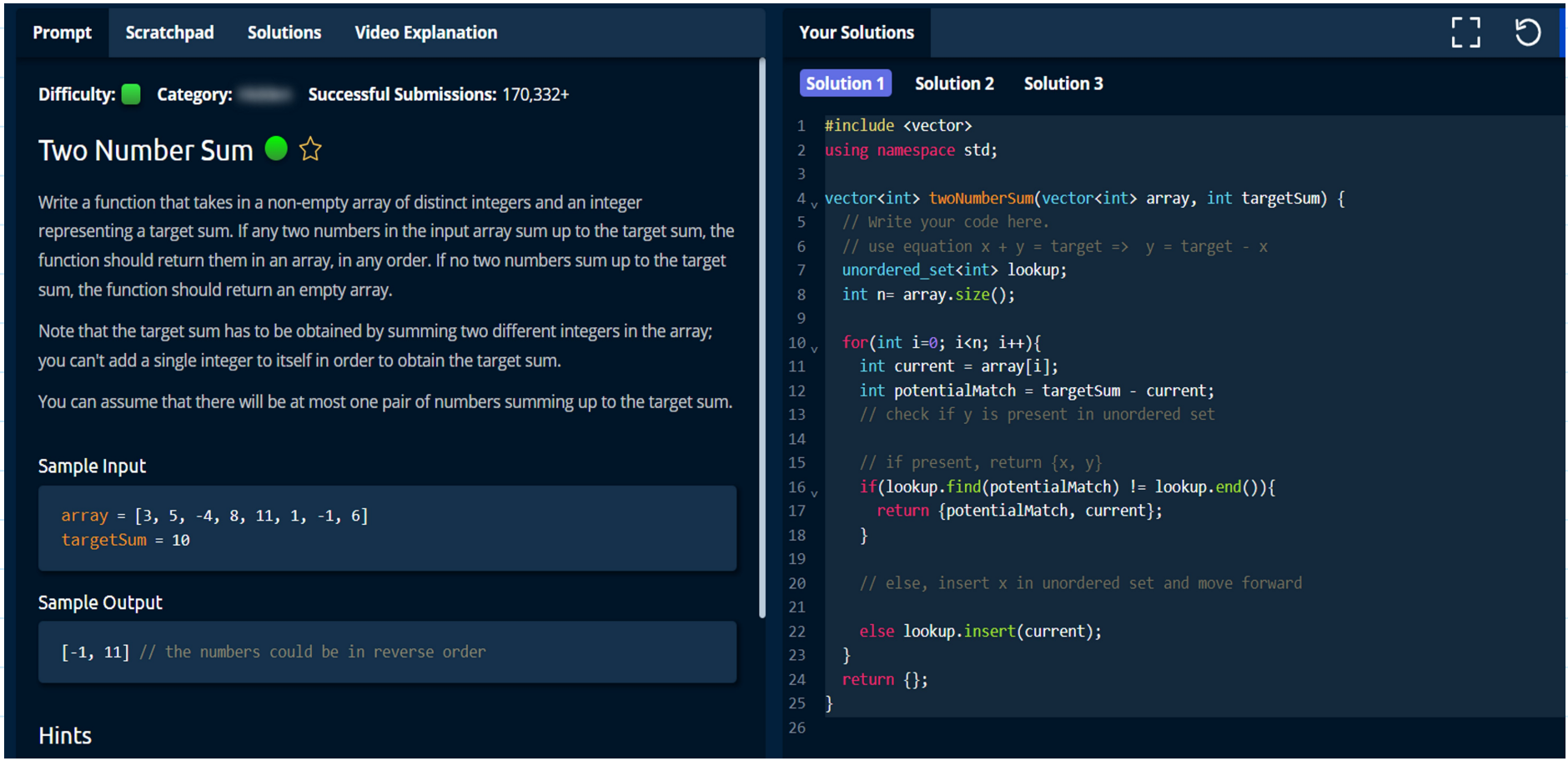
1

08 August 2023 08:04 PM



Prompt | Scratchpad | Solutions | Video Explanation

Difficulty: Category: [redacted] Successful Submissions: 170,332+

Two Number Sum

Write a function that takes in a non-empty array of distinct integers and an integer representing a target sum. If any two numbers in the input array sum up to the target sum, the function should return them in an array, in any order. If no two numbers sum up to the target sum, the function should return an empty array.

Note that the target sum has to be obtained by summing two different integers in the array; you can't add a single integer to itself in order to obtain the target sum.

You can assume that there will be at most one pair of numbers summing up to the target sum.

Sample Input

```
array = [3, 5, -4, 8, 11, 1, -1, 6]
targetSum = 10
```

Sample Output

```
[-1, 11] // the numbers could be in reverse order
```

Hints

Your Solutions

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
using namespace std;

vector<int> twoNumberSum(vector<int> array, int targetSum) {
  // Write your code here.
  // use equation x + y = target =>  y = target - x
  unordered_set<int> lookup;
  int n= array.size();

  for(int i=0; i<n; i++){
    int current = array[i];
    int potentialMatch = targetSum - current;
    // check if y is present in unordered set

    // if present, return {x, y}
    if(lookup.find(potentialMatch) != lookup.end()){
      return {potentialMatch, current};
    }

    // else, insert x in unordered set and move forward

    else lookup.insert(current);
  }
  return {};
}

```

2

08 August 2023 08:20 PM

Prompt | Scratchpad | Solutions | Video Explanation

Difficulty: **Category:** **Successful Submissions:** 149,720+

Validate Subsequence

Given two non-empty arrays of integers, write a function that determines whether the second array is a subsequence of the first one.

A subsequence of an array is a set of numbers that aren't necessarily adjacent in the array but that are in the same order as they appear in the array. For instance, the numbers `[1, 3, 4]` form a subsequence of the array `[1, 2, 3, 4]`, and so do the numbers `[2, 4]`. Note that a single number in an array and the array itself are both valid subsequences of the array.

Sample Input

```
array = [5, 1, 22, 25, 6, -1, 8, 10]
sequence = [1, 6, -1, 10]
```

Sample Output

```
true
```

Your Solutions

Solution 1 | Solution 2 | Solution 3

```cpp
using namespace std;

bool isValidSubsequence(vector<int> array, vector<int> sequence) {
  // Write your code here.
  int arrSize = array.size();
  int seqSize = sequence.size();
  for(int i=0,j=0; i<arrSize && j<seqSize;){
      if(array[i]==sequence[j])
      {
        i++; j++;
            if(j>=seqSize) return true;
      }
    else{
      i++;
    }
  }
  return false;
}

```

3

10 August 2023 06:14 PM

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
#include <cstdlib>
using namespace std;

vector<int> sortedSquaredArray(vector<int> array) {
  // Write your code here.

  int n = array.size();
  vector<int> newArray(n);

  for(int start=0,end=n-1, insertIndex=n-1 ; start<=end; insertIndex-- ){
    if(abs(array[start])> abs(array[end])){
      newArray[insertIndex] = array[start]*array[start];
      start++;
    }
    else{
      newArray[insertIndex] = array[end]*array[end];
      end--;
    }
  }
  return newArray;
}
```

Prompt | Scratchpad | Solutions | Video Explanation

Difficulty: **Category:** [redacted] **Successful Submissions:** 89,632+

Sorted Squared Array

Write a function that takes in a non-empty array of integers that are sorted in ascending order and returns a new array of the same length with the squares of the original integers also sorted in ascending order.

