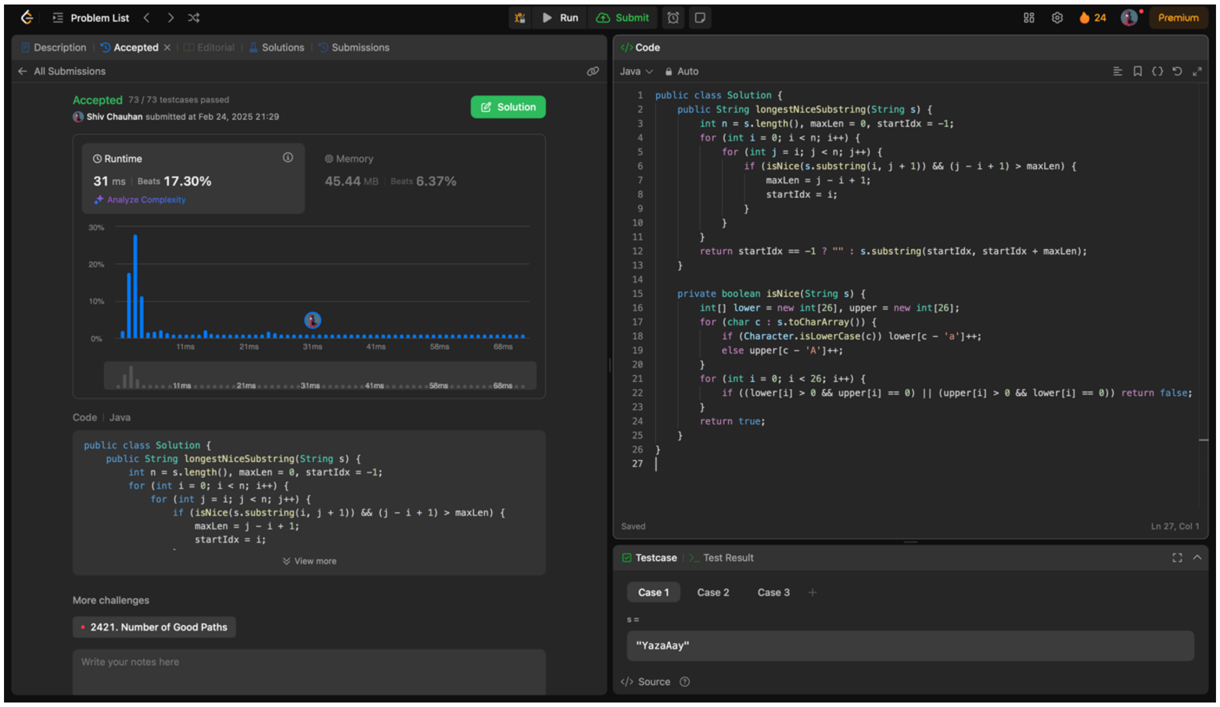
Longest Nice Substring

```java
public class Solution {
    public String longestNiceSubstring(String s) {
        int n = s.length(), maxLen = 0, startIdx = -1;
        for (int i = 0; i < n; i++) {
            for (int j = i; j < n; j++) {
                if (isNice(s.substring(i, j + 1)) && (j - i + 1) > maxLen) {
                    maxLen = j - i + 1;
                    startIdx = i;
                }
            }
        }
        return startIdx == -1 ? "" : s.substring(startIdx, startIdx +
maxLen);
    }

    private boolean isNice(String s) {
        int[] lower = new int[26], upper = new int[26];
        for (char c : s.toCharArray()) {
            if (Character.isLowerCase(c)) lower[c - 'a']++;
            else upper[c - 'A']++;
        }
        for (int i = 0; i < 26; i++) {
            if ((lower[i] > 0 && upper[i] == 0) || (upper[i] > 0 && lower[i]
== 0)) return false;
        }
        return true;
    }
}
```



