1. Huffman Coding

Huffman coding is a technique of compressing data to reduce its size without losing any of the details. The following code snippet is Incomplete. Order the list of options
Huffman coding algorithm.

```
 Create a priority queue Q consisting of each unique character.
 Sort it in the ascending order of their frequencies.
 for all the unique characters:
    create a newNode
    //Step 1
    //Step 2
    //Step 3
    //Step 4
return rootNode
```

Option 1: extract minimum value from Q and assign it to the left child of the newNode.
Option 2: insert this newNode into the tree.
Option 3: calculate the sum of these two minimum values and assign it to the value of the newNode.
Option 4: extract minimum value from Q and assign it to right child of the newNode.

Option 1: extract minimum value from Q and assign it to the left child of the newNode.
Option 2: insert this newNode into the tree.
Option 3: calculate the sum of these two minimum values and assign it to the value of the newNode.
Option 4: extract minimum value from Q and assign it to right child of the newNode.

Pick **ONE** option

- Step 1-> Option 1 Step 2-> Option 2 Step 3-> Option 3 Step 4-> Option 4
- Step 1-> Option 1 Step 2-> Option 3 Step 3-> Option 2 Step 4-> Option 4
- Step 1-> Option 1 Step 2-> Option 2 Step 3-> Option 4 Step 4-> Option 3
- Step 1-> Option 1 Step 2-> Option 4 Step 3-> Option 3 Step 4-> Option 2

Clear Selection

2. Number of Swaps II

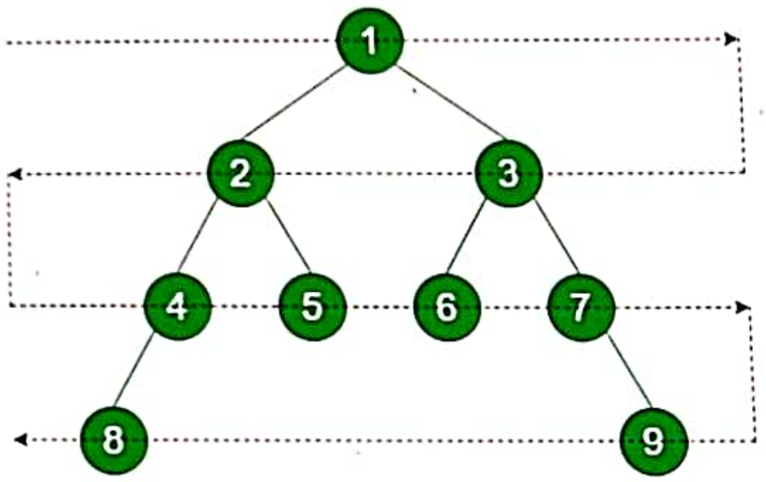
Find the minimum number of swaps required to convert *arr1* to *arr2.*

arr1 = [7, 6, 1, 3, 4, 9, 8, 2, 5]
arr2 = [3, 1, 8, 5, 4, 9, 2, 6, 7]

Pick **ONE** option

- ◯ 9
- ◯ 5
- ◯ 7
- ◯ 4

3. Complete the Algorithm



Consider the tree above with the path implemented in the code below. The expected output is: *1 3 2 4 5 6 7 9 8*

Choose the answer that corrects the logical errors in the code.

Note: Assume a class named *TreeNode* is written correctly.

Consider the tree above with the path implemented in the code below. The expected output is: *1 3 2 4 5 6 7 9 8*

Choose the answer that corrects the logical errors in the code.

Note: Assume a class named *TreeNode* is written correctly.

```
vector<vector<int>> zigzagTree(TreeNode* root) {
        vector<vector<int>> ans;
        if(root==NULL){
            return ans;
        }
        bool flag=false;
        deque<TreeNode*> d1;
        d1.push_back(root);
        while(!d1.empty()){
            int n = d1.size();
            vector<int> ar;
            ans.push_back(ar);
            while(n--){
                TreeNode *cur=d1.front();
                d1.pop_front();
                if(cur->left!=NULL){
                    d1.push_back(cur->left);
```

```
vector<vector<int>> zigzagTree(TreeNode* root) {
      vector<vector<int>> ans;
      if(root==NULL){
         return ans;
      }
      bool flag=false;
      deque<TreeNode*> d1;
      d1.push_back(root);
      while(!d1.empty()){
         int n = d1.size();
         vector<int> ar;
         ans.push_back(ar);
         while(n--){
            TreeNode *cur=d1.front();
            d1.pop_front();
            if(cur->left!=NULL){
               d1.push_back(cur->left);
            }
            if(cur->right!=NULL){
               d1.push_back(cur->right);
            }
         }
         flag = !flag;
      }
      return ans;
   }
```

Pick ONE option

- In the second while loop, after fetching the current TreeNode ('cur'), insert its value into the vector named 'ar'. Move the line "ans.push_back(ar)" to below the end of the second while loop.

