```
1
2
   给定一个没有重复数字的序列,返回其所有可能的全排列。
3
4
   示例:
5
6
   输入: [1,2,3]
7
   输出:
8
    [1,2,3],
9
10
    [1,3,2],
11
    [2,1,3],
12
    [2,3,1],
13
    [3,1,2],
14
    [3,2,1]
15 ]
16
17
   来源:力扣(LeetCode)
   链接: https://leetcode-cn.com/problems/permutations
18
19
   著作权归领扣网络所有。商业转载请联系官方授权,非商业转载请注明出处。
20 */
```

## 分析:

- 递归法:依次递归每一种排列可能,直到数组长度为1,即可推入结果容器.
- 方法三:回溯法步骤
  - o choose
  - explore
  - unchoose
  - 。 对于无重复数据的情况,遍历到所有可能保存即可.

## 方法一:C++\_递归法

```
class Solution
 1
 2
 3
        private:
             vector<vector<int>> ret_val;
 4
 5
             void __rank(vector<int>& nums, vector<int> level_rank)
 6
             {
                 if(nums.size() == 1)
 8
 9
                     level_rank.push_back(nums[0]);
10
                     ret_val.push_back(level_rank);
11
                     return ;
12
                 for(int i = 0; i < nums.size() ; i++)</pre>
13
14
15
                     vector<int> temp;
16
                     for(int j = 0; j < nums.size(); j++)
17
18
                         if(i!=j)
19
                         {
```

```
20
                           temp.push_back(nums[j]);
21
                       }
22
                       else
23
                       {
24
                           level_rank.push_back(nums[j]);
25
                       }
26
                   }
27
28
                   __rank(temp,level_rank);
29
                   level_rank.pop_back();
30
               }
31
            }
32
33
        public:
34
            vector<vector<int>>> permute(vector<int>& nums)
35
36
                ret_val.clear();
37
               vector<int> temp;
38
               if(nums.size() !=0)
39
40
                   __rank(nums,temp);
41
               }
42
               return ret_val;
43
           }
44
    };
45
    /*
46
47
    执行结果:
48
   通过
49
    显示详情
50 执行用时 :20 ms, 在所有 C++ 提交中击败了63.44% 的用户
   内存消耗 :10.4 MB, 在所有 C++ 提交中击败了10.14%的用户
51
52 */
```

## 方法三:C++\_回溯法逻辑整理

```
// https://leetcode-cn.com/submissions/detail/37429272/
 2
    // 回溯法 https://www.bilibili.com/video/av76286065
 3
 4
 5
    class Solution
 6
    {
 7
 8
        private:
 9
10
            void helper(
                             vector<vector<int>>&
                                                      vvi
11
                             vector<int>&
                                                      cur
12
                             vector<int>&
                                                      nums
13
                             vector<int>&
                                                      visited
                         )
14
            {
15
16
                if(cur.size() == nums.size())
17
                 {
18
                     vvi.push_back(cur);
19
                     return;
20
                }
```

```
21
22
              for(int i = 0 ; i < nums.size() ; i++)</pre>
23
24
25
                  if(visited[i])
26
                  {
27
                     continue;
28
                  }
29
                  else
30
                  {
31
                     cur.push_back(nums[i]);
                                           // choose
32
                     visited[i] = 1;
33
                     34
                     visited[i] = 0;
                                                    // unchoose
35
                     cur.pop_back();
36
                  }
37
38
              }
39
          }
40
41
42
43
       public:
44
          vector<vector<int>>> permute(vector<int>& nums)
45
          {
46
              vector<vector<int>> vvi;
47
              vector<int>
                               cur;
48
              vector<int>
                              visited(nums.size(),0);
49
50
              if(nums.size() < 1)</pre>
51
52
                  return vvi;
53
              }
54
              helper(vvi, cur, nums , visited);
55
              return vvi;
56
57
          }
58
59
   };
60
   /*
61
62
   执行结果:
63
   通过
64
   显示详情
65 执行用时 :24 ms, 在所有 cpp 提交中击败了33.40% 的用户
66 内存消耗 :9.1 MB, 在所有 cpp 提交中击败了88.83%的用户
   */
67
```