排序

1. 选择排序

从数组中选择最小元素，将它与数组的第一个元素交换位置。再从数组剩下的元素中选择出最小的元素，将它与数组的第二个元素交换位置。不断进行这样的操作，直到将整个数组排序。

2. 冒泡排序

从左到右不断交换相邻逆序的元素，在一轮的循环之后，可以让未排序的最大元素上浮到右侧。在一轮循环中，如果没有发生交换，那么说明数组已经是有序的，此时可以直接退出。

3. 插入排序

每次都将当前元素插入到左侧已经排序的数组中，使得插入之后左侧数组依然有序。

4. 希尔排序

5. 归并排序

归并排序的思想是将数组分成两部分，分别进行排序，然后归并起来。

拆分：

合并：合并有序数组

```
static void merge_sort_recursive(int[] arr, int[] result, int start, int end) {
    if (start >= end)
        return;
    int len = end - start, mid = (len >> 1) + start;
    int start1 = start, end1 = mid;
    int start2 = mid + 1, end2 = end;
//拆分
    merge_sort_recursive(arr, result, start1, end1);
    merge_sort_recursive(arr, result, start2, end2);
//合并
    int k = start;
    while (start1 <= end1 && start2 <= end2)
        result[k++] = arr[start1] < arr[start2] ? arr[start1++] : arr[start2++];
    while (start1 <= end1)
        result[k++] = arr[start1++];
    while (start2 <= end2)
```

```
        result[k++] = arr[start2++];
}
```

6. 快速排序

① 从数列中挑出一个元素，称为 “基准”（pivot），

② 重新排序数列，所有元素比基准值小的摆放在基准前面，所有元素比基准值大的摆在基准的后面（相同的数可以到任一边）。在这个分区退出之后，该基准就处于数列的中间位置。这个称为分区（partition）操作。

③ 递归地（recursive）把小于基准值元素的子数列和大于基准值元素的子数列排序。

```
private void quick_sort_recursive(int start, int end) {
        if (start >= end)
            return;
        int mid = arr[end];
        int left = start, right = end - 1;
        while (left < right) {
            while (arr[left] <= mid && left < right)
                left++;
            while (arr[right] >= mid && left < right)
                right--;
            swap(left, right);
        }
        if (arr[left] >= arr[end])
            swap(left, end);
        else
            left++;
        quick_sort_recursive(start, left - 1);
        quick_sort_recursive(left + 1, end);
    }
    public void sort(int[] arrin) {
        arr = arrin;
        quick_sort_recursive(0, arr.length - 1);
    }
```

7. 堆排序