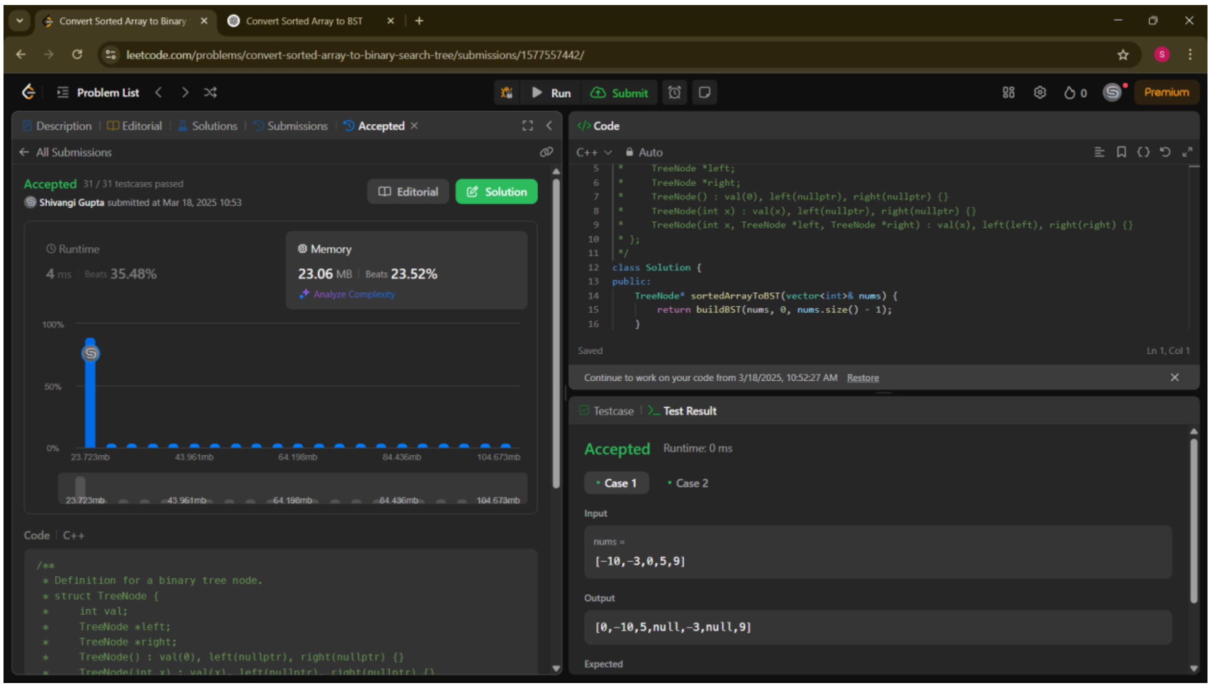
AP Assignment-6

Name: Shivangi Gupta   UID: 22BCS15008   Section: 601-A

108. https://leetcode.com/problems/convert-sorted-array-to-binary-search-tree/description/

Code:

```
class Solution {
public:
   TreeNode* sortedArrayToBST(vector<int>& nums) {
      return buildBST(nums, 0, nums.size() - 1);
   }

   TreeNode* buildBST(const vector<int>& nums, int left, int right) {
      if (left > right) {
         return nullptr;
      }

      int mid = left + (right - left) / 2;  // To prevent overflow
      TreeNode* root = new TreeNode(nums[mid]);

      root->left = buildBST(nums, left, mid - 1);
      root->right = buildBST(nums, mid + 1, right);

      return root;
   }
};
```


Accepted 31 / 31 testcases passed

Shivangi Gupta submitted at Mar 18, 2025 10:53

Runtime 4 ms | Beats 35.48%

Memory 23.06 MB | Beats 23.52%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
```

```
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 *     TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
 * };
 */
class Solution {
public:
    TreeNode* sortedArrayToBST(vector<int>& nums) {
        return buildBST(nums, 0, nums.size() - 1);
    }
```

Continue to work on your code from 3/18/2025, 10:52:27 AM Restore

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums =

