Recitation 6 Worksheet 1 and 2 Answers

Complexity

NOTE: I have only included the problems pertinent to the midterm

1. What is big-O for each of the following?
   - $f(x) = 17 + 3x + 21x^2$ $O(x^2)$
   - $g(x) = 19x + x(2x + 2)$ $O(x^2)$
   - $h(x) = x^{\frac{n}{2}} + x^{\frac{n}{3}} + x^{\frac{n}{4}}$ for some $n \geq 1$ $O(x^n)$
   - $j(x) = x^2 log(x)$ $O(x^2 log(x))$

2. Show that $g(x) = 19x + x(2x + 2)$ is not $O(x)$
   g(x) can be simplified into $2x^2 + 21x$
   If $x \geq 0$, then $Cx > 2x^2 + 21x$ - note: we have created a form similar to the "big O" formula
   Solve for x:
   $$C > 2x + 21$$
   $$\frac{C - 21}{2} < x, \; for \; any \; C$$
   Therefore, there is no pair (C, k), such that $Cx > 2x^2 + 21x$ for all $x \geq k$

3. Using $k = 1$, show that $h(x) = x^{\frac{n}{2}} + x^{\frac{n}{3}} + x^{\frac{n}{4}}$ for some $n \geq 1$ is $O(x^n)$ and state a witness C that can be used with $k = 1$ to demonstrate this fact.
   Start by looking at the exponents. We can assert:
   $$\frac{n}{2} \leq n, \; \frac{n}{3} \leq n, \; and \; \frac{n}{4} \leq n \; for \; any \; n \geq 1$$
   Therefore, we can assume:
   $$x^{\frac{n}{2}} \leq x^n, \; x^{\frac{n}{3}} \leq x^n, \; and \; x^{\frac{n}{4}} \leq x^n, \; for \; any \; n \geq 1$$
   Simplify the equation:
   $$3x^n \geq x^{\frac{n}{2}} + x^{\frac{n}{3}} + x^{\frac{n}{4}}$$
   Notice that we have simplified the equation to look like the big O formula.
   if C = 3, k = 1, we can conclude that $Cx^n \geq h(x)$ for all $x > k$.
   Therefore, we have also chosen a big C that proves $h(x)$ is $O(x^n)$

4. Given an integer array a[] of size n, What are the worst-case, best-case, and average-case complexity of the following?

```
for (int i=0; i<n; i++) {
     for (int j=0; j<n; j++) {
          if (i!=j && a[i]==a[j]) return a[i];
     }
}
```

Worst-case: conditional statement isn't met until end of the nested for loop

$$O(n^2)$$

Average-case: conditional statement is met middle of the nested for loop

$$This\ would\ be\ a\ complexity\ of\ \left(\frac{n}{2}\right)^2\ which\ simplifies\ to\ O(n^2)$$

Best-case: conditional statement is met at the beginning of the nested for loop

$$O(1)$$

---

5. Show that $f(x) = 17 + 3x + 21x^2$ is $O(x^2)$ using witnesses C and k
   Start by creating an equation that resembles the formula for Big O complexity:

   $$f(x) \geq f(x)$$
   $$21x^2 + 3x + 17 \geq 21x^2 + 3x + 17$$

   Refer to the complexity handout if this step is confusing.
   Now let's manipulating this equation to match the desired formula:

   $$21x^2 + 3x^2 + 17x^2 \geq 21x^2 + 3x + 17$$

   This can be simplified to:
   $$41x^2 \geq 21x^2 + 3x + 17$$

   By adding one more x^2 to the left hand side we can ensure that the left hand side is definitely greater than the right hand side (remember this is our big C, so we are manipulating the equation to find an appropriate big C).

   $$42x^2 \geq 21x^2 + 3x + 17$$

   Therefore, for $C = 42$ and $k = 1$, we have shown $Cx^2 > f(x)$ for all $x \geq k$ and have found witnesses that proves $f(x)$ is $O(x^2)$.

6. What is the complexity of a program to sort a list of 4 numbers, why?
   This this is a list where n is a set number, the complexity does not increase.
   $O(4)$ is simplified to $O(1)$.

7. Two ways of implementing queues and stacks are with a reference based implementation and with an array based implementation. For both of these implementations, what is the order of each of the following tasks in the worst case?

   - adding an item to a stack of n items
   - adding an item to a queue of n items

   This was just included to get you to start thinking about what would cause worst case complexity for these data structures when it comes to queues and stacks.
   Worst case will always be $O(n)$ for array based implementations and $O(1)$ for reference based implementations. Make sure you understand how stacks and queues work and think about what you would do if they were implemented as an array, list, or reference.

8. What is the complexity of a program that counts the 1-bits in a bit string by looking at each bit, one at a time? Count both comparisons and arithmetic operations.
   Best, worst, and average case complexity would be the same for this problem. I will show the average and worst case complexity.

   Average Case Complexity:
   Comparison complexity: $O(n)$ - (you have to compare each of the bits)
   Arithmetic operation complexity: $O\left(\frac{n}{2}\right)$ - (only half the bits are 1)
   Overall complexity: $O\left(\frac{3}{2}n\right)$, which simplifies to $O(n)$.

   Worst Case Complexity:
   Comparison complexity: $O(n)$ - (you have to compare each of the bits)
   Arithmetic operation complexity: $O(n)$ - (all the bits are 1)
   Overall complexity: $O(2n)$, which simplifies to $O(n)$.

9. Given an integer array, a[], of size n, and some integer value x, what is the complexity of the following code? Count both comparisons and arithmetic operations

```
int pow = 1;
int val = a[n-1];
for (int i = n-2; i >= 0; i--) {
     pow *= x;
     val += pow * x[i];
}
```

   $O(5n)$ which simplifies to $O(n)$