READ THIS NOW!

- Print your name in the space provided below.
- There are 7 short-answer questions, priced as marked. The maximum score is 100.
- This examination is closed book and closed notes, aside from the permitted one-page formula sheet. No calculators or other computing devices may be used. The use of any such device will be interpreted as an indication that you are finished with the test and your test form will be collected immediately.
- Until solutions are posted, 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.
- When you have finished, sign the pledge at the bottom of this page and turn in the test and your fact sheet.

Name (Last, First) Solution

printed

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

_____

signed

1. [16 points] Determine the exact count complexity function T(N) of the following algorithm. Your answer for T(N) should be in simplest form. Show supporting work!

```
for (i = 1; i <= 2*N; i++) {       // Line  1   1 before, 3 per pass, 2 to exit
   x = 0;                          // Line  2   1
   y = N;                          // Line  3   1
   z = 0;                          // Line  4   1
   for (j = 1; j <= i; j++) {      // Line  5   1 before, 2 per pass, 1 to exit
      if ( (i - j) % 2  == 0) {    // Line  6   3
         x = y + j;                // Line  7   2
         y--;                      // Line  8   1
      }
      else {
         x = 2*y - j;              // Line  9   3
         y++;                      // Line 10   1
      }
      z = x * y - j;               // Line 11   3
   }
}
```

$$T(N) = 1 + \sum_{i=1}^{2N} \left(2+1+1+1+1+ \sum_{j=1}^{i} \left(1+3+\max(3,4)+3+1\right)+1+1 \right) + 2$$

$$= \sum_{i=1}^{2N} \left(\sum_{j=1}^{i} (12) + 8 \right) + 3$$

$$= \sum_{i=1}^{2N} (12i + 8) + 3$$

$$= 12\frac{2N(2N+1)}{2} + 8 \cdot 2N + 3$$

$$= 24N^2 + 28N + 3$$

2. [16 points] Assume that $f(n)$, $g(n)$ and $h(n)$ are positive-valued functions defined on the set of non-negative integers.

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

$$\forall n \geq 10,\ 3\mathrm{g(n)} < f(n) \leq 7g(n)$$

Straight from the definition, this implies that $f(n)$ is $\Theta(g(n))$. And, of course, that means that we have every possible big-O and big-Ω relationship between the two functions.

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

$$\forall n \geq 1,\ 7g(n) \geq f(n)$$

Again, straight from the definition, this implies that $f(n)$ is $\mathrm{O}(g(n))$. And, of course, that means that it's also true that $g(n)$ is $\Omega(g(n))$.

3. [12 points] Executing an algorithm on an input of size 1000 requires 10 seconds on certain hardware. The algorithm is known to be $\Theta\left(n^2\right)$. How long would it take to execute the same algorithm on the same hardware on an input of size 4000? Justify your conclusion carefully.

The key here is what happens to the number of operations that must be performed, since the hardware will execute operations at the same rate in both cases. Now, if we let *T(N)* be the complexity function for this algorithm, we know that there are constants such that (from some point on, at least): $AN^2 \leq T(N) \leq BN^2$, so for some constant it's reasonable to expect that $T(N) \approx CN^2$ (ignoring smaller terms).

The relevant fact then is that:

$$\frac{T(4000)}{T(1000)} \approx \frac{C \cdot 4000^2}{C \cdot 1000^2} = \frac{16 \cdot 1000^2}{1000^2} = 16$$

So, we must execute about 16 times as many instructions, and that will take about 16 times as long.

Therefore, we should expect it will take about 160 seconds to execute the algorithm on the larger input set.

4. [12 points] Executing an algorithm on an input of size 1000 requires 10 seconds on certain hardware. The algorithm is known to be $\Theta(n \log n)$. How long would it take to execute the same algorithm on the same hardware on an input of size 4000? Justify your conclusion carefully.

This follows the same logic as the previous question, with a different complexity function. This time, the relevant fact is that:

$$\frac{T(4000)}{T(1000)} \approx \frac{C \cdot 4000 \log 4000}{C \cdot 1000 \log 1000} = 4 \frac{\log 4000}{\log 1000} = 4 \frac{\log 4 + \log 1000}{\log 1000}$$

$$\approx 4 \frac{2+10}{10} \approx 4.8$$

So, it will be necessary to execute about 4.8 times as many operations, which will take about 4.8 times as long. Therefore, we should expect it will take about 48 seconds.

5. [12 points] Let α be a positive constant. Given that $\lim_{n \to \infty} n^{\alpha} = \infty$, what is the relationship between $\log n$ and n^{α}? Justify your conclusion rigorously.