```
[-10,-3,0,5,9]
```

Output

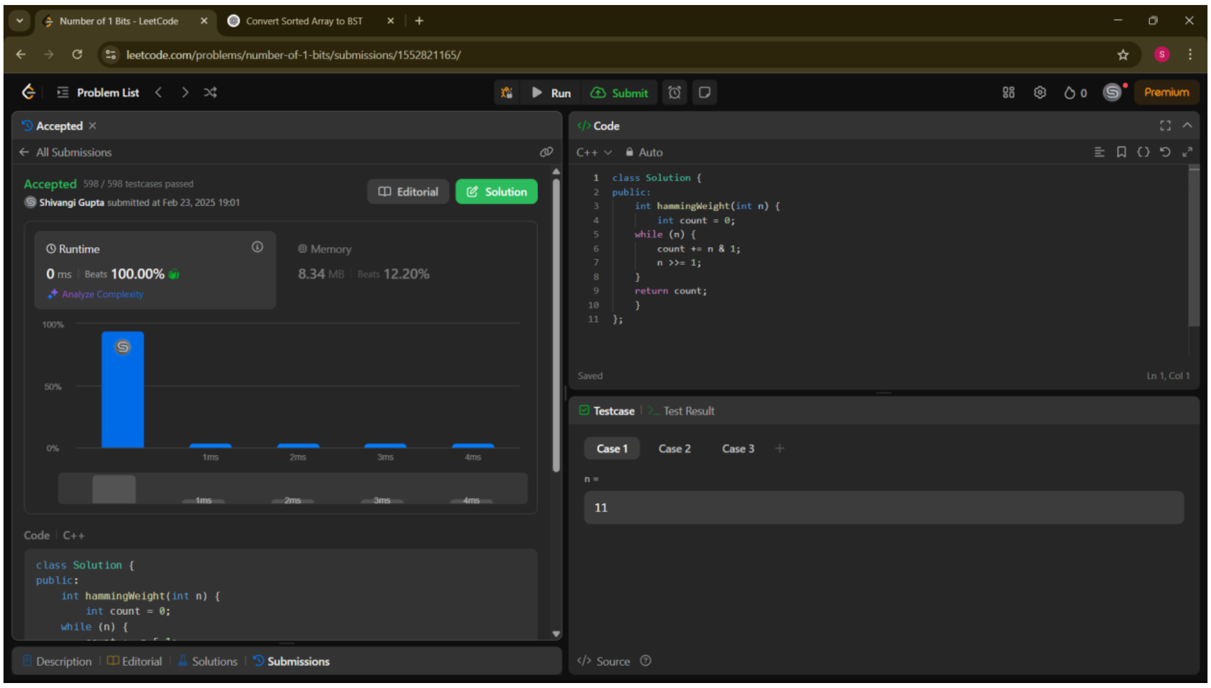
```
[0,-10,5,null,-3,null,9]
```

Expected

191. https://leetcode.com/problems/number-of-1-bits/description/

Code:

```
class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
    while (n) {
        count += n & 1;
        n >>= 1;
    }
    return count;
    }
};
```



leetcode.com/problems/number-of-1-bits/submissions/1552821165/

Accepted 598 / 598 testcases passed

Shivangi Gupta submitted at Feb 23, 2025 19:01

Runtime 0 ms | Beats 100.00%

Memory 8.34 MB | Beats 12.20%

