Benjamin Zheng
CSDS 233
Assignment 4 Written

1) Algorithm Design → Reverse two sum

- Iterate through array and put integer and index in hash table
- Check if A-D exists in hash table.
    If yes, return a pair (A, A-D)
    If no, check next integer
- If loop is done running, then, no such pair exists.

Pseudocode:

```
Map indexMap = new Hashmap

for(i=0; i<A.length; i++){
    complement = A[i]-D
    if (indexMap.containsKey(complement)) {
        return int[] pair = {indexMap.get(complement), i}
    }
    indexMap.put(A[i], i)
}
return null
```

2) Result of inserting keys into hash table
4371, 1323, 6173, 4199, 4344, 9679, 1989
hash function $h(x) = x \bmod 10$
hash table size 10.

4371 mod 10 = 1
1323 mod 10 = 3
6173 mod 10 = 3
4199 mod 10 = 9
4344 mod 10 = 4
9679 mod 10 = 9
1989 mod 10 = 9

a) Separate Chaining

0	1	2	3	4	5	6	7	8	9

1 → 4371
3 → 6173 → 1323
4 → 4344
9 → 1989 → 9679 → 4199

b) Open addressing with linear probing

0	1	2	3	4	5	6	7	8	9
9679	4371	1989	1323	6173	4344				4199

c) Open addressing with quadratic probing

0	1	2	3	4	5	6	7	8	9
9679	4371		1323	6173	4344			1989	4199

(6173+1) mod 10 = 4
(4344+1) mod 10 = 5
(9679+1) mod 10 = 0
(1989+1) mod 10 = 0
(1989+4) mod 10 = 3
(1989+9) mod 10 = 8

d) Open addressing with double hashing: $h_2(x) = 7-(x \bmod 7)$

Using the textbook's definition of double hashing (using first hash function as base, then adding second hash function if taken)

$7-(4371 \bmod 7) = 4$
$7-(1323 \bmod 7) = 7$
$7-(6173 \bmod 7) = 1$

$7-(4199 \bmod 7) = 1$
$7-(4344 \bmod 7) = 3$
$7-(9679 \bmod 7) = 2$

$7-(1989 \bmod 10) = 6$

0	1	2	3	4	5	6	7	8	9
	4371		1323	6173	9679		4344		4199

→ unable to find new space given increments of second hash function.

$4371 \bmod 23 = 1$
$1323 \bmod 23 = 12$
$6173 \bmod 23 = 9$
$4199 \bmod 23 = 13$
$4344 \bmod 23 = 20$

$9679 \bmod 23 = 19$
$1989 \bmod 23 = 11$

According to textbook, doubling array size to next prime (23)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
	4371								6173		1989	1323	4199						9679	4344		

3) Sorting: 9 8 8 5 7 7 4 4 4 2

Smallest to largest:

a) Selection Sort:

9 8 8 5 7 7 4 4 4 2
2 8 8 5 7 7 4 4 4 9
2 4 8 5 7 7 8 4 4 9
2 4 4 5 7 7 8 8 4 9
2 4 4 4 7 7 8 8 5 9
2 4 4 4 5 7 8 8 7 9
2 4 4 4 5 7 8 8 7 9
2 4 4 4 5 7 7 8 8 9
2 4 4 4 5 7 7 8 8 9
2 4 4 4 5 7 7 8 8 9
2 4 4 4 5 7 7 8 8 9

Items swapped

b) Insertion Sort:

9 8 8 5 7 7 4 4 4 2
8 9 8 5 7 7 4 4 4 2
8 8 9 5 7 7 4 4 4 2
5 8 8 9 7 7 4 4 4 2
5 7 8 8 9 7 4 4 4 2
5 7 7 8 8 9 4 4 4 2
4 5 7 7 8 8 9 4 4 2
4 4 5 7 7 8 8 9 4 2
4 4 4 5 7 7 8 8 9 2
2 4 4 4 5 7 7 8 8 9

