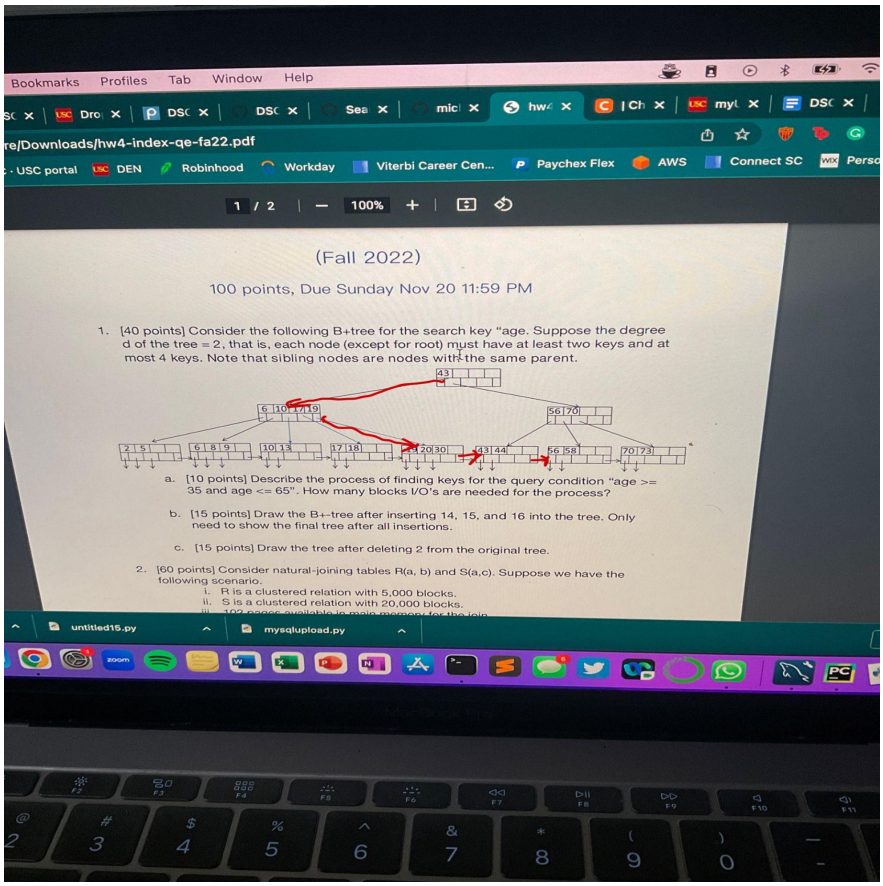
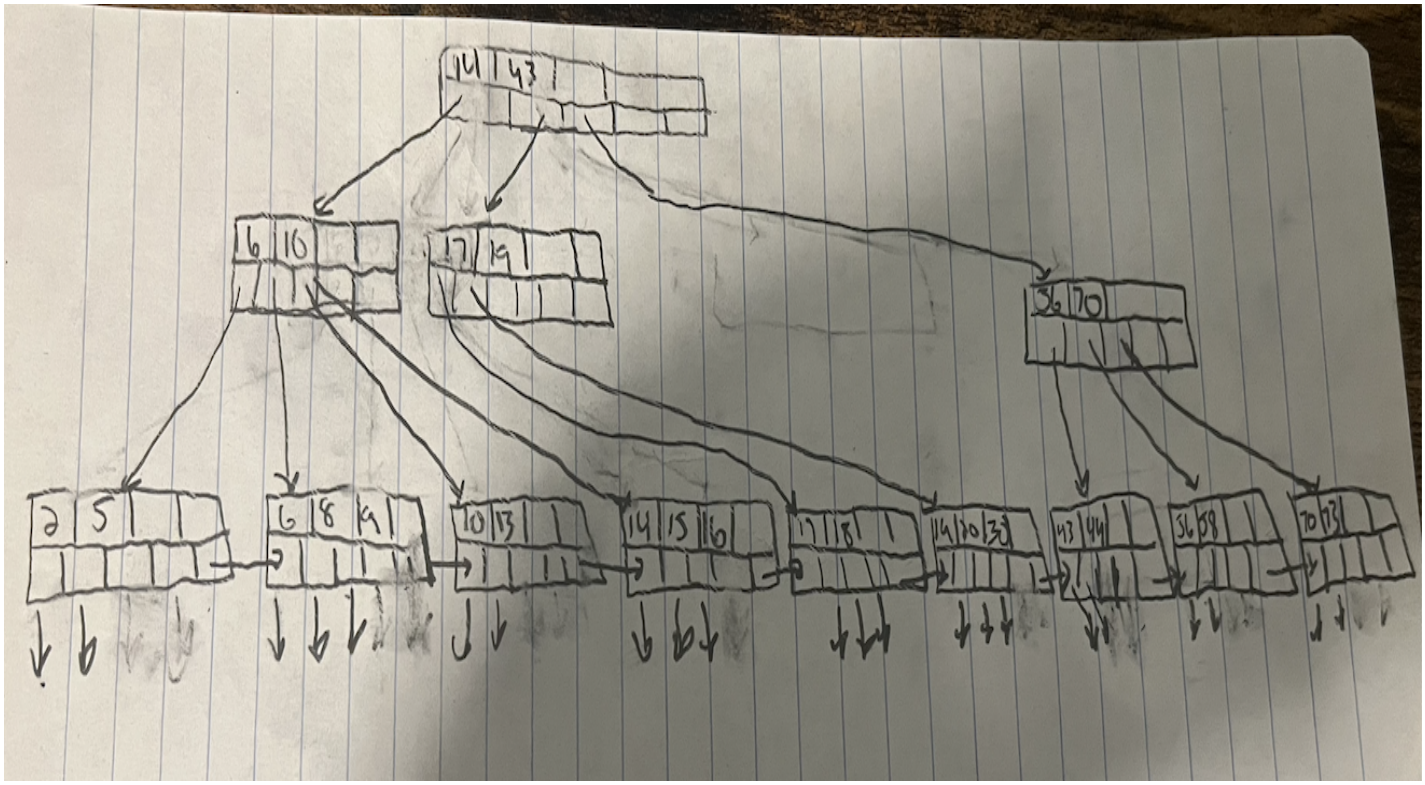
1a. The path below is the path I would take. Since 35 is less than 45, we go left from the root. Next we go to the internal node, since 35 is greater than 19, we go right. We keep going right until we get near 65. The last red line we check the last block and see that 65 is less than 70, so we stop. We read 6 blocks in tota.