Sample Input

```
array = [1, 2, 3, 5, 6, 8, 9]
```

Sample Output

```
[1, 4, 9, 25, 36, 64, 81]
```

```cpp
#include <vector>
using namespace std;

vector<int> sortedSquaredArray(vector<int> array) {
  // Write your code here.
  int n = array.size();
  vector<int> newArr;

  for(int i=0; i<n; i++){
    int s = array[i]*array[i];
    // if(array[i]<0)
    newArr.push_back(s);
  }
  sort(newArr.begin(), newArr.end());
  return newArr;


}
```

4

10 August 2023 06:14 PM

Tournament Winner

There's an algorithms tournament taking place in which teams of programmers compete against each other to solve algorithmic problems as fast as possible. Teams compete in a round robin, where each team faces off against all other teams. Only two teams compete against each other at a time, and for each competition, one team is designated the home team, while the other team is the away team. In each competition there's always one winner and one loser; there are no ties. A team receives 3 points if it wins and 0 points if it loses. The winner of the tournament is the team that receives the most amount of points.

Given an array of pairs representing the teams that have competed against each other and an array containing the results of each competition, write a function that returns the winner of the tournament. The input arrays are named `competitions` and `results`, respectively. The `competitions` array has elements in the form of `[homeTeam, awayTeam]`, where each team is a string of at most 30 characters representing the name of the team. The `results` array contains information about the winner of each corresponding competition in the `competitions` array. Specifically, `results[i]` denotes the winner of `competitions[i]`, where a `1` in the `results` array means that the home team in the corresponding competition won and a `0` means that the away team won.

It's guaranteed that exactly one team will win the tournament and that each team will compete against all other teams exactly once. It's also guaranteed that the tournament will always have at least two teams.

Sample Input

```
competitions = [
  ["HTML", "C#"],
  ["C#", "Python"],
  ["Python", "HTML"],
]
results = [0, 0, 1]
```

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
#include <string>
using namespace std;

string tournamentWinner(vector<vector<string>> competitions,
                        vector<int> results) {
  // Write your code here.

  map<string, int> myMap;

    int n = competitions.size();
    for(int i=0; i<n; i++){
      if(results[i]==1){
            std::string str = competitions[i][0];
            auto itr = myMap.find(competitions[i][0]);

            if(itr!=myMap.end())
                  (*itr).second +=3;
            else
                  myMap.insert( {competitions[i][0], 3});
      }
      else{
              std::string str = competitions[i][1];
                auto itr = myMap.find(competitions[i][1]);
            if(itr!=myMap.end())
                  (*itr).second +=3;
            else
                  myMap.insert( {competitions[i][1], 3});
      }

      }


  auto pr = std::max_element(myMap.begin(),myMap.end(),
  [] (const std::pair<string,int>& a, const std::pair<string,int>& b)
  { return a.second < b.second; });

  return pr->first;
}

```

5

10 August 2023 06:14 PM

Difficulty: **Category:** **Successful Submissions:** 59,739+

Non-Constructible Change

Given an array of positive integers representing the values of coins in your possession, write a function that returns the minimum amount of change (the minimum sum of money) that you **cannot** create. The given coins can have any positive integer value and aren't necessarily unique (i.e., you can have multiple coins of the same value).

For example, if you're given `coins = [1, 2, 5]`, the minimum amount of change that you can't create is `4`. If you're given no coins, the minimum amount of change that you can't create is `1`.

Sample Input

```
coins = [5, 7, 1, 1, 2, 3, 22]
```

Sample Output

```
20
```

Solution 1 Solution 2 Solution 3

```cpp
#include <vector>
using namespace std;

bool isSumInArray(vector<int> array, int targetSum){

}

int nonConstructibleChange(vector<int> coins) {
  // Write your code here.

  int n = coins.size();

  sort(coins.begin(), coins.end());
  int change =0;
  for(int i=0; i<n; i++){
    if(coins[i]>(change+1))
        return change+1;
    change += coins[i];
  }




  return change+1;
}

```

6

10 August 2023 06:14 PM

Bubble Sort

Write a function that takes in an array of integers and returns a sorted version of that array. Use the Bubble Sort algorithm to sort the array.

If you're unfamiliar with Bubble Sort, we recommend watching the Conceptual Overview section of this question's video explanation before starting to code.

Sample Input

```
array = [8, 5, 2, 9, 5, 6, 3]
```

Sample Output

```
[2, 3, 5, 5, 6, 8, 9]
```

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order. This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high.

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
using namespace std;

vector<int> bubbleSort(vector<int> array) {
  // Write your code here.
  int n = array.size();

  for(int i=0; i<n-1 ;i++){
    for(int j=0; j<n-1-i; j++){
      if(array[j]>array[j+1])
        swap(array[j],array[j+1]);
    }
  }
  return array;
}
```

7

10 August 2023 06:14 PM

Insertion Sort

Write a function that takes in an array of integers and returns a sorted version of that array. Use the Insertion Sort algorithm to sort the array.

If you're unfamiliar with Insertion Sort, we recommend watching the Conceptual Overview section of this question's video explanation before starting to code.

Sample Input

```
array = [8, 5, 2, 9, 5, 6, 3]
```

Sample Output

```
[2, 3, 5, 5, 6, 8, 9]
```

Insertion sort *is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.*

Insertion Sort Algorithm

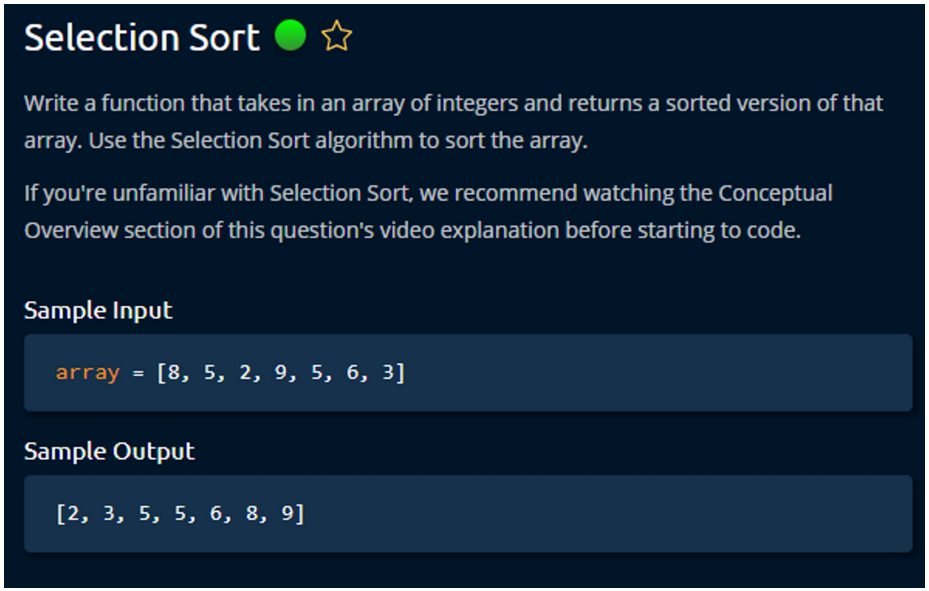
To sort an array of size N in ascending order iterate over the array and compare the current element (key) to its predecessor, if the key element is smaller than its predecessor, compare it to the elements before. Move the greater elements one position up to make space for the swapped element.