Accepted 73 / 73 testcases passed
Shiv Chauhan submitted at Feb 24, 2025 21:29

Runtime 31 ms | Beats 17.30%
Memory 45.44 MB | Beats 6.37%

Code | Java

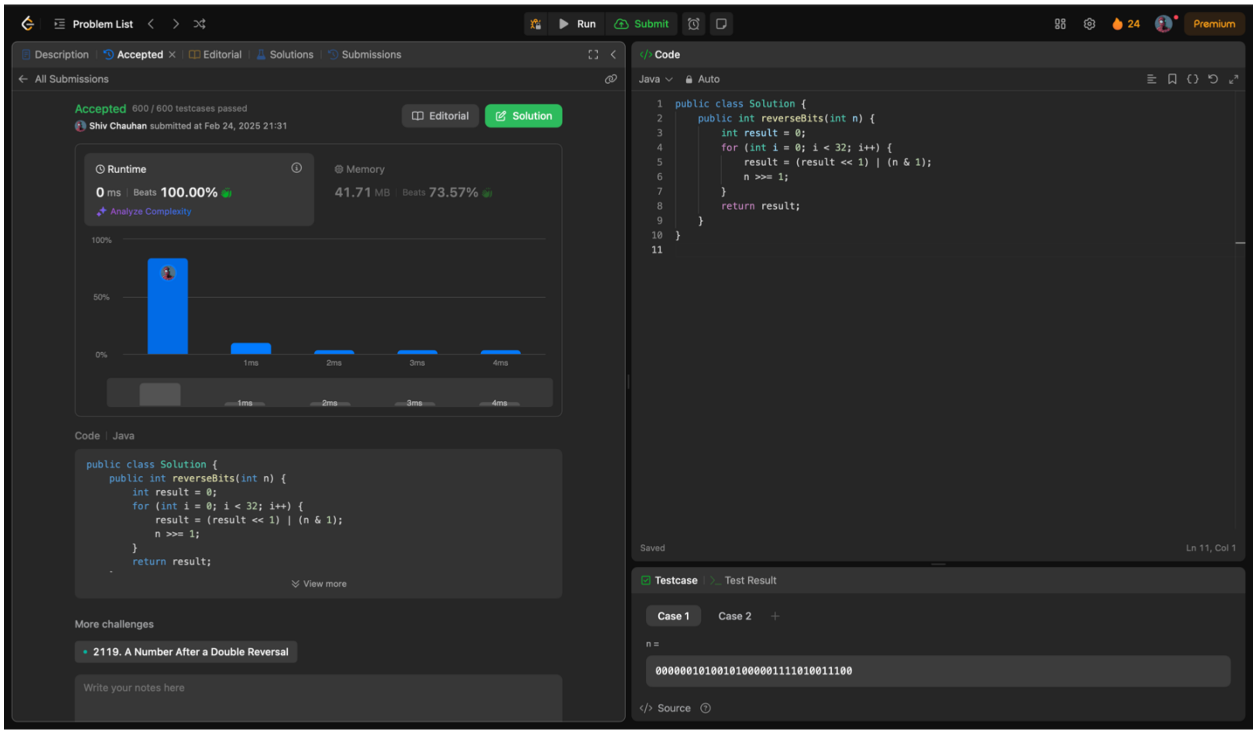
```java
public class Solution {
    public String longestNiceSubstring(String s) {
        int n = s.length(), maxLen = 0, startIdx = -1;
        for (int i = 0; i < n; i++) {
            for (int j = i; j < n; j++) {
                if (isNice(s.substring(i, j + 1)) && (j - i + 1) > maxLen) {
                    maxLen = j - i + 1;
                    startIdx = i;
                }
            }
        }
        return startIdx == -1 ? "" : s.substring(startIdx, startIdx + maxLen);
    }

    private boolean isNice(String s) {
        int[] lower = new int[26], upper = new int[26];
        for (char c : s.toCharArray()) {
            if (Character.isLowerCase(c)) lower[c - 'a']++;
            else upper[c - 'A']++;
        }
        for (int i = 0; i < 26; i++) {
            if ((lower[i] > 0 && upper[i] == 0) || (upper[i] > 0 && lower[i] == 0)) return false;
        }
        return true;
    }
}
```

More challenges
2421. Number of Good Paths

Testcase: Case 1, Case 2, Case 3
s =
"YazaAay"

Reverse Bits

```java
public class Solution {
    public int reverseBits(int n) {
        int result = 0;
        for (int i = 0; i < 32; i++) {
            result = (result << 1) | (n & 1);
            n >>= 1;
        }
        return result;
    }
}
```



Accepted 600 / 600 testcases passed
Shiv Chauhan submitted at Feb 24, 2025 21:31

Runtime
0 ms | Beats 100.00%

Memory
41.71 MB | Beats 73.57%

Code | Java

```java
public class Solution {
    public int reverseBits(int n) {
        int result = 0;
        for (int i = 0; i < 32; i++) {
            result = (result << 1) | (n & 1);
            n >>= 1;
        }
        return result;
```