Sorted Partition

c) Quicksort by partitioning last element

9 8 8 5 7 7 4 4 4 2   Pivot = 2

2   9 8 8 5 7 7 4 4 4   Pivot = 4

Pivot = 4   4 4 4   9 8 8 5 7 7   Pivot = 7

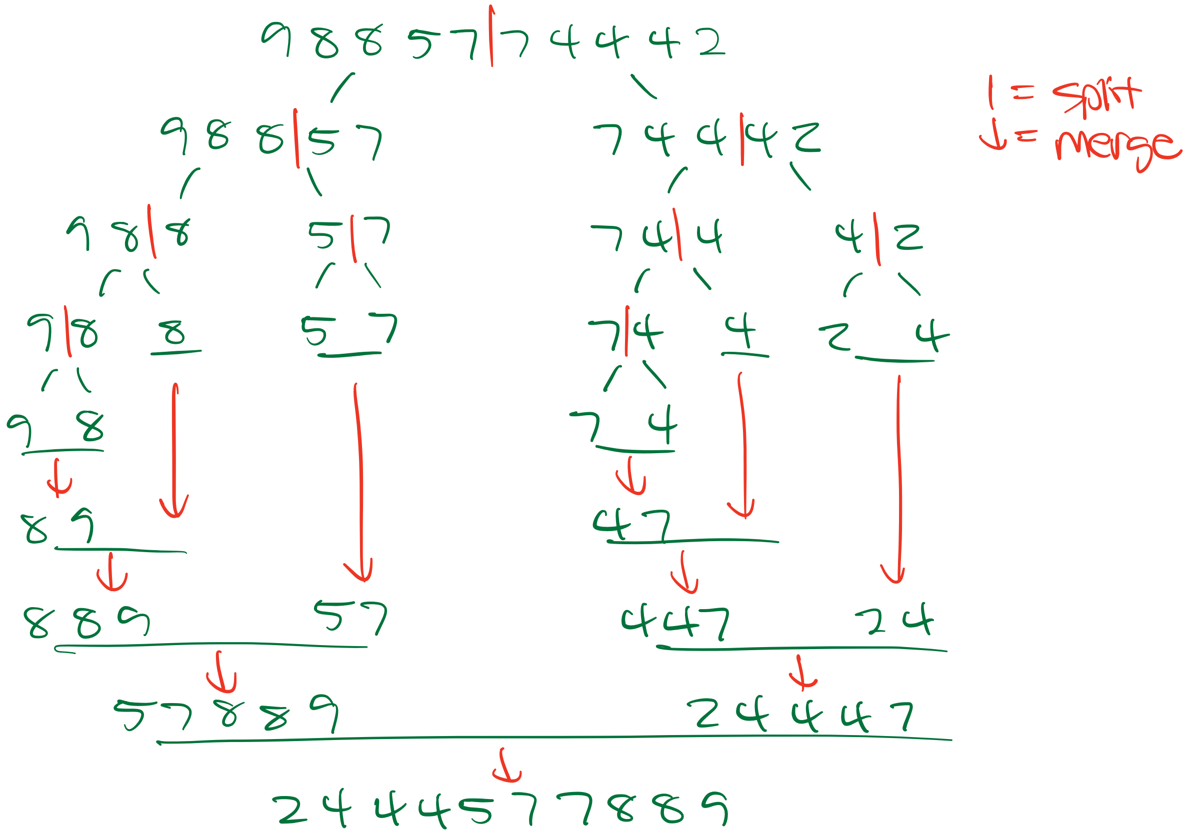
Pivot = 7   5 7 7   9 8 8   Pivot = 8

5   7 7   8 8   9

*Note the tree does not show the swapping of the partitions method. It only shows the partitions the algorithm makes.

Sorted array: 2 4 4 4 5 7 7 8 8 9

d) Mergesort



9 8 8 5 7 | 7 4 4 4 2
9 8 8 | 5 7
7 4 4 | 4 2
9 8 | 8
5 | 7
7 4 | 4
4 | 2
9 | 8
8
5 7
7 | 4
4
2 4
9 8
7 4
8 9
4 7
8 8 9
5 7
4 4 7
2 4
5 7 8 8 9
2 4 4 4 7
2 4 4 4 5 7 7 8 8 9
| = split
↓ = merge