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
using namespace std;

vector<int> insertionSort(vector<int> array) {
  // Write your code here.

  int n = array.size();
  for(int i=1; i<n ; i++){
    int j = i-1;
    int key = array[i];
    while(j>=0 && array[j]>key){
      array[j+1] = array[j];
      j--;
    }
    array[j+1] = key;
  }
  return array;
}
```

8

10 August 2023 06:14 PM



Selection Sort

Write a function that takes in an array of integers and returns a sorted version of that array. Use the Selection Sort algorithm to sort the array.

If you're unfamiliar with Selection Sort, we recommend watching the Conceptual Overview section of this question's video explanation before starting to code.

Sample Input

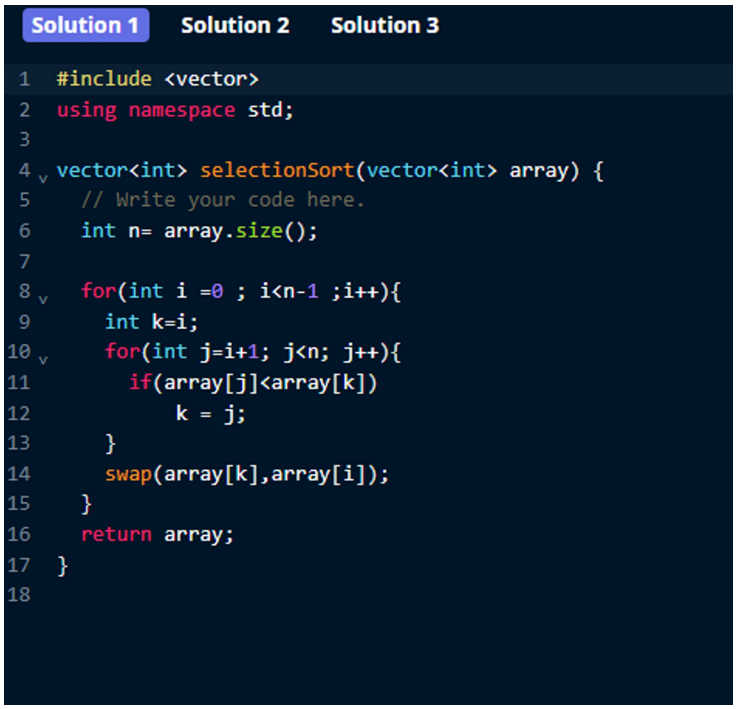
```
array = [8, 5, 2, 9, 5, 6, 3]
```

Sample Output

```
[2, 3, 5, 5, 6, 8, 9]
```

Selection sort *is a simple and efficient sorting algorithm that works by repeatedly selecting the smallest (or largest) element from the unsorted portion of the list and moving it to the sorted portion of the list.*

The algorithm repeatedly selects the smallest (or largest) element from the unsorted portion of the list and swaps it with the first element of the unsorted part. This process is repeated for the remaining unsorted portion until the entire list is sorted.



Solution 1 Solution 2 Solution 3

```cpp
#include <vector>
using namespace std;

vector<int> selectionSort(vector<int> array) {
  // Write your code here.
  int n= array.size();

  for(int i =0 ; i<n-1 ;i++){
    int k=i;
    for(int j=i+1; j<n; j++){
      if(array[j]<array[k])
          k = j;
    }
    swap(array[k],array[i]);
  }
  return array;
}

```

9

10 August 2023 06:14 PM

Binary Search

Write a function that takes in a sorted array of integers as well as a target integer. The function should use the Binary Search algorithm to determine if the target integer is contained in the array and should return its index if it is, otherwise `-1`.

If you're unfamiliar with Binary Search, we recommend watching the Conceptual Overview section of this question's video explanation before starting to code.

Sample Input

```
array = [0, 1, 21, 33, 45, 45, 61, 71, 72, 73]
target = 33
```

Sample Output

```
3
```

Binary Search *is defined as a searching algorithm used in a sorted array by* ***repeatedly dividing the search interval in half****. The idea of binary search is to use the information that the array is sorted and reduce the time complexity to O(log N).*

Conditions for when to apply Binary Search in a Data Structure:

To apply Binary Search algorithm:

- The data structure must be sorted.
- Access to any element of the data structure takes constant time.

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
using namespace std;

//binary search works on sorted array, it divides the array in l, h ,mid
// and check every time if target == array[mid] return mid else check
// if key < array[mid] if yes make h = mid-1 if vice versa then make l = mid+1

int binarySearch(vector<int> array, int target) {
  // Write your code here.
  int start =0 ;
  int last = array.size()-1;

  while(start <=last){
    int mid = (start+last)/2;
    if(array[mid]==target)
        return mid;
    else if(target < array[mid])
        last = mid-1;
    else
        start = mid+1;
  }
  return -1;
}

```

10

10 August 2023 06:36 PM

Difficulty: **Category:** **Successful Submissions:** 97,399+

Nth Fibonacci

The Fibonacci sequence is defined as follows: the first number of the sequence is `0`, the second number is `1`, and the nth number is the sum of the (n - 1)th and (n - 2)th numbers. Write a function that takes in an integer `n` and returns the nth Fibonacci number.

Important note: the Fibonacci sequence is often defined with its first two numbers as `F0 = 0` and `F1 = 1`. For the purpose of this question, the first Fibonacci number is `F0`; therefore, `getNthFib(1)` is equal to `F0`, `getNthFib(2)` is equal to `F1`, etc..

Sample Input #1

```
n = 2
```

Sample Output #1

```
1 // 0, 1
```

Sample Input #2

```
n = 6
```

Sample Output #2

```
5 // 0, 1, 1, 2, 3, 5
```

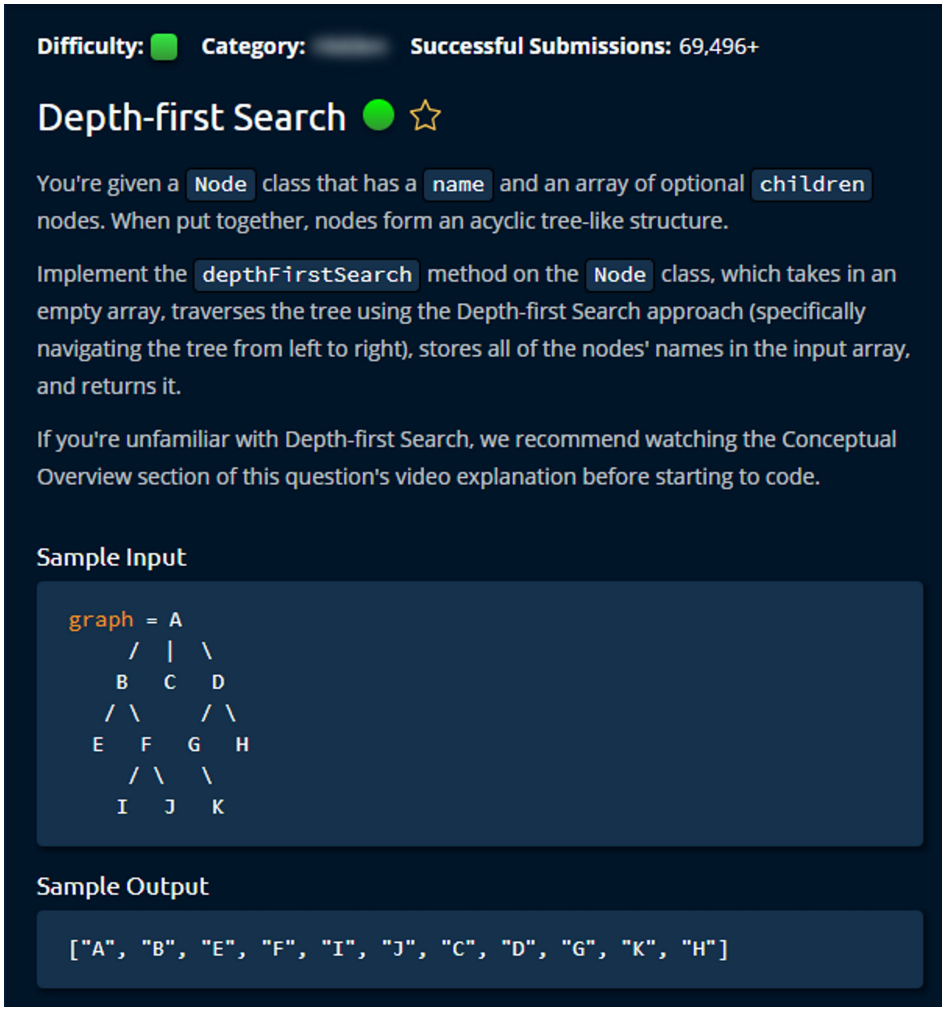
Solution 1 **Solution 2** **Solution 3**

```
using namespace std;

int getNthFib(int n) {
  // Write your code here.
  if(n==1)
      return 0;
  if(n==2)
     return 1;
  if(n>2){
    int j =0, k=1;
    int m;
    for(int i =2; i<n; i++){
      m = j+k;
      j=k;
      k=m;
    }
    return m;
  }


  // return -1;
}