More challenges
2119. A Number After a Double Reversal

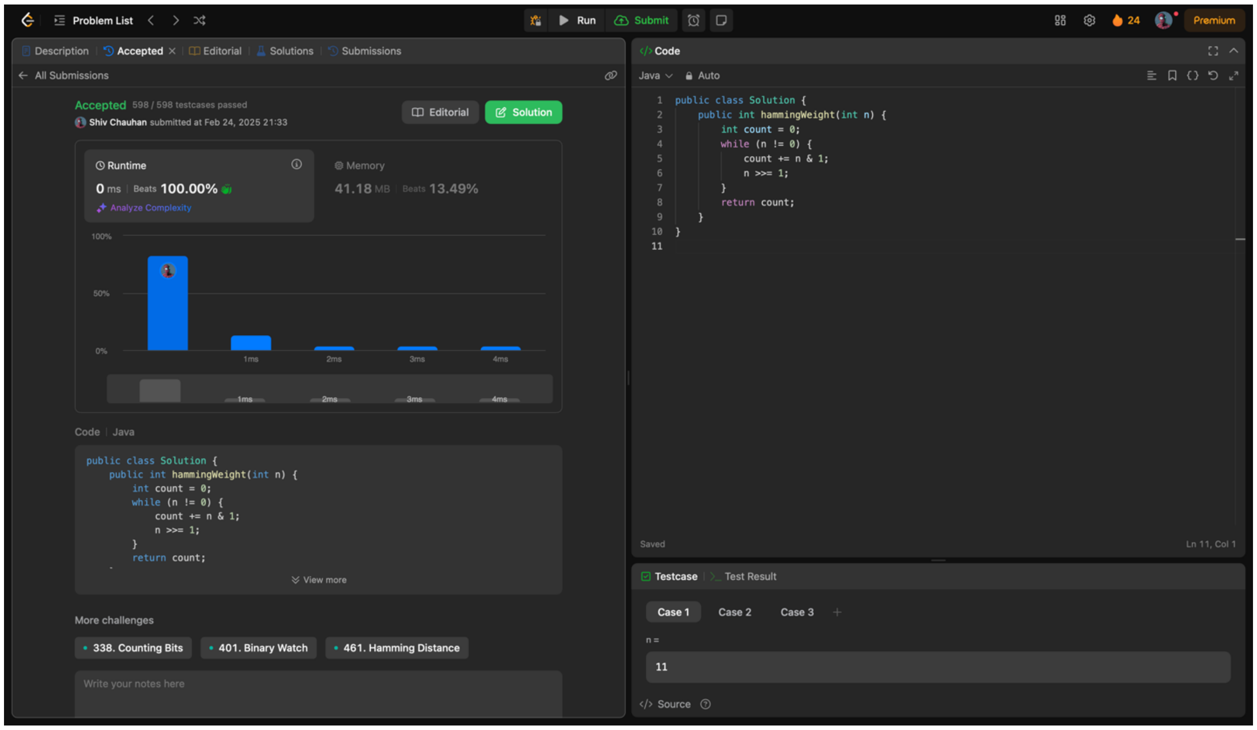
```java
public class Solution {
    public int reverseBits(int n) {
        int result = 0;
        for (int i = 0; i < 32; i++) {
            result = (result << 1) | (n & 1);
            n >>= 1;
        }
        return result;
    }
}
```

