

## 46. Permutations

Solved

Medium

Topics

Companies

Given an array `nums` of distinct integers, return *all the possible permutations*. You can return the answer in **any order**.

### Example 1:

**Input:** `nums = [1,2,3]`

**Output:** `[[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]`

### Example 2:

**Input:** `nums = [0,1]`

**Output:** `[[0,1],[1,0]]`

### Example 3:

**Input:** `nums = [1]`

**Output:** `[[1]]`

### Constraints:

- `1 <= nums.length <= 6`

The screenshot shows a code editor interface with two main panes. The left pane displays the problem description and a submission status of 'Accepted' for a user named 'ZhukovIlya' submitted at Jul 06, 2024 20:39. It includes performance metrics: Runtime 102 ms (Beats 58.19%) and Memory 46.74 MB (Beats 24.86%). A bar chart shows the distribution of runtime times, with a note that 2.57% of solutions used 113 ms of runtime. The right pane shows the C# code for the solution, which uses a recursive helper method to generate all permutations.

```
using System;
using System.Collections.Generic;

public class Solution {
    public IList<IList<int>> Permute(int[] nums) {
        var results = new List<IList<int>>();
        PermuteHelper(nums, 0, results);
        return results;
    }

    private void PermuteHelper(int[] nums, int start, List<IList<int>> results) {
        if (start == nums.Length) {
            results.Add(new List<int>(nums));
            return;
        }
        for (int i = start; i < nums.Length; i++) {
            Swap(nums, start, i);
            PermuteHelper(nums, start + 1, results);
            Swap(nums, start, i);
        }
    }

    private void Swap(int[] nums, int i, int j) {
        int temp = nums[i];
        nums[i] = nums[j];
        nums[j] = temp;
    }
}
```

Код:

```
using System;
using System.Collections.Generic;

public class Solution
{
    public IList<IList<int>> Permute(int[] nums)
    {
```

```

        var results = new List<IList<int>>();
        PermuteHelper(nums, 0, results);
        return results;
    }

    private void PermuteHelper(int[] nums, int start, List<IList<int>> results)
    {
        if (start == nums.Length)
        {
            results.Add(new List<int>(nums));
            return;
        }

        for (int i = start; i < nums.Length; i++)
        {
            Swap(nums, start, i);
            PermuteHelper(nums, start + 1, results);
            Swap(nums, start, i); // backtrack
        }
    }

    private void Swap(int[] nums, int i, int j)
    {
        int temp = nums[i];
        nums[i] = nums[j];
        nums[j] = temp;
    }
}

```