```

11

10 August 2023 06:36 PM

Depth First Traversal (or DFS) for a graph is similar to [Depth First Traversal of a tree.](#) The only catch here is, that, unlike trees, graphs may contain cycles (a node may be visited twice). To avoid processing a node more than once, use a Boolean visited array. A graph can have more than one DFS traversal.



Difficulty: **Category:** **Successful Submissions:** 69,496+

Depth-first Search

You're given a `Node` class that has a `name` and an array of optional `children` nodes. When put together, nodes form an acyclic tree-like structure.

Implement the `depthFirstSearch` method on the `Node` class, which takes in an empty array, traverses the tree using the Depth-first Search approach (specifically navigating the tree from left to right), stores all of the nodes' names in the input array, and returns it.

If you're unfamiliar with Depth-first Search, we recommend watching the Conceptual Overview section of this question's video explanation before starting to code.

Sample Input

```
graph = A
      /  |  \
     B   C   D
    / \     / \
   E   F   G   H
      / \   \
     I   J   K
```

Sample Output

```
["A", "B", "E", "F", "I", "J", "C", "D", "G", "K", "H"]
```

Solution 1 | Solution 2 | Solution 3

```cpp
#include <vector>
using namespace std;

// Do not edit the class below except
// for the depthFirstSearch method.
// Feel free to add new properties
// and methods to the class.
class Node {
public:
  string name;
  vector<Node *> children;

  Node(string str) { name = str; }

  vector<string> depthFirstSearch(vector<string> *array) {
    // Write your code here.

    //passing the data of current to the array
    array->push_back(this->name);

    // for every child that this current node has in children array,
    // run DFS on it
    for(auto child: this->children)
        child->depthFirstSearch(array);
    return *array;
  }

  Node *addChild(string name) {
    Node *child = new Node(name);
    children.push_back(child);
    return this;
  }
};
```

12

10 August 2023 06:36 PM

Difficulty: **Category:** **Successful Submissions:** 86,103+

Find Closest Value In BST

Write a function that takes in a Binary Search Tree (BST) and a target integer value and returns the closest value to that target value contained in the BST.

You can assume that there will only be one closest value.

Each `BST` node has an integer `value`, a `left` child node, and a `right` child node. A node is said to be a valid `BST` node if and only if it satisfies the BST property: its `value` is strictly greater than the values of every node to its left; its `value` is less than or equal to the values of every node to its right; and its children nodes are either valid `BST` nodes themselves or `None` / `null`.

Sample Input

```
tree =   10
       /     \
      5       15
    /   \   /   \
   2     5 13   22
 /           \
1            14
target = 12
```

Sample Output

```
13
```

Solution 1 Solution 2 Solution 3

```cpp
class BST {
public:
  int value;
  BST *left;
  BST *right;

  BST(int val);
  BST &insert(int val);
};

int findClosestValueInBst(BST *tree, int target) {
  // Write your code here.
  int minDiff = abs(target - tree->value);
  int key = tree->value;
  //checking for the root node itself
  if(minDiff==0){
    return tree->value;f
  }

  while(tree!=NULL){
    // as tree is BST so INorder trav gives us sorted nodes.
    // if target is less than node value traverse left side else right side.
    if(target < tree->value){
      if(abs(target - tree->value) < minDiff){
            minDiff = abs(target-tree->value);
            key = tree->value; }
      tree = tree->left;
    }
    else {
      if(abs(target - tree->value) < minDiff){
            minDiff = abs(target-tree->value);
            key = tree->value; }
      tree = tree->right;

    }
  }
  return key;

}
```

13

10 August 2023 06:36 PM

Difficulty: Easy **Category:** [redacted] **Successful Submissions:** 73,442+

Branch Sums

Write a function that takes in a Binary Tree and returns a list of its branch sums ordered from leftmost branch sum to rightmost branch sum.

A branch sum is the sum of all values in a Binary Tree branch. A Binary Tree branch is a path of nodes in a tree that starts at the root node and ends at any leaf node.

Each `BinaryTree` node has an integer `value`, a `left` child node, and a `right` child node. Children nodes can either be `BinaryTree` nodes themselves or `None` / `null`.

Sample Input

```
tree =     1
         /   \
        2     3
      /  \   /  \
     4    5 6    7
   /  \  /
  8   9 10
```

Sample Output

```
[15, 16, 18, 10, 11]
// 15 == 1 + 2 + 4 + 8
// 16 == 1 + 2 + 4 + 9
// 18 == 1 + 2 + 5 + 10
// 10 == 1 + 3 + 6
// 11 == 1 + 3 + 7
```

Solution 1 | Solution 2 | Solution 3

```cpp
using namespace std;

// This is the class of the input root. Do not edit it.
class BinaryTree {
public:
  int value;
  BinaryTree *left;
  BinaryTree *right;

  BinaryTree(int value) {
    this->value = value;
    left = nullptr;
    right = nullptr;
  }
};

void solveBranches(BinaryTree *root, vector<int> &res, int sum){
  if(!root){
    return;
  }
  sum+= root->value;

  // when that node has no left or right child push sum value in array
  if(root->left ==NULL && root->right==NULL){
    res.push_back(sum);
  }
  solveBranches(root->left, res, sum);
  solveBranches(root->right, res, sum);

}

vector<int> branchSums(BinaryTree *root) {
  // Write your code here.
  vector<int> res;
  solveBranches(root, res,0);
  return res;
}

```

14

10 August 2023 06:36 PM

Difficulty: **Category:** **Successful Submissions:** 67,729+

Node Depths

The distance between a node in a Binary Tree and the tree's root is called the node's depth.

Write a function that takes in a Binary Tree and returns the sum of its nodes' depths.

Each `BinaryTree` node has an integer `value`, a `left` child node, and a `right` child node. Children nodes can either be `BinaryTree` nodes themselves or `None` / `null`.

