HW6

Name: Key

1. Exercise 5.3.2 (pg.185) in the book.

just insert an "else if" after else

else if ($T_{Left} = \emptyset$ && $T_{right} = \emptyset$) return 1

(the alg. can't return a number > 0, obviously)

2. Exercise 5.5.1 a) and b), pg. 197.

a)
```
closestPair1D( pts[0..n-1] ){
    sort(pts) // pts in 1D are just numbers...
    return cp1D(pts, 0, n-1)
}

cp1D(pts, l, r){
    if (r-l < 3) {
        List<int> Lpts = <list of points>
        return BruteforceClosestPair(Lpts)
    }
    m = ⌊n/2⌋ - 1
    dLeft = cp1D(pts, l, m)
    dRight = cp1D(pts, m+1, r)
    dBorder = pts[m+1] - pts[m]
    return min(dLeft, dRight, dBorder)
}
```

b) Not really - can just sort and check adjacent pairs

many reasonable approaches

Mergesort((4 2 6 8 7 3 1 5))

Merge (Mergesort(4 2 6 8), Mergesort(7 3 1 5))

Merge (Merge(M.S.(4 2), M.S(6 8)), Merge(M.S.(7 3), M.S.(1 5)))

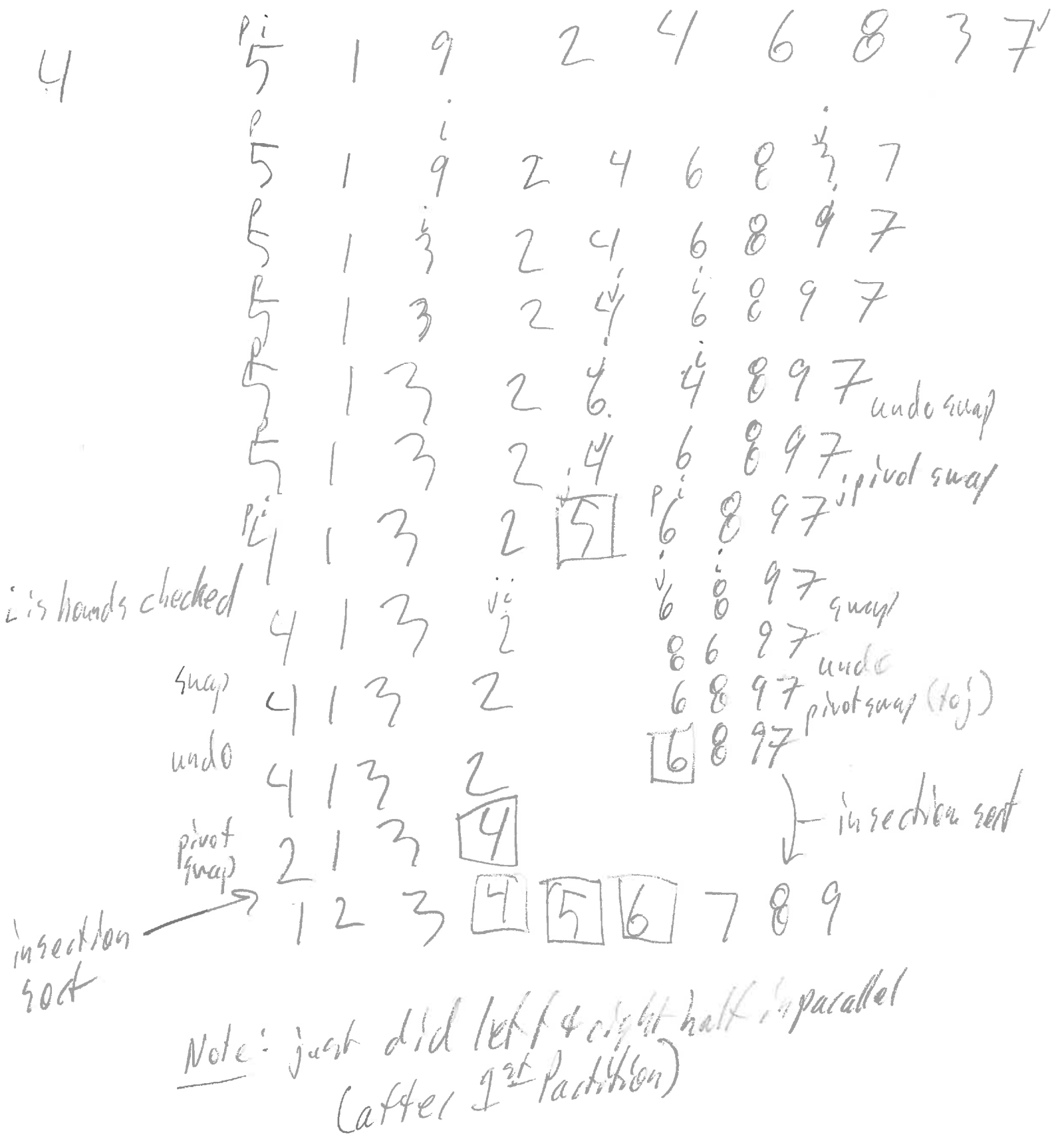
Merge (Merge (Merge (MS(4), MS(2)), Merge(MS(6), MS(...

(M.S.=MergeSort(1)) are the actual calls, ...
can just do mergesorts

answer

4	2	6	8	7	3	1	5	initial
2	4	6	8	3	7	1	5	2
2	4	6	8	1	3	5	7	4
1	2	3	4	5	6	7	8	8

<done>



Note: just did left & right half in parallel (after 1st partition)