Description | Editorial | Solutions (1.7K) | Submissions

628. Maximum Product of Three Numbers

Easy | 3.8K | 621

Companies

Given an integer array `nums`, *find three numbers whose product is maximum and return the maximum product.*

Example 1:

```
Input: nums = [1,2,3]
Output: 6
```

Example 2:

```
Input: nums = [1,2,3,4]
Output: 24
```

Example 3:

```
Input: nums = [-1,-2,-3]
Output: -6
```

Constraints:

- `3 <= nums.length <= 10^4`
- `-1000 <= nums[i] <= 1000`

Accepted 269.7K | Submissions 591.3K | Acceptance Rate 45.6%

Java | Auto

```java
class Solution {
    int maxi(int[] a,int n)
    {
        int max=(int)-9999999;
        int f=-1;
        for(int i=0;i<n;i++)
        {
            if(max<=a[i])
            {
                max=a[i];
                f=i;
            }
        }
        a[f]=(int)-9999999;
        return max;
    }
    int min(int[] a,int n)
    {
        int max=(int)999999;
        int f=-1;
```

Testcase | Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

nums =
[1,2,3]

Output

6

Expected

Console | Run | Submit

Description | Editorial | Solutions (390) | Submissions

864. Shortest Path to Get All Keys

Hard | 1.5K | 68

Companies

You are given an `m x n` grid `grid` where:

- `'.'` is an empty cell.
- `'#'` is a wall.
- `'@'` is the starting point.
- Lowercase letters represent keys.
- Uppercase letters represent locks.

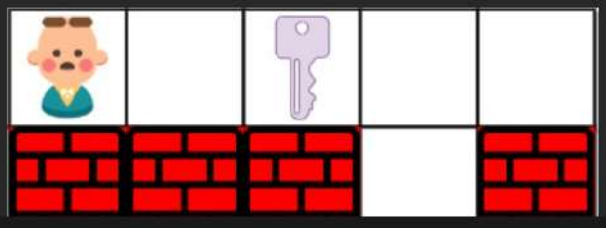
You start at the starting point and one move consists of walking one space in one of the four cardinal directions. You cannot walk outside the grid, or walk into a wall.

If you walk over a key, you can pick it up and you cannot walk over a lock unless you have its corresponding key.

For some `1 <= k <= 6`, there is exactly one lowercase and one uppercase letter of the first `k` letters of the English alphabet in the grid. This means that there is exactly one key for each lock, and one lock for each key; and also that the letters used to represent the keys and locks were chosen in the same order as the English alphabet.

Return *the lowest number of moves to acquire all keys*. If it is impossible, return `-1`.

Example 1:



Java | Auto

```java
import java.util.*;

class Solution {
    public int shortestPathAllKeys(String[] grid) {
        int m = grid.length;
        int n = grid[0].length();

        Map<Character, Integer> key_bit = new HashMap<>();
        int bit_start = 0;

        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                if (Character.isLowerCase(grid[i].charAt(j))) {
                    key_bit.put(grid[i].charAt(j), bit_start++);
                }
            }
        }

        int mask_end = (1 << bit_start) - 1;
        int mask_size = (1 << bit_start);
```

Testcase | Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

grid =
["@.a..","###.#","b.A.B"]

Output

8

Expected

Console | Run | Submit