## 模拟面试题10（我亲自经历的面试题）

### 1. 排序

**冒泡排序**

class Solution:
 def sortArray(self, nums: List[int]) -> List[int]:
 flag = True
 while flag:
 flag = False
 for i in range(len(nums)-1):
 if nums[i] > nums[i+1]:
 nums[i], nums[i+1] = nums[i+1], nums[i]
 flag = True
 return nums

**选择排序**

class Solution:
 def sortArray(self, nums: List[int]) -> List[int]:
 for i in range(len(nums)):
 lowest\_index = i
 for j in range(i+1, len(nums)):
 if nums[j] < nums[lowest\_index]:
 lowest\_index = j
 nums[i], nums[lowest\_index] = nums[lowest\_index], nums[i]
 return nums

**快速排序**

class Solution:
 def sortArray(self, nums: List[int]) -> List[int]:
 if len(nums) <= 1:
 return nums
 pivot = nums[len(nums) // 2]
 left = [x for x in nums if x < pivot]
 middle = [x for x in nums if x == pivot]
 right = [x for x in nums if x > pivot]
 return self.sortArray(left) + middle + self.sortArray(right)

class Solution:
 def randomized\_partition(self, nums, l, r):
 pivot = random.randint(l, r)
 nums[pivot], nums[r] = nums[r], nums[pivot]
 i = l - 1
 for j in range(l, r):
 if nums[j] < nums[r]:
 i += 1
 nums[j], nums[i] = nums[i], nums[j]
 i += 1
 nums[i], nums[r] = nums[r], nums[i]
 return i

 def randomized\_quicksort(self, nums, l, r):
 if r - l <= 0:
 return
 mid = self.randomized\_partition(nums, l, r)
 self.randomized\_quicksort(nums, l, mid - 1)
 self.randomized\_quicksort(nums, mid + 1, r)

 def sortArray(self, nums: List[int]) -> List[int]:
 self.randomized\_quicksort(nums, 0, len(nums) - 1)
 return nums

冒泡 稳定 选择 不稳定 插入 稳定

参考资料： https://www.cnblogs.com/huang-yc/p/9774287.html https://blog.csdn.net/MobiusStrip/article/details/83785159?depth\_1-utm\_source=distribute.pc\_relevant.none-task&utm\_source=distribute.pc\_relevant.none-task