READ THIS NOW!

- Print your name in the space provided below.
- Unless a question involves determining whether given C++ code is syntactically correct, assume that it is. Unless a question specifically deals with compiler #include directives, you should assume the necessary header files have been included.
- There are 7 short-answer questions, priced as marked. The maximum score is 100.
- When you have finished, sign the pledge at the bottom of this page and turn in the test and your fact sheet.
- Aside from the allowed one-page fact sheet, this is a closed-book, closed-notes examination.
- No laptops, calculators, cell phones or other electronic devices may be used during this examination.
- You may not discuss this examination with any student who has not taken it.
- Failure to adhere to any of these restrictions is an Honor Code violation.

Name (Last, First) ______________________________
printed

Pledge: On my honor, I have neither given nor received unauthorized aid on this examination.

____
signed

1. Assume that f(n), g(n) and h(n) are positive-valued functions.

   a) [6 points] What does the fact given below imply regarding any big-O, big-Ω and/or big-Θ relationships between the functions? Be complete. No justifications are needed.

   $\forall n \geq 50, f(n) \leq 8g(n)$

   Applying the definition, this implies that f is $O(g)$**.**

   But if you rearrange by dividing through by 8, you also see that g is $\Omega(f)$**.**

   b) [6 points] What does the fact given below imply regarding any big-O, big-Ω and/or big-Θ relationships between the functions? Be complete. No justifications are needed.

   $\forall n \leq 100000, g(n) \leq 17h(n)$

   In this case, the inequality on n points in the wrong direction. This implies no relevant relationships.

   c) [6 points] What does the fact given below imply regarding any big-O, big-Ω and/or big-Θ relationships between the functions? Be complete. No justifications are needed.

   $\forall n \geq 1, 2g(n) \leq f(n) \leq 23g(n)$

   By the definitions, the first inequality implies that f is $\Omega(g)$ **and the second inequality implies that** f is $O(g)$**. So, therefore,** f is $\Theta(g)$**.**

2. Recall the two types of locality, *spatial* and *temporal*, that were discussed in class. Briefly explain whether using a buffer pool be expected to improve performance in the following situations:

a) [5 points] In an application that uses a linked list to store a collection of records that are searched repeatedly

There would be no significant advantage; the buffer pool can only significantly decrease the cost of retrieving records from disk.

There might be a slight gain however; if the searches exhibit some temporal locality, using a cache to store recently-accessed records could make repeated searches for those records cheaper, since searching the buffer pool would probably be faster than a linear search of the entire linked list.

b) [5 points] In an application that reads sequentially through a data file to build a linked list of records

There would be no advantage from temporal locality; no record would be accessed more than once.

Since a sequential pass through the file would exhibit spatial locality (in a very simple way), there would be some merit in retrieving a block of records at once from disk. However, this doesn't need, or benefit from the use of a buffer pool, since no block would be needed more than once.

c) [5 points] In an application that uses a file to store a collection of records that are searched repeatedly

In this case, there is the potential that the buffer pool will lead to improved performance, depending on whether the record searches exhibit enough temporal locality.

3. Suppose that a buffer pool implementation uses a fixed number of slots, organized by some simple linear data structure.

   a) [7 points] Choose a specific physical data structure and describe why it be a good choice if the replacement policy is FIFO.

      The records are removed in the order they are inserted, so this is just a queue.

      Either an array or a singly-linked list would be appropriate.

      Since the number of slots is fixed, the linked list would use slightly more memory, but that may be offset by the fact that the array would require extra arithmetic for the circular indexing that would be needed.

   b) [7 points] Repeat the previous question if the replacement policy is least recently used (LRU).

      We want to be able to find the least-recently used record quickly and to remove it quickly, and we want to b e able to add a new record quickly.

      The first goal implies we'd like to store the records in a way that keeps the LRU record at a fixed spot; that can be accomplished by using a linear list if we adopt the rule that when a record takes a hit we move it to the front or top of the list. Then, the LRU record is always at the tail or bottom of the list.

      New records would then just be pushed onto the front or top of the list.