- In the second while loop, after fetching the current TreeNode ('cur'), insert its value into the vector named 'ar'. After that loop, do not change the value of the 'flag' (keep it as it is). Move the line "ans.push_back(ar)" to below the end of the second while loop.

- After the declaration of 'ar', insert the elements stored in 'd1' into 'ar'. The insertion should be done from 0 to d1.size() if flag is true and from d1.size() - 1 to 0 if flag is false.

- After the declaration of 'ar', insert the elements stored in 'd1' into 'ar'. The insertion should be done from 0 to d1.size() if flag is false and from d1.size() - 1 to 0 if flag is true.

Clear Selection

4. Rank of Matrix

What is the rank of the given matrix:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 8 & 4 & 6 & 0 \\ 2 & 0 & 0 & 0 \\ 8 & 0 & 6 & 0 \end{bmatrix}$$

Pick **ONE** option

- ○ 2
- ○ 3
- ○ 4
- ○ 0

Clear Selection

5. Compare Scores

A student received the following marks for exams in 2 subjects. Which subject showed better relative performance? (Assume scores follow a normal distribution)

Subject.	**Marks obtained (out of 100)**	**Mean Score**	**Standard Deviation**
Maths	70	60	15
Science	72	68	6

Pick **ONE** option

Pick ONE option

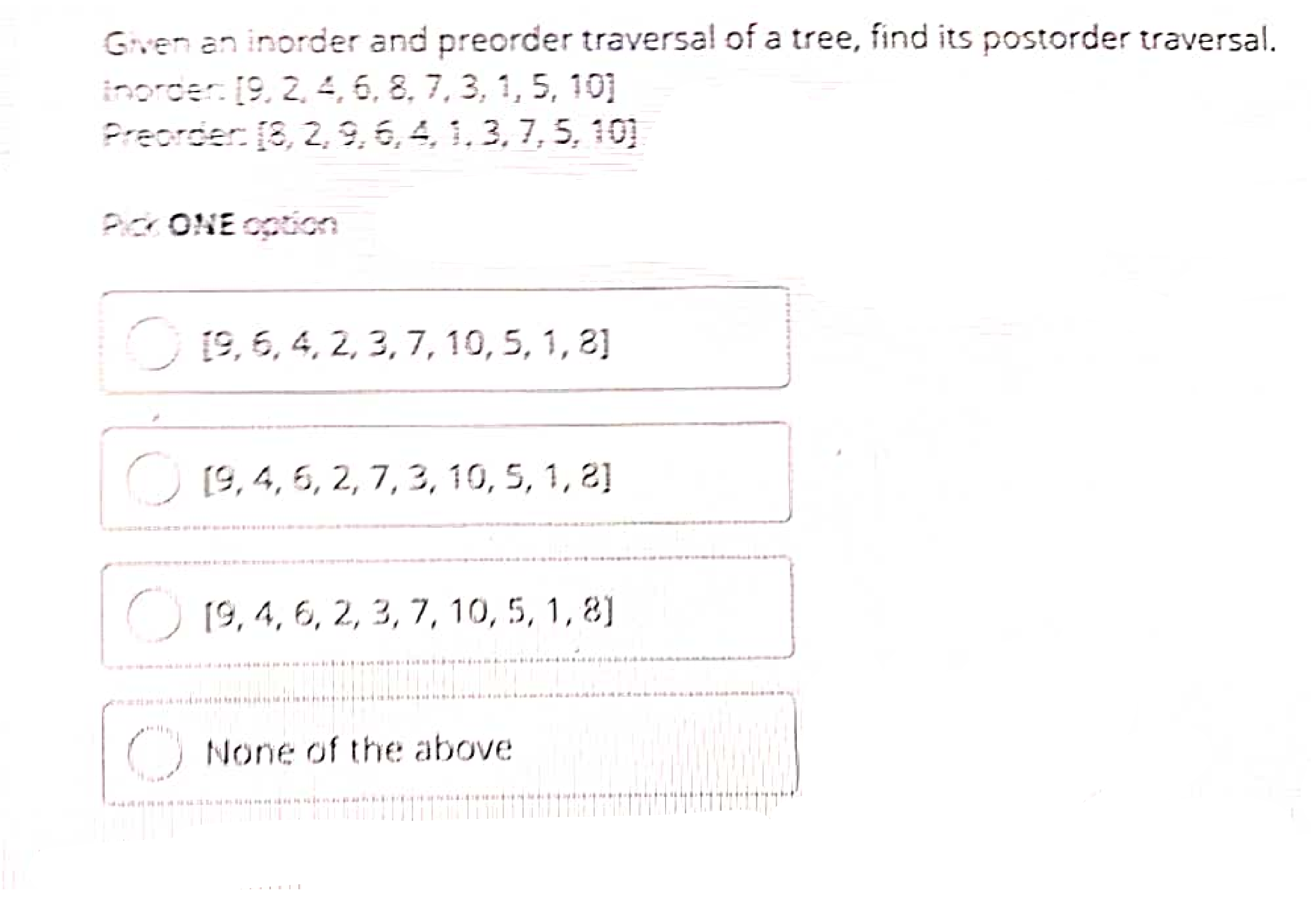
Maths

Science

The performance is the same in both subjects.

Clear Selection

6. Inorder and Preorder



Given an inorder and preorder traversal of a tree, find its postorder traversal.
Inorder: [9, 2, 4, 6, 8, 7, 3, 1, 5, 10]
Preorder: [8, 2, 9, 6, 4, 1, 3, 7, 5, 10]