Saved
Ln 11, Col 1

n =
00000010100101000001111010011100

Number of 1 Bits

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



Accepted 598 / 598 testcases passed
Shiv Chauhan submitted at Feb 24, 2025 21:33

Runtime
0 ms | Beats 100.00%

Memory
41.18 MB | Beats 13.49%

Code | Java

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

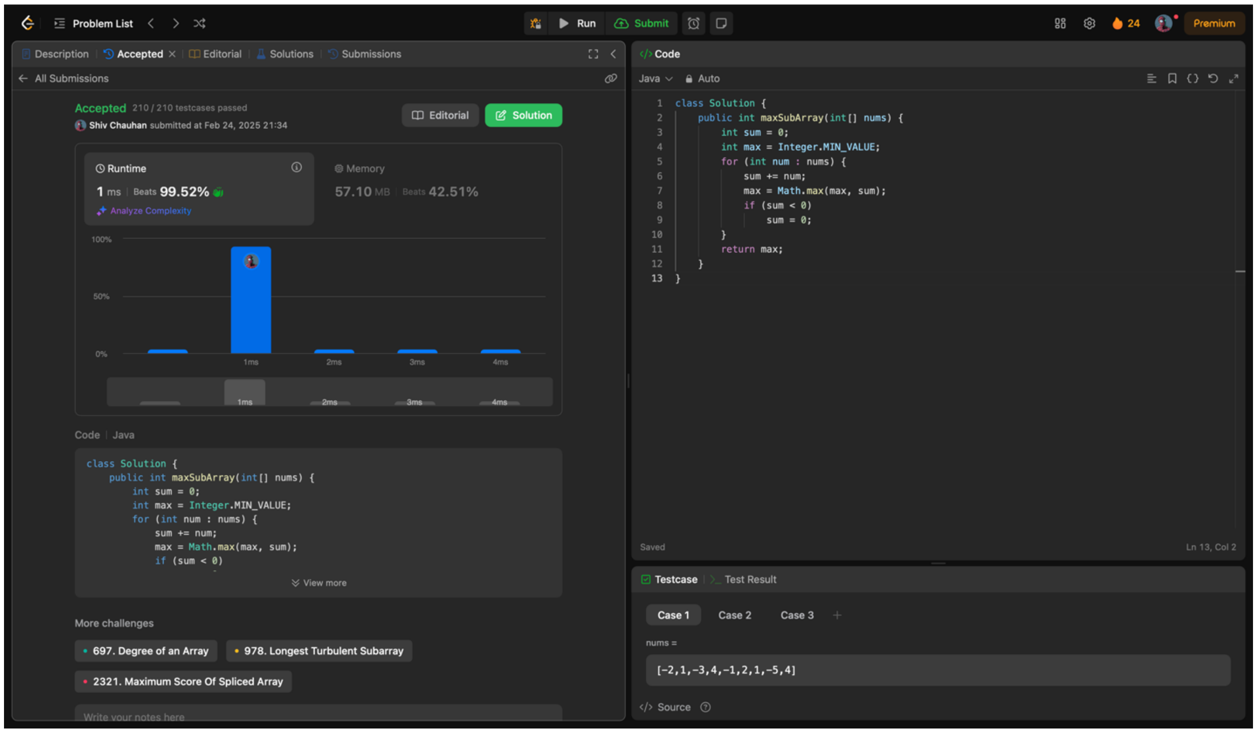
```java
public class Solution {
    public int hammingWeight(int n) {
        int count = 0;
        while (n != 0) {
            count += n & 1;
            n >>= 1;
        }
        return count;
    }
}
```

n =
11

Shiv Chauhan 22BCS15180

Maximum Subarray

```java
class Solution {
    public int maxSubArray(int[] nums) {
        int sum = 0;
        int max = Integer.MIN_VALUE;
        for (int num : nums) {
            sum += num;
            max = Math.max(max, sum);
            if (sum < 0)
                sum = 0;
        }
        return max;
    }
}
```



Accepted 210 / 210 testcases passed
Shiv Chauhan submitted at Feb 24, 2025 21:34

Runtime
1 ms | Beats 99.52%

Memory
57.10 MB | Beats 42.51%

Code | Java

```java
class Solution {
    public int maxSubArray(int[] nums) {
        int sum = 0;
        int max = Integer.MIN_VALUE;
        for (int num : nums) {
            sum += num;
            max = Math.max(max, sum);
            if (sum < 0)
```

View more

More challenges

- 697. Degree of an Array
- 978. Longest Turbulent Subarray
- 2321. Maximum Score Of Spliced Array