```
class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
    while (n) {
        count += n & 1;
        n >>= 1;
    }
    return count;
    }
};
```

Testcase

n =

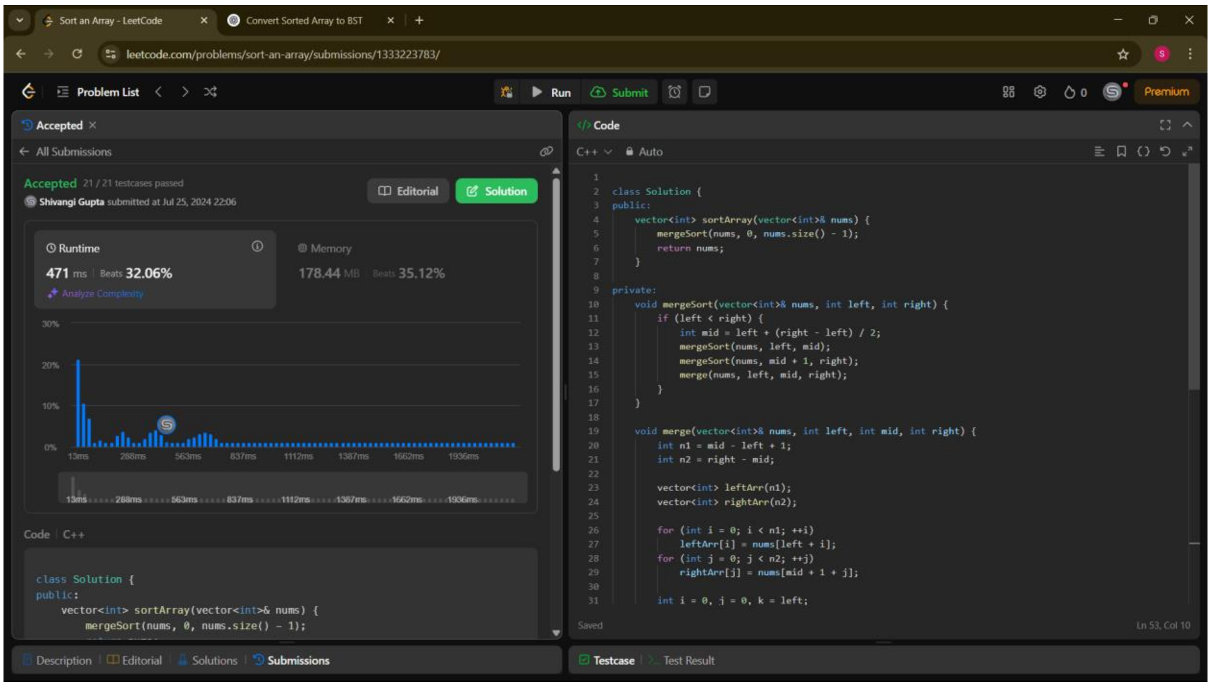
11

912. https://leetcode.com/problems/sort-an-array/description/

Code:

```
class Solution {
public:
    vector<int> sortArray(vector<int>& nums) {
        mergeSort(nums, 0, nums.size() - 1);
        return nums;
    }

private:
    void mergeSort(vector<int>& nums, int left, int right) {
        if (left < right) {
            int mid = left + (right - left) / 2;
            mergeSort(nums, left, mid);
            mergeSort(nums, mid + 1, right);
            merge(nums, left, mid, right);
        }
    }
```

```
void merge(vector<int>& nums, int left, int mid, int right) {
    int n1 = mid - left + 1;
    int n2 = right - mid;

    vector<int> leftArr(n1);
    vector<int> rightArr(n2);

    for (int i = 0; i < n1; ++i)
        leftArr[i] = nums[left + i];
    for (int j = 0; j < n2; ++j)
        rightArr[j] = nums[mid + 1 + j];

    int i = 0, j = 0, k = left;
    while (i < n1 && j < n2) {
        if (leftArr[i] <= rightArr[j]) {
            nums[k] = leftArr[i];
            ++i;
        } else {
            nums[k] = rightArr[j];
            ++j;
        }
        ++k;
    }