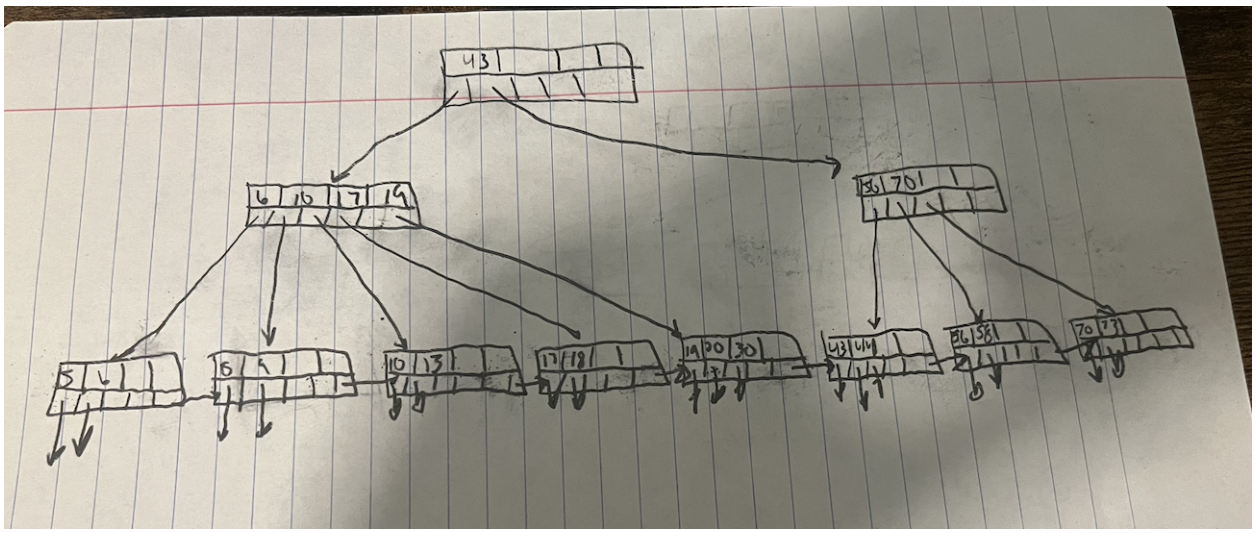


1b.



In this diagram, we split both on the middle of 10,13,14,15,16 (which 14 is in the middle so it goes right) and then we split again 6,10,17,19 because there are too many pointers and we add 14 to the root node.

1c.



For this one, once we delete two theres only one 5 in that leaf node. We move the 6 to left most leaf node, so there is a minimum of two numbers and leave the 8 and 9.

2a.

Cost of R = B(R) + B(R)/(M-2) * B(S)

5000 + 5000/(102-2) * 20000 = **1,005,000**

2b.

Cost of S = B(S) + B(S)/(M-2) * B(R)

20000 + 20000 (102-2) *5000 = **1,020,000**

2c.

Pass 1:
Sort S = 20,000/ 100 = 200 runs
Sort R = 5,000/100 = 50 runs

Cost of Reading/Writing:
2 B(R) + 2 B(S)

Pass 2:
However, since B(S) = 20,000, we need to run two times:
2 B(R) + 4 B(S)

Pass 3:
We then need to merge after, at the cost of B(R) + B(S):

TOTAL COST:
3 B(R) + 5 B(S)
3* 5000 + 5* 20000 = **115,000**

2d.

Pass 1:
Load and write the bucket
2 B(S) + 2 B(R)

Pass 2:
Joining
B(R) + B(S)

Total Cost: 3B(R) + 3B(S)

= 3* 5000 + 3* 20000 = **75,000**

Therefore, partition,based-hashing is the most efficient because it runs the smallest amount of i/o blocks and therefore is the fastest.