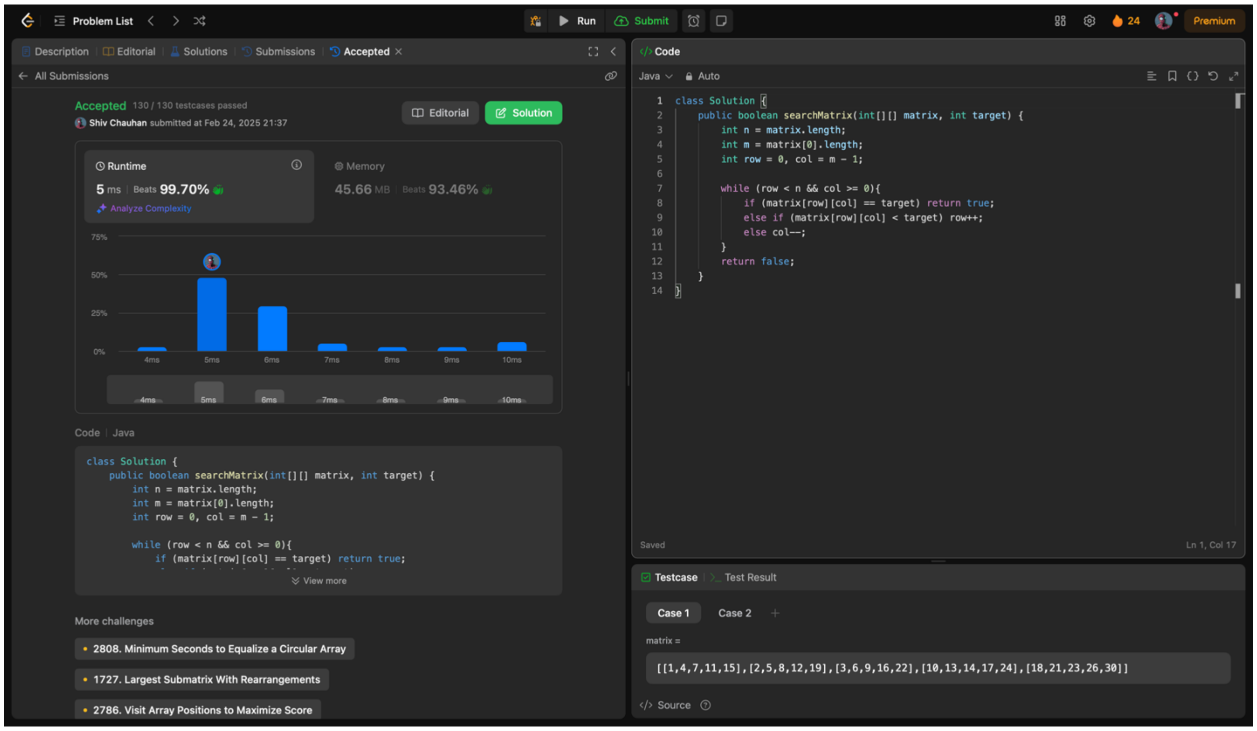
```java
class Solution {
    public int maxSubArray(int[] nums) {
        int sum = 0;
        int max = Integer.MIN_VALUE;
        for (int num : nums) {
            sum += num;
            max = Math.max(max, sum);
            if (sum < 0)
                sum = 0;
        }
        return max;
    }
}
```

nums =

```
[-2,1,-3,4,-1,2,1,-5,4]
```

Search a 2D Matrix II

```java
class Solution {
    public boolean searchMatrix(int[][] matrix, int target) {
        int n = matrix.length;
        int m = matrix[0].length;
        int row = 0, col = m - 1;

        while (row < n && col >= 0){
            if (matrix[row][col] == target) return true;
            else if (matrix[row][col] < target) row++;
            else col--;
        }
        return false;
    }
}
```



Accepted 130 / 130 testcases passed

Shiv Chauhan submitted at Feb 24, 2025 21:37

Runtime 5 ms | Beats 99.70%

Memory 45.66 MB | Beats 93.46%

Code | Java

```java
class Solution {
    public boolean searchMatrix(int[][] matrix, int target) {
        int n = matrix.length;
        int m = matrix[0].length;
        int row = 0, col = m - 1;

        while (row < n && col >= 0){
            if (matrix[row][col] == target) return true;
```

More challenges

- 2808. Minimum Seconds to Equalize a Circular Array
- 1727. Largest Submatrix With Rearrangements
- 2786. Visit Array Positions to Maximize Score

```java
class Solution {
    public boolean searchMatrix(int[][] matrix, int target) {
        int n = matrix.length;
        int m = matrix[0].length;
        int row = 0, col = m - 1;

        while (row < n && col >= 0){
            if (matrix[row][col] == target) return true;
            else if (matrix[row][col] < target) row++;
            else col--;
        }
        return false;
    }
}
```