    while (i < n1) {
        nums[k] = leftArr[i];
        ++i;
        ++k;
    }

    while (j < n2) {
            nums[k] = rightArr[j];
            ++j;
            ++k;
        }
    }
};
```



Sort an Array - LeetCode | Convert Sorted Array to BST

leetcode.com/problems/sort-an-array/submissions/1333223783/

Accepted 21 / 21 testcases passed

Shivangi Gupta submitted at Jul 25, 2024 22:06

Runtime 471 ms | Beats 32.06%

Memory 178.44 MB | Beats 35.12%

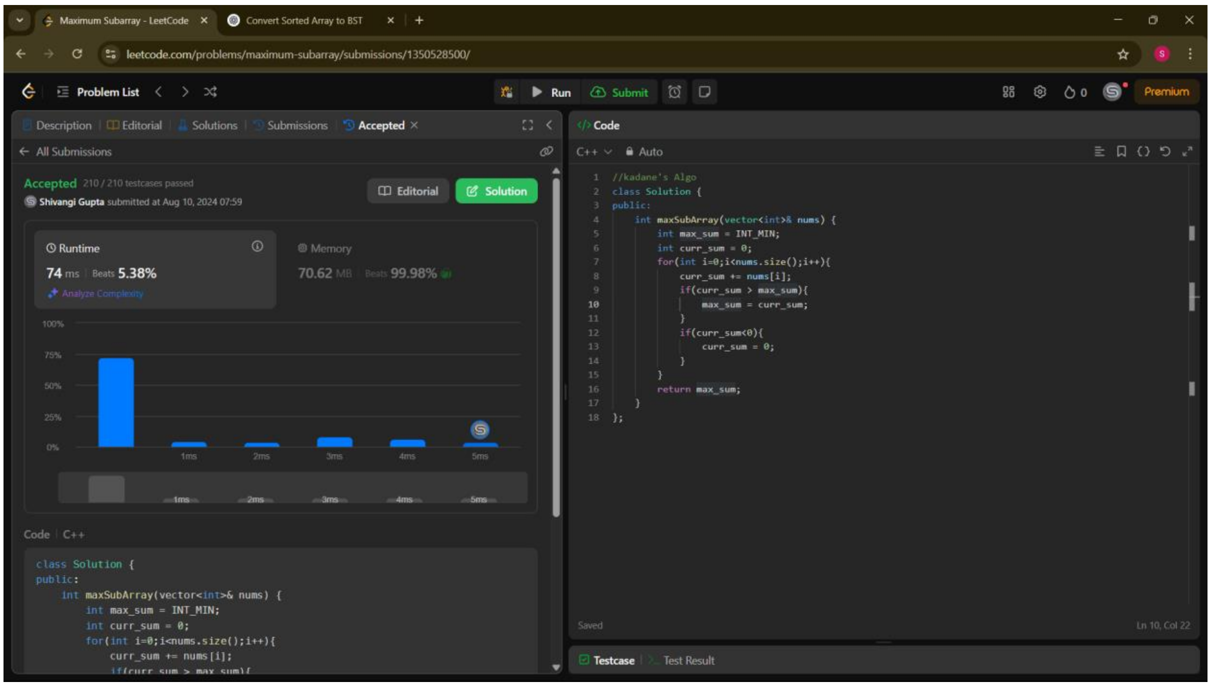
Code | C++

```
class Solution {
public:
    vector<int> sortArray(vector<int>& nums) {
        mergeSort(nums, 0, nums.size() - 1);
        return nums;
    }

private:
    void mergeSort(vector<int>& nums, int left, int right) {
        if (left < right) {
            int mid = left + (right - left) / 2;
            mergeSort(nums, left, mid);
            mergeSort(nums, mid + 1, right);
            merge(nums, left, mid, right);
        }
    }

    void merge(vector<int>& nums, int left, int mid, int right) {
        int n1 = mid - left + 1;
        int n2 = right - mid;

        vector<int> leftArr(n1);
        vector<int> rightArr(n2);

        for (int i = 0; i < n1; ++i)
            leftArr[i] = nums[left + i];
        for (int j = 0; j < n2; ++j)
            rightArr[j] = nums[mid + 1 + j];

        int i = 0, j = 0, k = left;
```

53. https://leetcode.com/problems/maximum-subarray/

Code:

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int max_sum = INT_MIN;
        int curr_sum = 0;
        for(int i=0;i<nums.size();i++){
            curr_sum += nums[i];
            if(curr_sum > max_sum){
                max_sum = curr_sum;
            }
            if(curr_sum<0){
                curr_sum = 0;
            }
        }
        return max_sum;
    }
};
```