We can determine the relationship by working out the limit of the ratio of the two functions:

$$\lim_{n \to \infty} \frac{\log n}{n^{\alpha}} = \lim_{n \to \infty} \frac{1/(n \ln 2)}{\alpha n^{\alpha - 1}} = \frac{1}{\alpha \ln 2} \lim_{n \to \infty} \frac{1}{n^{\alpha}} = \frac{1}{\alpha \ln 2} \cdot 0 = 0$$

So, we see that $\log n$ is strictly $\mathrm{O}(n^{\alpha})$.

6. [16 points] Write an algorithm to perform a one-sided range search in a BST. Base your solution on the BST interface given for Minor Project 2, and assume that the following public method has been added to the interface:

```
//  Returns a vector holding all the elements in the tree that are strictly
//  less than the object upper (according to the compareTo() method for the
//  type T).
public Vector<T> lessThan( T upper) {

    Vector<T> matches = new Vector<T>();
    lessThan(upper, root, matches);
    return matches;
}
```

Complete the implementation by writing the body of the private recursive helper function shown below. The grading will not penalize for errors in Java syntax unless those errors make it impossible to interpret your answer.

```
private void lessThan(T upper, BinaryNode sroot, Vector<T> matches) {

   // If no node, nothing to check:
   if ( sroot == null ) return;

   // If value in current node is less than upper:
   //    we must add that element to matches
   //    we must consider values in the right subtree of this node
   //
   if ( sroot.element.compareTo( upper ) < 0 ) {

      matches.push_back(sroot.element);

      lessThan(upper, sroot.right, matches);
   }

   // In any case, we must consider values in the left subtree of
   // this node:
   lessThan(upper, sroot.left, matches);

}
```

7. [16 points] Consider using a PR quadtree (with bucket size 1) to organize data objects that have integer coordinates in the square region of the xy-plane that is bounded between the points (0, 0) and (1024, 1024).

a) If there are 517 data points, what is the smallest number of levels the quadtree could have? Justify your conclusion.

The key is the number of leaves that can occur in each level of a PR quadtree; since the branching factor is 4, we can have 1 leaf in level 0, or 4 in level 1, or 16 in level 2, or 64 in level 3, or 256 in level 4, or 1024 in level 5. So there must be more than 5 levels (0 through 4) in order to have enough nodes to store 517 data values.

Hence, the tree must contain at least 6 levels.

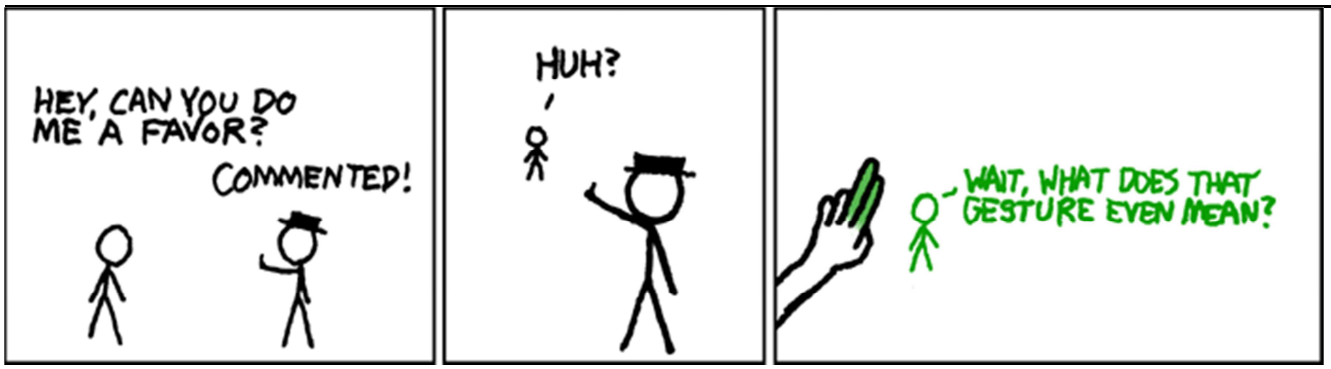
b) Suppose the quadtree contains 893 data points, and that every leaf node is the child of an internal node that represents a region that is 32 x 32.

Suppose that another data object is inserted to the quadtree, and that the insertion of the new data object causes the splitting of a former leaf region. What is the maximum distance the new data object can be from the data object that was already in that leaf region (before the insertion)? Explain your reasoning clearly.

If every leaf is as described, then every leaf represents a 16x16 region. If the insertion of a new object causes a split, it must fall into the same leaf as a previous object.

The maximum distance between two points in a 16x16 square is along the diagonal of the square (which is possible, depending on how objects on the boundaries are handled).

The Pythagorean Theorem tells us the length of the diagonal would be $16\sqrt{2}$.



xkcd.com