CSE 444 – Homework 2
Indexes and Operator Algorithms

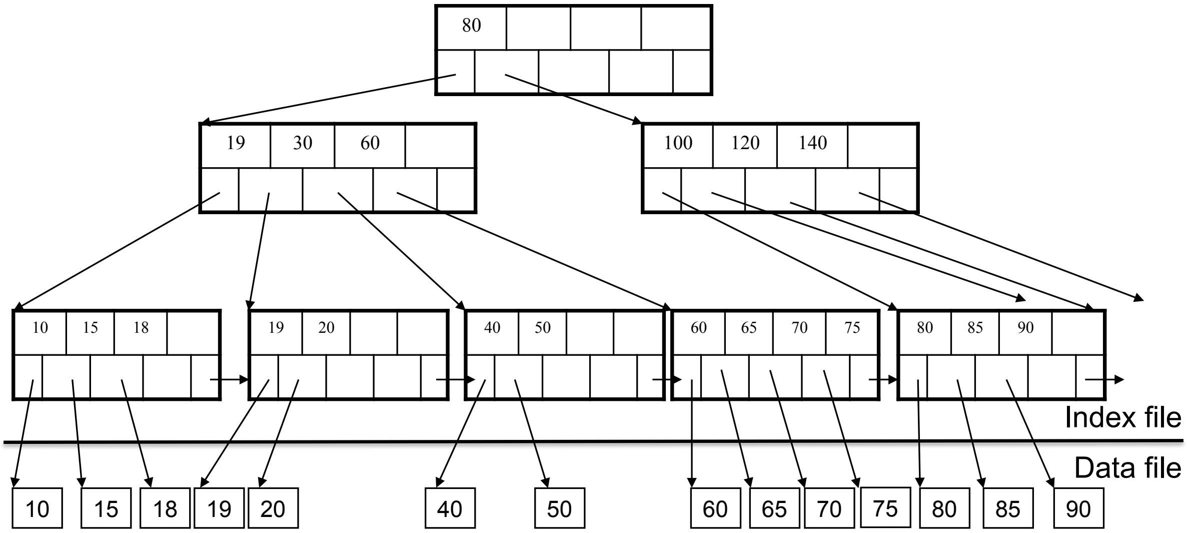
Name: ______________________________

Question	Points	Score
1	15	
2	15	
3	10	
Total:	40	

1 B+ Trees

1. (15 points)

   Consider the following B+ tree index:



   (a) (5 points) Draw the modified tree after the insertion of tuples with search key values 25, 26, and 27 into the base relation and index. Draw the final tree only.

(b) (5 points) Consider the *original* tree again. Draw the modified tree after the deletion of tuples with search key values 10 and 15. Draw the final tree only.

(c) (5 points) Consider the original tree again. How many pages will be read from disk to answer each of the following queries. Assume that the buffer pool is empty before each query executes. Assume also that 4 tuples from the base relation fit on one page in the data file and that all pages are full:

- Look up all tuples with search key value of 40 if the index is *clustered*.
- Look up all tuples with search key value of 40 if the index is *unclustered*.
- Look up all tuples with search key value in the range $[60, 90]$ if the index is *clustered*.
- Look up all tuples with search key value in the range $[60, 90]$ if the index is *unclustered*. Assume the worst-case scenario in terms of buffer pool page evictions.

2 Operator Algorithms

2. (15 points)

   Consider two relations R and S with the following sizes:

Relation	# Blocks	# Tuples
R	100	1000
S	80	800

   (a) (5 points) Compute the cost of joining these two relations in each of the following two scenarios:

   - Using a tuple-at-a-time nested loop join algorithm with R as the outer relation.
   - Using a page-at-a-time nested loop join algorithm with R as the outer relation.

(b) (5 points) Assume the join operator is allowed to use $M = 10$ pages of memory. What is the least-cost method that we could use to join the two relations with a nested-loop type of join algorithm? Describe the method for the R and S relations and compute its cost.

(c) (5 points) Now consider that the join is a primary-key to foreign-key join and that S is indexed on the join attribute but R is not. Describe how R and S can be joined using an index-based nested loop join algorithm. Assume that all index pages are in memory. Indicate the cost assuming each tuple of R joins with exactly one tuple in S.

3 Multi-Pass Algorithms

3. (10 points)

   Consider again the relations R and S from the previous question but this time **assume they have 10 times as many pages. That is, assume** $B(R) = 1000$ **and** $B(S) = 800$. Explain how a DBMS could efficiently join these two relations given that only **11 pages** can fit in main memory at a time. Your explanation should be detailed: specify how many pages are allocated in memory and what they are used for; specify what exactly is written to disk and when. Compute the cost of the join operation.

   (a) (5 points) Present a solution that uses a hash-based algorithm.

(b) (5 points) Present a solution that uses a sort-merge-based algorithm.