Pick **ONE** option

- [9, 6, 4, 2, 3, 7, 10, 5, 1, 8]
- [9, 4, 6, 2, 7, 3, 10, 5, 1, 8]
- [9, 4, 6, 2, 3, 7, 10, 5, 1, 8]
- None of the above

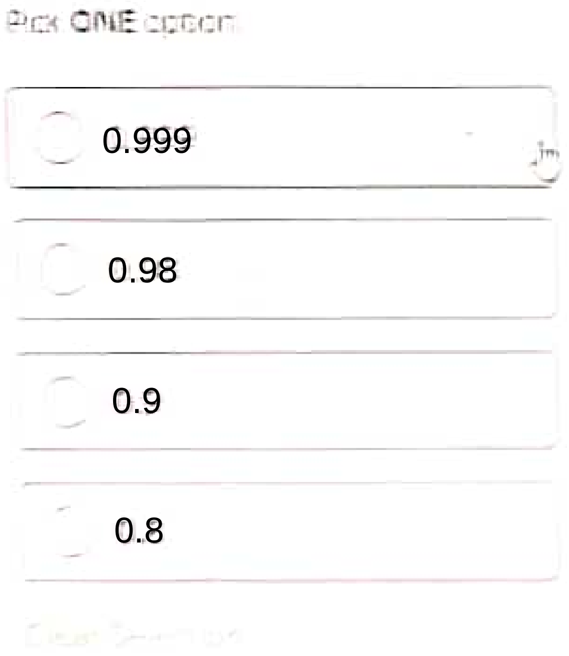
7. Population Proportion

In 1990, about 50% of Americans considered the low quality of education as a serious problem for the country. What is the chance that the sample propo
total population with an accuracy of 3 percent for a random poll of 1,500 Americans?

Quantile table for standard normal distribution:

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993



Pick **ONE** option

○ 0.999

○ 0.98

○ 0.9

○ 0.8

Clear Selection

8. Stack Contents

Starting with an empty stack, the following operations are performed. What is the final state of the stack?

```
Push(78)
Pop()
Push(44)
Push(12)
Push(18)
Pop()
Push(19)
Pop()
Pop()
Push(12)
Push(28)
```

Pick **ONE** option

- ○ [44, 12, 19]
- ○ [78,44, 12]
- ○ [19, 12, 28]
- ○ [44, 12, 28]

Clear Selection

9. Create Subarrays

There is an array of integers arr[n], an integer sum k, and the number of subarrays to create, q. One can perform two operations at any element of

1. Increase the i^{th} element of arr by 1. i.e. arr[i] = arr[i]+1
2. Decrease the i^{th} element of arr by 1. i.e. arr[i] = arr[i]-1

A desirable subarray is defined as one whose sum is equal to the integer value k.
You want to determine the minimum number of operations required to convert arr to q contiguous desirable subarrays.
Select the best data structure to solve the problem.