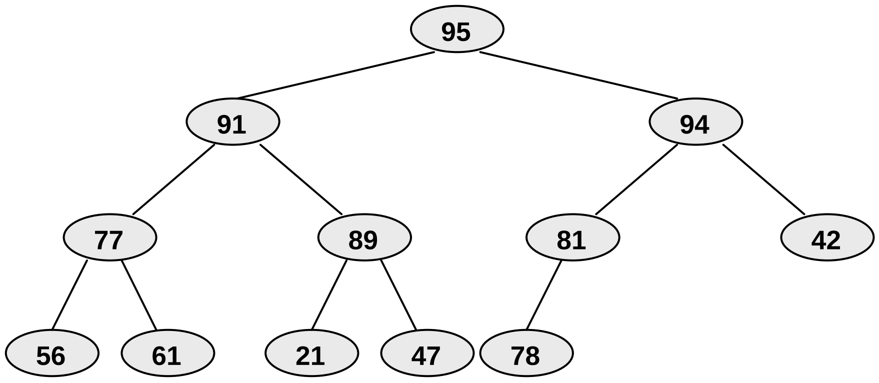
4. [12 points] Indicate whether each of the following statements is true or false; no justification is required

a) $f(n) = 5n^2 + n\log n + 200$ is $\Theta(n\log n)$ TRUE [✓ FALSE]

b) $f(n) = 5n^2 + n\log n + 200$ is $O(n^3)$ [✓ TRUE] FALSE

c) $f(n) = 17n\log n + 200n$ is $\Theta(n\log n)$ [✓ TRUE] FALSE

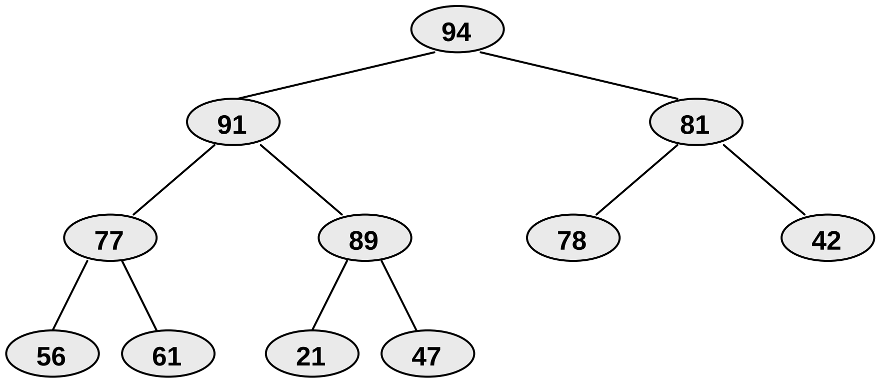
5. Consider the max-heap given below.



a) [8 points] Of course, the heap would be stored in an array. Draw that array.

95	91	94	77	89	81	42	56	61	21	47	78

b) [9 points] Show the effect of deleting the value **95** from the given heap. You may give your answer as an array or draw it as a binary tree.



94	91	81	77	89	78	42	56	61	21	47

6. Consider the implementation of an AVL tree.

   a) [7 points] In big-Θ terms, what is the worst-case cost of performing a rotation? Explain why. (They all have the same cost, in this sense.)

   Even for a double-rotation, the total work required only involves resetting a smallish number of pointers (say 5 or so) and resetting a smallish number of balance factors (say 3 or so). So, the worst-case cost is clearly $\Theta(1)$.

   b) [7 points] In big-Θ terms, what is the worst-case cost of searching for an element? Explain why.

   The worst-case search cost is determined by the number of levels in the tree, since that determines the maximum number of element comparisons that must be performed on a search.

   The height of a AVL tree with N nodes is no more than 1.44 log N, so the worst-case search cost is clearly $\Theta(\log N)$.

7. [10 points] An engineer is designing a new hard drive, using an older drive as the starting point for his design. <u>What</u> aspect of the older drives design should she change in order to improve the average latency, and exactly <u>how</u> should that aspect be changed, and <u>why</u> would this change have the desired effect?

   The latency is determined by the formula:

$$\frac{\theta}{2\pi} \times \frac{1000}{R}$$

   The only aspect of this that the drive designer can change is the rotational speed R, and clearly making R larger will reduce the second factor and hence the overall latency.

   So… make the platters spin faster.