Maximum Subarray - LeetCode | Convert Sorted Array to BST

leetcode.com/problems/maximum-subarray/submissions/1350528500/

Accepted 210 / 210 testcases passed

Shivangi Gupta submitted at Aug 10, 2024 07:59

Runtime 74 ms | Beats 5.38%

Memory 70.62 MB | Beats 99.98%

Code | C++

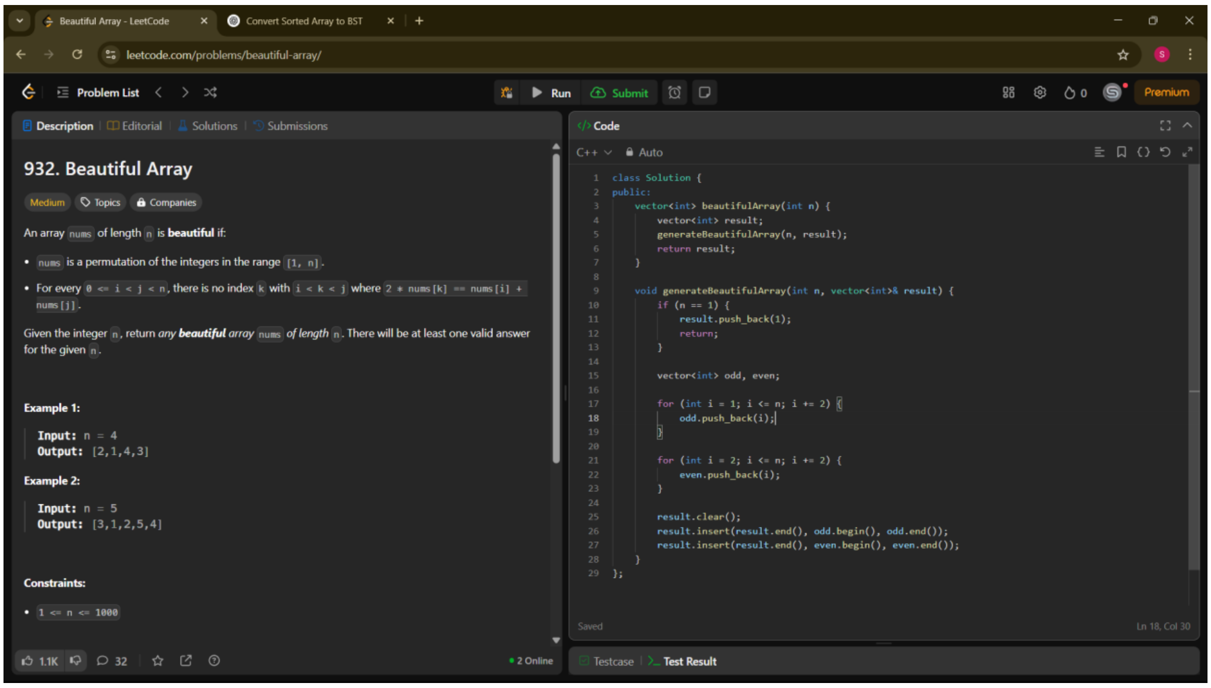
```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int max_sum = INT_MIN;
        int curr_sum = 0;
        for(int i=0;i<nums.size();i++){
            curr_sum += nums[i];
            if(curr_sum > max_sum){
```

```
//kadane's Algo
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int max_sum = INT_MIN;
        int curr_sum = 0;
        for(int i=0;i<nums.size();i++){
            curr_sum += nums[i];
            if(curr_sum > max_sum){
                max_sum = curr_sum;
            }
            if(curr_sum<0){
                curr_sum = 0;
            }
        }
        return max_sum;
    }
};
```

932. https://leetcode.com/problems/beautiful-array/description/

Code:

```
class Solution {
public:
    vector<int> beautifulArray(int n) {
        vector<int> result;
        generateBeautifulArray(n, result);
        return result;
    }

    void generateBeautifulArray(int n, vector<int>& result) {
        if (n == 1) {
            result.push_back(1);
            return;
        }

        vector<int> odd, even;

        for (int i = 1; i <= n; i += 2) {
            odd.push_back(i);
        }

        for (int i = 2; i <= n; i += 2) {
            even.push_back(i);
        }

        result.clear();
        result.insert(result.end(), odd.begin(), odd.end());
        result.insert(result.end(), even.begin(), even.end());
    }
};
```



932. Beautiful Array

An array `nums` of length `n` is **beautiful** if:

- `nums` is a permutation of the integers in the range `[1, n]`.
- For every `0 <= i < j < n`, there is no index `k` with `i < k < j` where `2 * nums[k] == nums[i] + nums[j]`.

Given the integer `n`, return *any* ***beautiful*** *array* `nums` *of length* `n`. There will be at least one valid answer for the given `n`.

Example 1:

```
Input: n = 4
Output: [2,1,4,3]
```

Example 2:

```
Input: n = 5
Output: [3,1,2,5,4]
```

Constraints:

