimum possible price possible. Given the lengths of lines provided to him, determine the cost of the process.
Input:
N – number of lines
N lengths of the lines
Output:
Minimum cost
Constraints:
$N<10^5$
Length of each line $< 10^9$
Complexity: O(NlogN)
Sample Input:
4
4 2 3 6
Sample Output:
29
Explanation:
2 and 3 are riveted: 5
5 and 4 are riveted: 9
9 and 6 are riveted: 15
Cost = 15+9+5 = 29

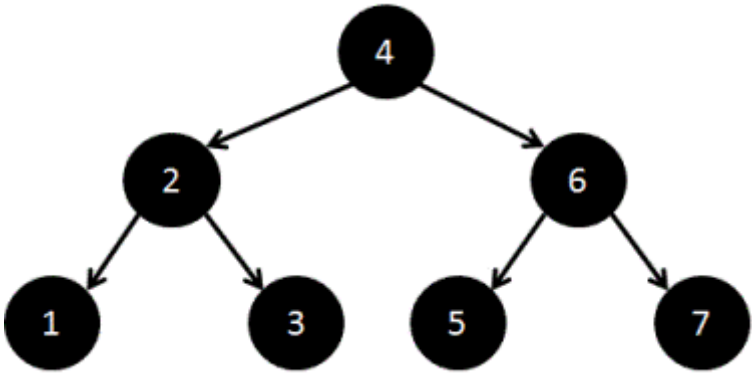
9. A) Given two linked lists of size M and size N find intersection of 2 linked lists in O(M+N) time complexity and O(1) space complexity.



Here, 15 is the intersection point

B) Given a doubly linked list in sorted order with previous and next nodes. Convert the doubly linked list to a binary search tree with left as previous node and right as next node.
Consider the list below:

The list should be converted to following BST:



10. Use prim's Algorithm to find the Minimum Spanning tree of a graph.
A minimum spanning tree (MST) or minimum weight spanning tree is a subset of the edges of a connected, edge-weighted (un)directed graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight. That is, it is a spanning tree whose sum of edge weights is as small as possible.

Input: First line has two integers, N denoting the number of nodes in the graph and, M denoting the number of edges in the graph.
The next M lines each consist of three space separated integers x, y, r where x and y denote the two nodes between which the undirected edge exists, r denotes the length of edge between the corresponding nodes.

Output: Sum of weights in the Minimum Spanning Tree

Constraints:
2 <= N <= 3000
1 <= M <= (N*(N-1))/2
1 <= x,y <= N
0 <= r <= 10^5

Sample Input:
5 6
1 2 3
1 3 4
4 2 6

5 2 2
2 3 5
3 5 7
1

Sample Output:

15