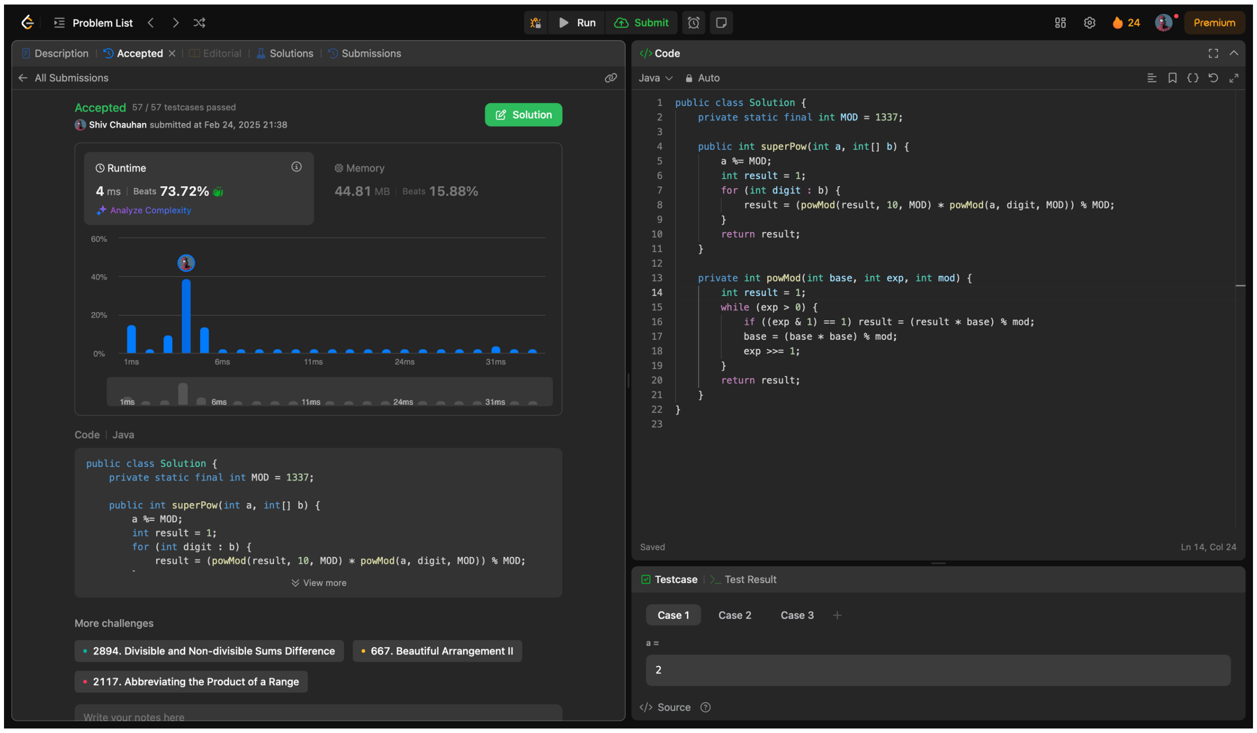
Saved   Ln 1, Col 17

matrix =

```
[[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]]
```

Super Pow

```java
public class Solution {
    private static final int MOD = 1337;

    public int superPow(int a, int[] b) {
        a %= MOD;
        int result = 1;
        for (int digit : b) {
            result = (powMod(result, 10, MOD) * powMod(a, digit, MOD)) % MOD;
        }
        return result;
    }

    private int powMod(int base, int exp, int mod) {
        int result = 1;
        while (exp > 0) {
            if ((exp & 1) == 1) result = (result * base) % mod;
            base = (base * base) % mod;
            exp >>= 1;
        }
        return result;
    }
}
```



Accepted 57 / 57 testcases passed

Shiv Chauhan submitted at Feb 24, 2025 21:38

Runtime
4 ms | Beats 73.72%

Memory
44.81 MB | Beats 15.88%

Code | Java

```java
public class Solution {
    private static final int MOD = 1337;

    public int superPow(int a, int[] b) {
        a %= MOD;
        int result = 1;
        for (int digit : b) {
            result = (powMod(result, 10, MOD) * powMod(a, digit, MOD)) % MOD;
```

More challenges

- 2894. Divisible and Non-divisible Sums Difference
- 667. Beautiful Arrangement II
- 2117. Abbreviating the Product of a Range

Code

Java | Auto

```java
public class Solution {
    private static final int MOD = 1337;

    public int superPow(int a, int[] b) {
        a %= MOD;
        int result = 1;
        for (int digit : b) {
            result = (powMod(result, 10, MOD) * powMod(a, digit, MOD)) % MOD;
        }
        return result;
    }

    private int powMod(int base, int exp, int mod) {
        int result = 1;
        while (exp > 0) {
            if ((exp & 1) == 1) result = (result * base) % mod;
            base = (base * base) % mod;
            exp >>= 1;
        }
        return result;
    }
}
```

a =

2