1. Suppose you need to sort a relation of 40 gigabytes, with 4 kilobyte blocks, using a memory size of 40 megabytes. Assume that t_S = 5ms and t_T = 0.1ms. Find the cost of sorting the relation, in seconds, with b_b = 1, 9, 30, 300, and 900, where b_b is the number of buffer blocks allocated to each input run and output run.

2. Apply merge-join to the following numerical example used in the lecture.

$n_{student}$ =5,000 $b_{student}$ =100

n_{takes} =10,000 b_{takes} =400

(a) What will be the cost if the relations are ***not*** sorted and the memory size is still ***3*** blocks?

(b) What will be the cost if the memory size is increased to ***25*** blocks and the relations are ***not*** sorted?

(c) In order to reduce the number of seeks (without increasing the number of block transfers), what is the number of buffer blocks (i.e., b_b) that should be allocated

(i) to each run and the output in the merge step of sorting and,

(ii) for buffering each relation and the output in the join step?

(d) What will be the cost of part (c)?

3. Consider two relations $r_1(A, B, C)$ and $r_2(C, D, E)$. For r_1 it has 20,000 tuples and 25 tuples fit on one block. For r_2, it has 45,000 tuples and 30 tuples fit on one block. Which join algorithm gives the lowest worst-case cost estimate for $r_1 \bowtie r_2$? Consider only number of block transfers in this exercise. Specify the minimum amount of memory in number of blocks for the worst-case estimates.