CS 61B
Spring 2021

Basic Sorts

Discussion 12: April 12, 2021

1 All Sorts Of Sorts

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

```
0, 4, 2, 7, 6, 1, 3, 5
```

(a) Insertion sort

(b) Selection sort

(c) Merge sort

(d) Use heapsort to sort the following array (hint: draw out the heap). Draw out the array at each step:

```
0, 6, 2, 7, 4
```

2 Sorta Interesting, Right?

(a) What does it mean to sort ”in place”, and why would we want this?

(b) What does it mean for a sort to be ”stable”? Which sorting algorithms that we have seen are stable?

(c) Which algorithm would run the fastest on an already sorted list?

(d) Given any list, what is the ideal pivot for quicksort?

(e) So far, in class, we’ve mostly applied our sorts to lists of numbers. In practice, how would we typically make sure our sorts can be applied to other types?

3 Zero One Two-Step

(a) Given an array that only contains 0's, 1's and 2's, write an algorithm to sort it in linear time. You may want to use the provided helper method, `swap`.

```
public static int[] specialSort(int[] arr) {
    int front = 0;
    int back = arr.length - 1;
    int curr = 0;
















}
```

```
private static void swap(int[] arr, int i, int j) {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
```

(b) We just wrote a linear time sort, how cool! Can you explain in a sentence or two why we can't always use this sort, even though it has better runtime than Mergesort or Quicksort?