Sample Input

```
tree =    1
        /     \
       2       3
     /   \   /   \
    4     5 6     7
  /   \
 8     9
```

Sample Output

```
16
// The depth of the node with value 2 is 1.
// The depth of the node with value 3 is 1.
// The depth of the node with value 4 is 2.
// The depth of the node with value 5 is 2.
// Etc..
// Summing all of these depths yields 16.
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

class BinaryTree {
public:
  int value;
  BinaryTree *left;
  BinaryTree *right;

  BinaryTree(int value) {
    this->value = value;
    left = nullptr;
    right = nullptr;
  }
};

int FindingDepth(int depth, BinaryTree *root ){
  if(!root)
      return 0;

  // increasing depth by 1 at each pass as depth increases by 0 at each level of tree
  return depth + FindingDepth(depth+1 , root->left) + FindingDepth(depth+1, root->right);
}

int nodeDepths(BinaryTree *root) {
  // Write your code here.

  //passing depth as 0 initially.
  return FindingDepth(0 , root);
  return -1;
}

```

15

10 August 2023 06:45 PM

Difficulty: **Category:** **Successful Submissions:** 45,572+

Minimum Waiting Time

You're given a non-empty array of positive integers representing the amounts of time that specific queries take to execute. Only one query can be executed at a time, but the queries can be executed in any order.

A query's **waiting time** is defined as the amount of time that it must wait before its execution starts. In other words, if a query is executed second, then its waiting time is the duration of the first query; if a query is executed third, then its waiting time is the sum of the durations of the first two queries.

Write a function that returns the minimum amount of total waiting time for all of the queries. For example, if you're given the queries of durations `[1, 4, 5]`, then the total waiting time if the queries were executed in the order of `[5, 1, 4]` would be `(0) + (5) + (5 + 1) = 11`. The first query of duration `5` would be executed immediately, so its waiting time would be `0`, the second query of duration `1` would have to wait `5` seconds (the duration of the first query) to be executed, and the last query would have to wait the duration of the first two queries before being executed.

Note: you're allowed to mutate the input array.

Sample Input

```
queries = [3, 2, 1, 2, 6]
```

Sample Output

```
17
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

int minimumWaitingTime(vector<int> queries) {
  // Write your code here.

  int n = queries.size();
  sort(queries.begin(), queries.end());

  int sum=queries[0];

  vector<int> result;
  result.push_back(sum);

  for(int i=1; i<n-1; i++){
    sum+=queries[i];
    result.push_back(sum);
    cout << result[i] <<endl;
    cout << sum <<endl;
  }


  int sumRes = 0;
  for(int j=0; j<n-1; j++){
    sumRes += result[j];
  }

  return sumRes;
}

```

16

10 August 2023 06:45 PM

Difficulty: **Category:** **Successful Submissions:** 40,963+

Class Photos

It's photo day at the local school, and you're the photographer assigned to take class photos. The class that you'll be photographing has an even number of students, and all these students are wearing red or blue shirts. In fact, exactly half of the class is wearing red shirts, and the other half is wearing blue shirts. You're responsible for arranging the students in two rows before taking the photo. Each row should contain the same number of the students and should adhere to the following guidelines:

- All students wearing red shirts must be in the same row.
- All students wearing blue shirts must be in the same row.
- Each student in the back row must be strictly taller than the student directly in front of them in the front row.

You're given two input arrays: one containing the heights of all the students with red shirts and another one containing the heights of all the students with blue shirts. These arrays will always have the same length, and each height will be a positive integer. Write a function that returns whether or not a class photo that follows the stated guidelines can be taken.

Note: you can assume that each class has at least 2 students.

Sample Input

```
redShirtHeights = [5, 8, 1, 3, 4]
blueShirtHeights = [6, 9, 2, 4, 5]
```

Sample Output

```
true // Place all students with blue shirts in the back row.
```

Solution 1 Solution 2 Solution 3

```cpp
#include <vector>
using namespace std;

bool classPhotos(vector<int> redShirtHeights, vector<int> blueShirtHeights) {
  // Write your code here.

  int n = redShirtHeights.size();

    sort(redShirtHeights.begin(), redShirtHeights.end());
    sort(blueShirtHeights.begin(), blueShirtHeights.end());

  // finding the max element so we can find out which colored student will be at back.

  int maxR = *max_element(redShirtHeights.begin(), redShirtHeights.end());
  int maxB = *max_element(blueShirtHeights.begin(), blueShirtHeights.end());
  cout << maxR<<endl;
  cout << maxB;

  if(maxR==maxB)
      return false;
  else if(maxB> maxR){
    for(int i=0; i<n ;i++){
      if(blueShirtHeights[i] - redShirtHeights[i]<=0)
            return false;
    }
  }
  else{
        for(int i=0; i<n ;i++){
      if(redShirtHeights[i] - blueShirtHeights[i]<=0)
            return false;
    }
  }


  return true;
}

```

17

10 August 2023 06:45 PM

Difficulty: **Category:** **Successful Submissions:** 33,913+

Tandem Bicycle

A tandem bicycle is a bicycle that's operated by two people: person A and person B. Both people pedal the bicycle, but the person that pedals faster dictates the speed of the bicycle. So if person A pedals at a speed of `5`, and person B pedals at a speed of `4`, the tandem bicycle moves at a speed of `5` (i.e., `tandemSpeed = max(speedA, speedB)`).

You're given two lists of positive integers: one that contains the speeds of riders wearing red shirts and one that contains the speeds of riders wearing blue shirts. Each rider is represented by a single positive integer, which is the speed that they pedal a tandem bicycle at. Both lists have the same length, meaning that there are as many red-shirt riders as there are blue-shirt riders. Your goal is to pair every rider wearing a red shirt with a rider wearing a blue shirt to operate a tandem bicycle.

Write a function that returns the maximum possible total speed or the minimum possible total speed of all of the tandem bicycles being ridden based on an input parameter, `fastest`. If `fastest = true`, your function should return the maximum possible total speed; otherwise it should return the minimum total speed.

"Total speed" is defined as the sum of the speeds of all the tandem bicycles being ridden. For example, if there are 4 riders (2 red-shirt riders and 2 blue-shirt riders) who have speeds of `1, 3, 4, 5`, and if they're paired on tandem bicycles as follows: `[1, 4], [5, 3]`, then the total speed of these tandem bicycles is `4 + 5 = 9`.

Sample Input

```
redShirtSpeeds = [5, 5, 3, 9, 2]
blueShirtSpeeds = [3, 6, 7, 2, 1]
fastest = true
```

Sample Output

```
32
```

Solution 1 **Solution 2** **Solution 3**

```cpp
#include <vector>
using namespace std;

int tandemBicycle(vector<int> redShirtSpeeds, vector<int> blueShirtSpeeds,
                  bool fastest) {
  // Write your code here.



  if(fastest == true){
        sort(redShirtSpeeds.begin(),redShirtSpeeds.end());
        sort(blueShirtSpeeds.begin(),blueShirtSpeeds.end(), greater<int>());
        int n = redShirtSpeeds.size();
        int MaxTotalSpeed = 0;
        for(int i=0; i<n; ){
            int maxSpeed = max(redShirtSpeeds[i], blueShirtSpeeds[i]);
            MaxTotalSpeed+=maxSpeed;
            i++;
        }
    return MaxTotalSpeed;
  }
  else{
        sort(redShirtSpeeds.begin(),redShirtSpeeds.end());
        sort(blueShirtSpeeds.begin(),blueShirtSpeeds.end());
        int n = redShirtSpeeds.size();
        int MinTotalSpeed = 0;
        for(int i=0; i<n; ){
            int minSpeed = max(redShirtSpeeds[i], blueShirtSpeeds[i]);
            MinTotalSpeed+=minSpeed;
            i++;
        }
    return MinTotalSpeed;

  }

  return 0;
}

```

18

10 August 2023 06:52 PM

Difficulty: 🟩 **Category:** [redacted] **Successful Submissions:** 49,779+

Remove Duplicates From Linked List 🟢 ☆

You're given the head of a Singly Linked List whose nodes are in sorted order with respect to their values. Write a function that returns a modified version of the Linked List that doesn't contain any nodes with duplicate values. The Linked List should be modified in place (i.e., you shouldn't create a brand new list), and the modified Linked List should still have its nodes sorted with respect to their values.

