## 1 Sorting I

Show the steps taken by each sort on the following unordered list:

`106, 351, 214, 873, 615, 172, 333, 564`

(a) Insertion sort. Show the sorted and unsorted portions at every step.

(b) Selection sort. Show the sorted and unsorted portions at every step.

(c) Merge sort. Show how the list is broken up at every step.

(d) Use heapsort to sort the following array (hint: draw out the heap):

`106, 615, 214, 873, 351.`

(e) Give an example of a situation when using insertion sort is more efficient than using merge sort.

## 2 Sorting II

Match the sorting algorithms to the sequences, each of which represents several intermediate steps in the sorting of an array of integers.
Algorithms: Heapsort, merge sort, insertion sort, selection sort.

(a)
```
12, 7, 8, 4, 10, 2, 5, 34, 14
2, 4, 5, 7, 8, 12, 10, 34, 14
```

(b)
```
23, 45, 12, 4, 65, 34, 20, 43
12, 23, 45, 4, 65, 34, 20, 43
```

(c)
```
45, 23, 5, 65, 34, 3, 76, 25
23, 45, 5, 65, 3, 34, 25, 76
5, 23, 45, 65, 3, 25, 34, 76
```

(d)
```
12, 32, 14, 34, 17, 38, 23, 11
12, 14, 17, 32, 34, 38, 23, 11
```

## 3 Runtimes

Fill in the best and worst case runtimes of the following sorting algorithms with respect to $n$, the length of the list being sorted, along with when that runtime would occur. For quicksort, assume the pivot is always the first item in the sublist being sorted.

| | Insertion sort | Selection sort | Merge sort | Heapsort |
|---|---|---|---|---|
| Worst case | | | | |
| Best case | | | | |

## 4 MergeTwo

Suppose you are given two sorted arrays of ints. Fill in the method mergeTwo to return a new array containing all of the elements of both arrays in sorted order. Duplicates are allowed (if an element appears $s$ times in $a$ and $t$ times in $b$, then it should appear $s+t$ times in the returned array).

```
public static int[] mergeTwo(int[] a, int[] b) {




}
```

What is the runtime of this method?