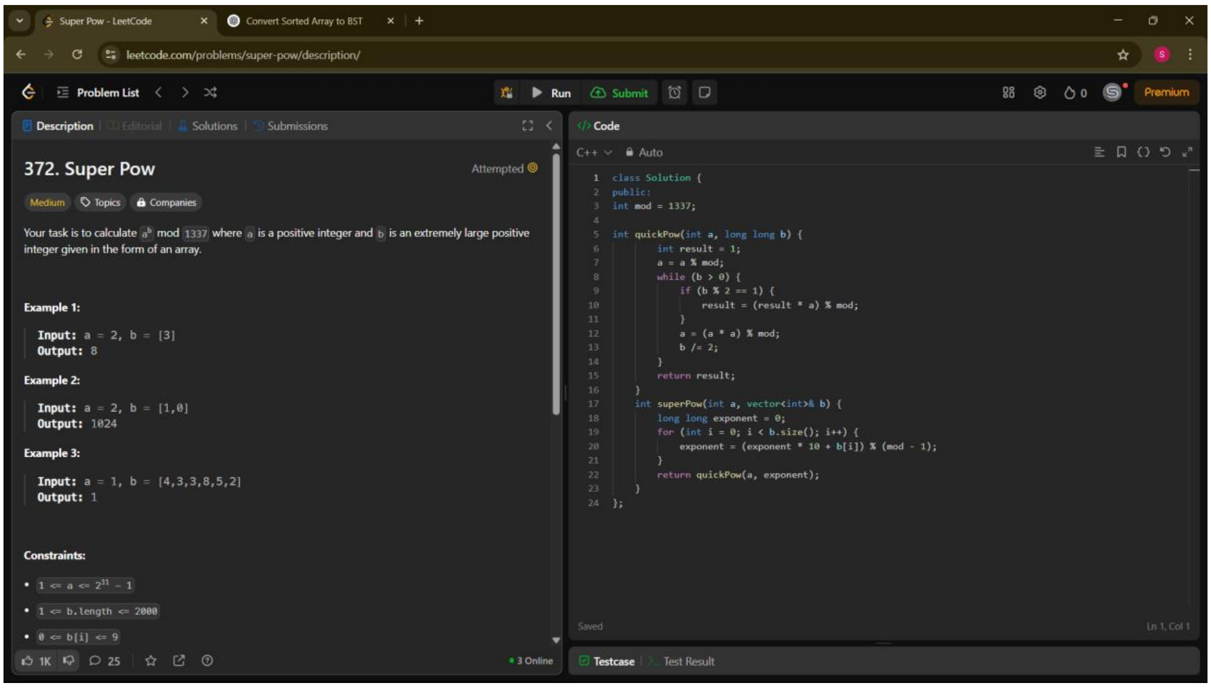
- `1 <= n <= 1000`

```
class Solution {
public:
    vector<int> beautifulArray(int n) {
        vector<int> result;
        generateBeautifulArray(n, result);
        return result;
    }

    void generateBeautifulArray(int n, vector<int>& result) {
        if (n == 1) {
            result.push_back(1);
            return;
        }

        vector<int> odd, even;

        for (int i = 1; i <= n; i += 2) {
            odd.push_back(i);
        }

        for (int i = 2; i <= n; i += 2) {
            even.push_back(i);
        }

        result.clear();
        result.insert(result.end(), odd.begin(), odd.end());
        result.insert(result.end(), even.begin(), even.end());
    }
};
```

372. https://leetcode.com/problems/super-pow/description/

Code:

```
class Solution {
public:
int mod = 1337;

int quickPow(int a, long long b) {
        int result = 1;
        a = a % mod;
        while (b > 0) {
            if (b % 2 == 1) {
                result = (result * a) % mod;
            }
            a = (a * a) % mod;
            b /= 2;
        }
        return result;
    }
    int superPow(int a, vector<int>& b) {
        long long exponent = 0;
        for (int i = 0; i < b.size(); i++) {
            exponent = (exponent * 10 + b[i]) % (mod - 1);
        }
        return quickPow(a, exponent);
    }
};
```



372. Super Pow

Attempted

Medium | Topics | Companies

Your task is to calculate a^b mod `1337` where `a` is a positive integer and `b` is an extremely large positive integer given in the form of an array.

Example 1:

```
Input: a = 2, b = [3]
Output: 8
```

Example 2:

```
Input: a = 2, b = [1,0]
Output: 1024
```

Example 3:

```
Input: a = 1, b = [4,3,3,8,5,2]
Output: 1
```

Constraints:

- $1 <= a <= 2^{31} - 1$
- `1 <= b.length <= 2000`
- `0 <= b[i] <= 9`

```cpp
class Solution {
public:
int mod = 1337;

int quickPow(int a, long long b) {
        int result = 1;
        a = a % mod;
        while (b > 0) {
            if (b % 2 == 1) {
                result = (result * a) % mod;
            }
            a = (a * a) % mod;
            b /= 2;
        }
        return result;
    }
    int superPow(int a, vector<int>& b) {
        long long exponent = 0;
        for (int i = 0; i < b.size(); i++) {
            exponent = (exponent * 10 + b[i]) % (mod - 1);
        }
        return quickPow(a, exponent);
    }
};
```