Note:

- $0 \le n \le 10^3$ (size of the array)
- $-10^5 \le a[i] \le 10^5$

Pick **ONE** option

- ○ Hash map
- ○ Stack
- ○ Queue
- ○ Priority queue

Clear Selection

10. Max Stack

Find the maximum length the stack will have if it performs the following operations.

1. Push(10);
2. Pop();
3. Push(17);
4. Push(29);
5. Push(6);
6. Pop();
7. Push(10);
8. Push(77);
9. Pop();

Pick **ONE** option

- ◯ 3
- ◯ 5
- ◯ 4
- ◯ None of the above

11. ML

How do you drop a column named "column_name" from a DataFrame called "df" in Pandas?

Pick **ONE** option

- ◯ df.drop_column("column_name")
- ◯ df.drop("column_name", axis=1)
- ◯ df.remove_column("column_name")
- ◯ df.remove("column_name", axis=1)

12. "Maximum Likelihood Estimate 2"

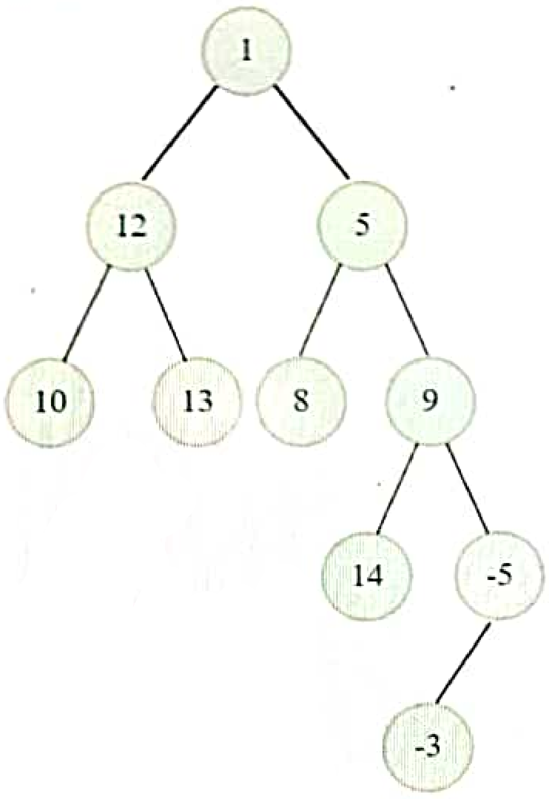
The data $x_1, \ldots, x_n$ is drawn from an exponential distribution *exp(λ)*. Find the maximum likelihood estimate for λ.

Pick **ONE** option

- ○ $1/(x_1+\cdots+x_n)$
- ○ $(x_1+\cdots+x_n)$
- ○ $(x_1+\cdots+x_n)/n$
- ○ $n/(x_1+\cdots+x_n)$

13. View of the Tree

Find the bottom view of the following binary tree.



Pick **ONE** option

- ◯ 10, 12, 13, 1, 8, 14, -3, -5
- ◯ 10, 12, 8, 14, -3, -5
- ◯ 10, 12, 13, 1, 8, 14, 9, -3, -5
- ◯ 10, 12, 1, 5, 9, -5

14. ML

Which method is used to check if a DataFrame contains any missing values in Pandas?

Pick **ONE** option

- is_empty()
- is_missing()
- is_null()
- isna()

15. Circular Linked List

A circular linked list can be used to implement

Pick **ONE** option

- ◯ A stack
- ◯ A queue
- ◯ Both
- ◯ Neither

16. Error

Which of the choices are true?

Pick **ONE OR MORE** options

- ☐ A type 1 error is when you incorrectly reject a true null hypothesis. A type 2 error is when you don't reject a false null hypothesis.
- ☐ A type 1 error is when you don't reject a false null hypothesis. A type 2 error is when you incorrectly reject a true null hypothesis.
- ☐ A type 1 error is called a false positive, whereas a type 2 error is called a false negative.
- ☐ A type 1 error is called a false negative, whereas a type 2 error is called a false positive.

17. Logic Puzzles 4

There are two horizontal rows of five students each facing each other. The rows are called Row-1 and Row-2. The ten students are named from

The following conditions apply:

1. The extremes are occupied by A, B, C and D.
2. E stands at the center of Row-1.
3. F stands to the opposite of E in Row-2.
4. J stands to the immediate right of A.
5. J stands in Row-1.
6. A and B do not stand in the same row.
7. A and B are not opposite to each other.
8. I stands between F and B.