Each `LinkedList` node has an integer `value` as well as a `next` node pointing to the next node in the list or to `None` / `null` if it's the tail of the list.

Sample Input

```
linkedList = 1 -> 1 -> 3 -> 4 -> 4 -> 4 -> 5 -> 6 -> 6 // the head node
```

Sample Output

```
1 -> 3 -> 4 -> 5 -> 6 // the head node with value 1
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

// This is an input struct. Do not edit.
class LinkedList {
public:
  int value;
  LinkedList *next = nullptr;

  LinkedList(int value) { this->value = value; }
};

LinkedList *removeDuplicatesFromLinkedList(LinkedList *linkedList) {
  // Write your code here.
  LinkedList *p;

  p = linkedList;

  while(p->next!=NULL){
    if(p->value == p->next->value){
      p->next = p->next->next;
      continue;
    }
     p = p->next;
  }

  return linkedList;
}
```

19

10 August 2023 06:52 PM

Difficulty: **Category:** **Successful Submissions:** 56,939+

Product Sum

Write a function that takes in a "special" array and returns its product sum.

A "special" array is a non-empty array that contains either integers or other "special" arrays. The product sum of a "special" array is the sum of its elements, where "special" arrays inside it are summed themselves and then multiplied by their level of depth.

The depth of a "special" array is how far nested it is. For instance, the depth of `[]` is `1`; the depth of the inner array in `[[]]` is `2`; the depth of the innermost array in `[[[]]]` is `3`.

Therefore, the product sum of `[x, y]` is `x + y`; the product sum of `[x, [y, z]]` is `x + 2 * (y + z)`; the product sum of `[x, [y, [z]]]` is `x + 2 * (y + 3z)`.

Sample Input

```
array = [5, 2, [7, -1], 3, [6, [-13, 8], 4]]
```

Sample Output

```
12 // calculated as: 5 + 2 + 2 * (7 - 1) + 3 + 2 * (6 + 3 * (-13 + 8) +
```

Solution 1 **Solution 2** **Solution 3**

```cpp
#include <any>
#include <vector>

using namespace std;

// Tip: You can use el.type() == typeid(vector<any>) to check whether an item is
// a list or an integer.
// If you know an element of the array is vector<any> you can cast it using:
//     any_cast<vector<any>>(element)
// If you know an element of the array is an int you can cast it using:
//     any_cast<int>(element)

int productSumHelper(vector<any> array, int multiplier=1){
  int sum=0;
  for(any x:array){
    if(x.type()==typeid(vector<any>)){
    sum= sum + productSumHelper(any_cast<vector<any>>(x), multiplier+1);
    }
    else{
      sum+=any_cast<int>(x);
    }
  }
return sum*multiplier;
}
int productSum(vector<any> array) {
  // Write your code here.
  return productSumHelper(array);
  return -1;
}

```

20

10 August 2023 06:52 PM

Difficulty: **Category:** **Successful Submissions:** 58,121+

Find Three Largest Numbers

Write a function that takes in an array of at least three integers and, without sorting the input array, returns a sorted array of the three largest integers in the input array.

The function should return duplicate integers if necessary; for example, it should return `[10, 10, 12]` for an input array of `[10, 5, 9, 10, 12]`.

Sample Input

```
array = [141, 1, 17, -7, -17, -27, 18, 541, 8, 7, 7]
```

Sample Output

```
[18, 141, 541]
```

Solution 1 Solution 2 Solution 3

```cpp
#include <vector>
using namespace std;

vector<int> maxNumPushInResult(vector<int> array){
  vector<int> result;
  int n = 3;
  cout << n << endl;

  while(n>0){
    int max = *max_element(array.begin(), array.end());
    int maxIndex = max_element(array.begin(), array.end()) - array.begin();
    vector<int>::iterator it;
   it = array.begin()+ maxIndex;
    result.push_back(max);
    array.erase(it);
    n--;
  }
 sort(result.begin(),result.end());
  return result;
}

vector<int> findThreeLargestNumbers(vector<int> array) {
  // Write your code here.
  return maxNumPushInResult(array);
  return {};
}

```

21

10 August 2023 06:52 PM

Difficulty: **Category:** **Successful Submissions:** 80,267+

Palindrome Check

Write a function that takes in a non-empty string and that returns a boolean representing whether the string is a palindrome.

A palindrome is defined as a string that's written the same forward and backward. Note that single-character strings are palindromes.

Sample Input

```
string = "abcdcba"
```

Sample Output

```
true // it's written the same forward and backward
```

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;
bool isPalindrome(string str) {
  // Write your code here.
  int n = str.size();
  char revStr[n];

  //reversing original array into other array and then will compare to the
  // original one.
  int i;
  for(i=0; str[i]!= '\0'; i++){}
  i=i-1;
  for(int j=0; i>=0 ; i--,j++){
    revStr[j]= str[i];
  }
  cout << str <<endl;
  cout << revStr <<endl;

  // compairing both the strings.
  for(int k=0; k<n ;k++){
    if(str[k]==revStr[k]){
      continue;
    }
    else {
      return false;
    }
  }

  return true;
}

```

22

10 August 2023 07:54 PM

Difficulty: **Category:** **Successful Submissions:** 58,243+

Caesar Cipher Encryptor

Given a non-empty string of lowercase letters and a non-negative integer representing a key, write a function that returns a new string obtained by shifting every letter in the input string by k positions in the alphabet, where k is the key.

Note that letters should "wrap" around the alphabet; in other words, the letter `z` shifted by one returns the letter `a`.

Sample Input

```
string = "xyz"
key = 2
```

Sample Output

```
"zab"
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

string caesarCypherEncryptor(string str, int key) {
  // Write your code here.
  int n = str.size();

  string newStr= "";


  for(int i=0; i<n; i++){
    char ch = ((((unsigned int)str[i] - (unsigned int)'a')+key)%26 + (unsigned int)'a');
    newStr.push_back(ch);
  }

  return newStr;
  return "";}
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;


string caesarCypherEncryptor(string str, int key) {
  // Write your code here.
  int n = str.size();

  vector<char> newStr;


  for(int i=0; i<n; i++){
    char ch = ((((unsigned int)str[i] - (unsigned int)'a')+key)%26 + (unsigned int)'a');
    newStr.push_back(ch);
  }

  string s(newStr.begin(),newStr.end());
  return s;
  return "";
}

```

23

10 August 2023 07:54 PM

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;
#include <sstream>


string runLengthEncoding(string str) {
  // Write your code here.
  ostringstream stream;

  int count =1;
  int n = str.size();

  for(int i=1; i<n; i++){
    if(str[i]==str[i-1] || count<=9){
      count +=1;
    }
    else{
      stream << count << str[i-1];
      count=1;
    }
  }
  cout<< count <<endl;
  stream << count << str.back();
  return stream.str();

}

```

Difficulty: **Category:** **Successful Submissions:** 44,454+

Run-Length Encoding

Write a function that takes in a non-empty string and returns its run-length encoding.

