Q1: Query processing (3 points)

Consider the natural join of the relation R and S on attribute A. Neither relations have any indexes built on them. Assume that R and S have 80000 and 20000 blocks, respectively. The cost of a join is the number of its block I/Os accesses

1. Assume that there are 300 buffer blocks available in the main memory. We would like to have the output of join sorted according to attribute A. What is the fastest join algorithm for computing the join of R and S? What is the cost of this algorithm?

   - If we do not have an index that matches the join condition on either table, we cannot use index nested loops. Since the size of the relations are not greater than or equal to M^2, we can say that sort-merge join would be the fastest algorithm for computing the join of R and S. The cost of this join would be 5 (B(R) + B(S)) = 5 × (80000 + 20000) = 50000.

2. Assume that there are 40 buffer blocks available in the main memory. What is the fastest join algorithm to compute the join of R and S? What is the cost of this algorithm?

   - Since M^2 should be less than the relations specifies we cannot use sort-merge join algorithm. We don't use hash because B(R) should be less than equal to M^2.
     General Multi-way Merge: $[O(B(R)*\log_m B(R))+O(B(S)*\log_m B(s))+2B(s)+2B(R)]$= [O(80000*3.060491)+O(20000*2.684687)+2(20000)+2(80000)]= **498533.02**
     Improved Nested Loop: B(R) + [B(R) / (M-2)]*B(S) = 80000 + [80000/(40-2)]*20000=**42185263.2**
     The fastest join would be the General Multi-way Merge with the cost of **498533.2**

3. Assume that there are 200 buffer blocks available in the main memory. What is the fastest join algorithm to compute the join of R and S? What is the cost of this algorithm?

   - We don't use hash because B(R) should be less than equal to M^2.
     We cannot apply the optimized sort merge since B(R) + B(S) is not less than or equal to 40000.
     Hash memory is good with smaller amount of memory. Since B(S) <= M^2 i.e. B (20000) <= 40000
     So we can use the hash-join algorithm for this join. Cost would be 3 B(R) + 3 B(S) that is 3 B(80000) + 3 B(20000) = 300000 number of I/O access.

Q2. Coding