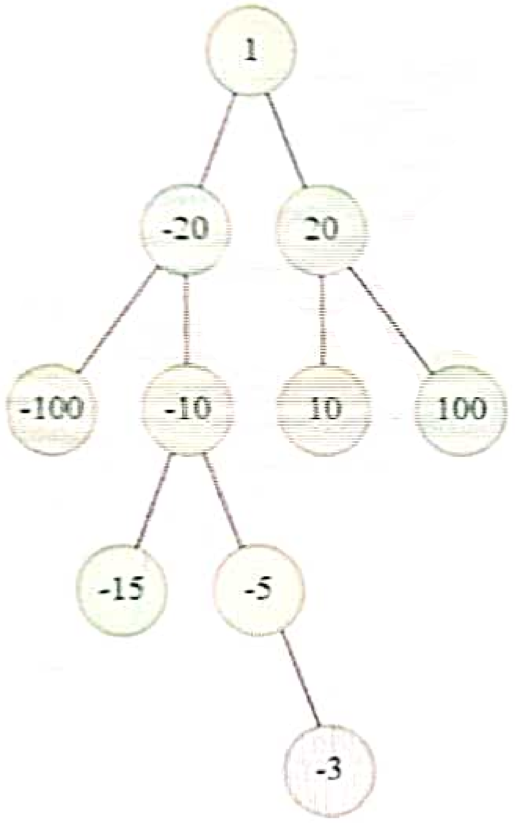
If it is given that H and C stand in the same row, then which of the following pairs definitely stand in the same row?

Pick **ONE** option

- ◯ G and D
- ◯ G and C
- ◯ H and D
- ◯ D and C

18. BST II

Which of the following is the postorder successor of node -100 in the given BST?



Pick **ONE** option

- ○ 100
- ○ -15
- ○ -3
- ○ -20

19. Compressing Array v2

Given an array of integers, *a,* in one operation one can select any two adjacent elements and replace them with their product. This operation can only be applied if the product of those adjacent elements is less than or equal to *k.*

The goal is to reduce the length of the array as much as possible by performing any number of operations. Return that minimum size.

Exampl
Let array *a = [2, 3, 3, 7, 3, 5]* and *k = 20*

This is the list of operations that will give us the smallest array (1-based indexing):

- Merge the elements at indices (1, 2), resulting array will be - *[6, 3, 7, 3, 5]*
- Merge the elements at indices (1, 2), resulting array will be - *[18, 7, 3, 5]*
- Merge the elements at indices (3, 4), resulting array will be - *[18, 7, 15]*

Hence, the answer is 3.

Function Description
Complete the function *getMinLength* in the editor below.

getMinLength has the following parameters:
  int a[n]: an array of integers
  int k: the constraint of the operation

getMinLength has the following parameters:

int a[n]: an array of integers

int k: the constraint of the operation

Returns

int: the minimum length of the array after performing any number of operations

Constraints

- $1 \le n \le 2.10^5$
- $1 \le a[i] \le k \le 10^9$

▶ Input Format For Custom Testing

▼Sample Case 0

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       --------
5       →   n = 5
1
3
2
5
4       →   a = [1,3,2,5,4]
6       →   k = 6
```

Sample Output

```
3
```

20. Subsequence Length v2

As an assignment, some students at HackerLand High School are to determine the length of the longest subsequence with non-zero bitwise AND for an array *arr* of *n* integers.

Bitwise AND of a sequence is defined as the bitwise AND of all the elements of the sequence. Bitwise AND of a sequence of length one is the element itself.

Notes:

- A subsequence is a sequence that can be derived from the given sequence by deleting zero or more elements without changing the order of the remaining elements.
- Assuming '&' is the symbol for a bitwise AND, the bitwise AND of the sequence [1, 3, 5, 7, 3] is (1 & 3 & 5 & 7 & 3) = 1

Example

n = 5, *arr* = [7, 4, 11, 8, 3].

Consider some subsequences and their bitwise AND.

Subsequence	Bitwise AND	Length
[7, 4, 3]	0	Not considered
[11, 3]	3	2
[11, 8]	8	2

Here, the longest subsequence that has a non-zero bitwise AND is [7, 11, 3] has a length of 3, so return 3.

Function Description

Complete the function *getMaxLength* in the editor below.

getMaxLength has the following parameter(s):

int arr[n]: the array

Returns

int: the length of the longest subsequence with a non-zero bitwise AND

Constraints

- $1 \le n \le 10^5$
- $1 \le arr[i] \le 10^9$, $0 \le i < n$