From Wikipedia, "run-length encoding is a form of lossless data compression in which runs of data are stored as a single data value and count, rather than as the original run." For this problem, a run of data is any sequence of consecutive, identical characters. So the run `"AAA"` would be run-length-encoded as `"3A"`.

To make things more complicated, however, the input string can contain all sorts of special characters, including numbers. And since encoded data must be decodable, this means that we can't naively run-length-encode long runs. For example, the run `"AAAAAAAAAAAA"` (12 `A`s), can't naively be encoded as `"12A"`, since this string can be decoded as either `"AAAAAAAAAAAA"` or `"1AA"`. Thus, long runs (runs of 10 or more characters) should be encoded in a split fashion; the aforementioned run should be encoded as `"9A3A"`.

Sample Input

```
string = "AAAAAAAAAAAAABBCCCCDD"
```

Sample Output

```
"9A4A2B4C2D"
```

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;
// #define ASCII_0 48
string runLengthEncoding(string str) {
  // Write your code here.
  int n = str.size();
  string newStrr = "";


  int count = 1 ;
  for(int i=0; i<n; i++){
      if(count>=9 || str[i]!=str[i+1] || i==n){
          newStrr.append(to_string(count));

          newStrr.push_back(str[i]);
          count =1;
      }
    else{
      count++;
    }

  }
  return newStrr;
}

```

24

10 August 2023 07:54 PM

Difficulty: **Category:** [redacted] **Successful Submissions:** 39,310+

Generate Document

You're given a string of available characters and a string representing a document that you need to generate. Write a function that determines if you can generate the document using the available characters. If you can generate the document, your function should return `true`; otherwise, it should return `false`.

You're only able to generate the document if the frequency of unique characters in the characters string is greater than or equal to the frequency of unique characters in the document string. For example, if you're given `characters = "abcabc"` and `document = "aabbccc"` you **cannot** generate the document because you're missing one `c`.

The document that you need to create may contain any characters, including special characters, capital letters, numbers, and spaces.

Note: you can always generate the empty string (`""`).

Sample Input

```
characters = "Bste!hetsi ogEAxpelrt x "
document = "AlgoExpert is the Best!"
```

Sample Output

```
true
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

bool generateDocument(string characters, string document) {
  // Write your code here.
  sort(characters.begin(), characters.end());

  sort(document.begin(), document.end());

  cout << characters <<endl;
  cout << document;

  int characterLength = characters.size();
  int documentLength = document.size();

  for(int i =0 ; i<documentLength; i++){
      if(document[i]!=characters[i])
        return false;
      else{
        continue;
      }
    }
  return true;
}
```

Solution 1 | Solution 2 | Solution 3

```cpp
using namespace std;

bool generateDocument(string characters, string document) {
  // Write your code here.
  vector<int> charHash(256,0);
  vector<int> docHash(256,0);
  int characterLength = characters.size();
  int documentLength = document.size();

  //incrementing in char hash table;
  for(int i=0; i<characterLength; i++){
    charHash[characters[i]]+=1;
  }

//incrementing in doc hash table;
  for(int i=0; i<documentLength; i++){
    docHash[document[i]]+=1;
  }


  //compairing the frequency in char and doc hash. if char fre in char is
  //equal to or greater than
  // we are compairing fre of each character from both hash vectors.
  for(int i=0; i<256; i++){
    if(docHash[i]> charHash[i])
      return false;
    else continue;
  }


  return true;
}
```

25

10 August 2023 07:54 PM

Difficulty: **Category:** **Successful Submissions:** 41,997+

First Non-Repeating Character

Write a function that takes in a string of lowercase English-alphabet letters and returns the index of the string's first non-repeating character.

The first non-repeating character is the first character in a string that occurs only once.

If the input string doesn't have any non-repeating characters, your function should return `-1`.

Sample Input

```
string = "abcdcaf"
```

Sample Output

```
1 // The first non-repeating character is "b" and is found at index 1.
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

int firstNonRepeatingCharacter(string string) {
  // Write your code here.
  //size of string array
  int n = string.size();
  //hash vector
  vector<int> charHash(25,0);

  //traversing the array and incrementing the index based
  //on ASCII char as only 0-25 char are there we are
  // subtracting 97 to get index starting from 0.
  for(int i=0; i<n; i++){
    charHash[string[i]-97]+=1;
  }

  char ch;

  //traverse the hashTable with string[i]-97 as index of hash
  // vector as we ar checking froM the string starting for each
  // character that if this character value is 1 . if yes return that index
  // else move to the next char in string.
  for(int i=0; i<26; i++){
    if(charHash[string[i] - 97]==1)
      return i;
  }


  return -1;
}
```

26

10 August 2023 07:54 PM

Difficulty: **Category:** **Successful Submissions:** 6,984+

Semordnilap

Write a function that takes in a list of unique strings and returns a list of semordnilap pairs.

A semordnilap pair is defined as a set of different strings where the reverse of one word is the same as the forward version of the other. For example the words "diaper" and "repaid" are a semordnilap pair, as are the words "palindromes" and "semordnilap".

The order of the returned pairs and the order of the strings within each pair does not matter.

Sample Input

```
words = ["diaper", "abc", "test", "cba", "repaid"]
```

Sample Output

```
[["diaper", "repaid"], ["abc", "cba"]]
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;


vector<vector<string>> semordnilap(vector<string> words) {
  // Write your code here.
  int n = words.size();
  vector<string> str(n);
  vector<string> :: iterator it;

  //traversing the words array and sorting each element based on
  //ASCII in that and pushing in new array
  for(it = words.begin(); it!=words.end(); it++){
    string a = *it;
    sort(a.begin(), a.end());
    str.push_back(a);
  }

  for(it = str.begin(); it!=str.end(); it++){
      cout << *it <<endl;
  }

  //creating a 2D array.
  vector<vector<string>> arr(n);

  cout << "hey" << str.at(4) <<endl;
  return {};
}

```

27

10 August 2023 08:03 PM

```
//APPROACH :
  // we initialize an unorderd set with the characters of first word of string.
  // then we start traversing the words and for each word we are traversing
  // char of the word. we start checking in the main set if that char is
  // there or not if there then move it to the temp set and when after one round
  // of traversing char of one word, swap temp set with main and after the
  // last word main set will be left with only common characters.
  // as each traversal if a char is not common in present word, as temp is updated
  // at every new word so that char is removed from common if it was preesent there.
```

Solution 1 Solution 2 Solution 3

```
using namespace std;

vector<string> commonCharacters(vector<string> strings) {
  // Write your code here.
  //create a unordered set
  unordered_set<char> mainSet(strings[0].begin(), strings[0].end());
  //loop through element in array
  for(auto word : strings){
    //create a temp unordered set.
      unordered_set<char> temp;
      // loop through each character of word
      for(auto alp : word ){

         if(mainSet.count(alp)==1)
              temp.insert(alp);
      }
      // outside of inner for loop
      //swap the data
    mainSet.swap(temp);
  }

  // now the mainSet has all the common alp.
  //initializing vector that will hold the common char.
  vector<string> result;

//loop through the set and push it in result vector.
  for(auto c:mainSet)
      result.emplace_back(1, c);

  return result;
}
```

Difficulty: **Category:** **Successful Submissions:** 4,013+

Common Characters

Write a function that takes in a non-empty list of non-empty strings and returns a list of characters that are common to all strings in the list, ignoring multiplicity.

Note that the strings are not guaranteed to only contain alphanumeric characters. The list you return can be in any order.

Sample Input

```
strings = ["abc", "bcd", "cbaccd"]
```

Sample Output

```
["b", "c"] // The characters could be ordered differently.
```

28

10 August 2023 08:03 PM

Difficulty: **Category:** **Successful Submissions:** 896+

Optimal Freelancing

You recently started freelance software development and have been offered a variety of job opportunities. Each job has a deadline, meaning there is no value in completing the work after the deadline. Additionally, each job has an associated payment representing the profit for completing that job. Given this information, write a function that returns the maximum profit that can be obtained in a 7-day period.

Each job will take 1 full day to complete, and the deadline will be given as the number of days left to complete the job. For example, if a job has a deadline of 1, then it can only be completed if it is the first job worked on. If a job has a deadline of 2, then it could be started on the first or second day.

Note: There is no requirement to complete all of the jobs. Only one job can be worked on at a time, meaning that in some scenarios it will be impossible to complete them all.

Sample Input

```
jobs = [
  {"deadline": 1, "payment": 1},
  {"deadline": 2, "payment": 1},
  {"deadline": 2, "payment": 2}
]
```

Sample Output

```
3 // Job 0 would be completed first, followed by job 2. Job 1 is not co
```

Solution 1 | Solution 2 | Solution 3

```cpp
#include<vector>
#include<algorithm>

using namespace std;

int optimalFreelancing(vector<unordered_map<string, int>> jobs) {
  // Write your code here.
  sort(jobs.begin(), jobs.end(),[](const auto &jobOne, const auto&jobTwo){
    return jobTwo.at("payment") < jobOne.at("payment");
  });

  const int lengthTime = 7;
  int profit=0;

  vector<bool> timeline(lengthTime, false);

  for(const auto&job:jobs){
    int maxTime= min(job.at("deadline"), lengthTime);
    for(int time=maxTime-1 ; time>=0; time--){
      if(!timeline[time]){
        timeline[time] = true;
        profit+=job.at("payment");
        break;
      }
    }
  }
  return profit;
}
```

29

10 August 2023 08:03 PM

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;

// This is an input struct. Do not edit.
class LinkedList {
public:
  int value;
  LinkedList *next = nullptr;

  LinkedList(int value) { this->value = value; }
};

LinkedList *middleNode(LinkedList *linkedList) {
  // Write your code here.
  // size of linkedList
  int size = 0;
  for(LinkedList *p=linkedList; p!=NULL; ){
    size++;
    p =p->next;
  }

  //if length is odd we can find the middle node easily
  if(size%2!=0){
      LinkedList *q=linkedList;
    int odd = ((size-1)/2)+1;
      for(int p=0; p<((size-1)/2); p++){
          q = q->next;
      }
    return q;
  }else{
    LinkedList *q=linkedList;
    for(int p=0; p<(size/2); p++)
        q = q->next;
     return q;
  }

  return nullptr;
}
```

Difficulty: **Category:** **Successful Submissions:** 4,410+

Middle Node

You're given a Linked List with at least one node. Write a function that returns the middle node of the Linked List. If there are two middle nodes (i.e. an even length list), your function should return the second of these nodes.

Each `LinkedList` node has an integer `value` as well as a `next` node pointing to the next node in the list or to `None` / `null` if it's the tail of the list.

Sample Input

```
linkedList = 2 -> 7 -> 3 -> 5
```

Sample Output

```
        // The middle could be 7 or 3,
3 -> 5 // we return the second middle node
```

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;

// This is an input struct. Do not edit.
class LinkedList {
public:
  int value;
  LinkedList *next = nullptr;

  LinkedList(int value) { this->value = value; }
};

LinkedList *middleNode(LinkedList *linkedList) {
  // Write your code here.

  // we can also use fast and slow. we will run fast by fast=head->next->next.
  // for every moment like this slow will go slow= head->next.

  //approach is after one moment.
  // middleNode points to second node whereas forward points to 3rd node.

  LinkedList *forward = linkedList;
  LinkedList *middleNode = linkedList;

  while(forward!= nullptr && forward->next!=nullptr){
    forward = forward->next->next;
    middleNode = middleNode->next;
  }
  return middleNode;
}
```

30

10 August 2023 08:09 PM

Difficulty: **Category:** **Successful Submissions:** 3,941+

Evaluate Expression Tree

You're given a binary expression tree. Write a function to evaluate this tree mathematically and return a single resulting integer.

All leaf nodes in the tree represent operands, which will always be positive integers. All of the other nodes represent operators. There are 4 operators supported, each of which is represented by a negative integer:

- `-1`: Addition operator, adding the left and right subtrees.
- `-2`: Subtraction operator, subtracting the right subtree from the left subtree.
- `-3`: Division operator, dividing the left subtree by the right subtree. If the result is a decimal, it should be rounded towards zero.
- `-4`: Multiplication operator, multiplying the left and right subtrees.

You can assume the tree will always be a valid expression tree. Each operator also works as a grouping symbol, meaning the bottom of the tree is always evaluated first, regardless of the operator.

Sample Input

```
tree =    -1
        /    \
      -2      -3
     /  \    /  \
   -4    2  8    3
  /  \
 2    3
```

Sample Output

```
6 // (((2 * 3) - 2) + (8 / 3))
```

Solution 1 Solution 2 Solution 3

```cpp
using namespace std;

// This is an input class. Do not edit.
class BinaryTree {
public:
  int value;
  BinaryTree *left = nullptr;
  BinaryTree *right = nullptr;

  BinaryTree(int value) { this->value = value; }
};

int evaluateExpressionTree(BinaryTree *tree) {
  // Write your code here.
  if(tree->value >=0)
      return tree->value;
  int left = evaluateExpressionTree(tree->left);
  int right = evaluateExpressionTree(tree->right);

  if(tree->value==-1)
      return left+right;
  else if(tree->value==-2)
      return left-right;
  else if(tree->value==-3)
      return left/right;
  else if(tree->value==-4)
      return left*right;


  // return tree->value;
}
```

31

10 August 2023 08:09 PM

Difficulty: **Category:** **Successful Submissions:** 4,518+

Transpose Matrix

You're given a 2D array of integers `matrix`. Write a function that returns the transpose of the matrix.

The transpose of a matrix is a flipped version of the original matrix across its main diagonal (which runs from top-left to bottom-right); it switches the row and column indices of the original matrix.

You can assume the input matrix always has at least 1 value; however its width and height are not necessarily the same.

Sample Input #1

```
matrix = [
  [1, 2],
]
```

Sample Output # 1

```
[
  [1],
  [2]
]
```

Solution 1 **Solution 2** **Solution 3**

```cpp
using namespace std;

vector<vector<int>> transposeMatrix(vector<vector<int>> matrix) {
  // Write your code here.
  int rows = matrix.size();
  int columns = matrix[0].size();
  vector<vector<int>> transposeM(columns, vector<int>(rows));

  for(int i=0; i<rows; i++){
    for(int j=0; j<columns; j++){
        transposeM[j][i] = matrix[i][j];
    }
  }

  return transposeM;
}
```