Homework One, for Fri 10/2

CSE 101

Prepare a PDF file in which your solution to each of the following problems (1–7) begins on a fresh page. Upload the file to Gradescope, using your campus email address as login. The deadline is noon on Friday.

These problems are a review of material from earlier courses that will serve us well in 101:

- Using big-O notation, and also Ω and Θ
- Geometric series
- Simple proofs by induction
- Logarithms and exponents

---

1. *Geometric series.*
   (a) Give a simple upper bound on $1 + 2 + 4 + \cdots + 2^n$. Conclude that this sum is $O(2^n)$.
   (b) Do the same for $1 + \frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{2^n}$ and conclude that the sum is $O(1)$.
   (c) By considering a more general series $S(n) = 1 + c + c^2 + \cdots + c^n$, establish the following very useful rule: in big-O terms, the sum of a geometric series is simply:
   - the first term if the series is strictly decreasing,
   - the last term if the series is strictly increasing,
   - the number of terms if the series is unchanging.

2. *Proving by induction.* We'd like to establish the following formula for the sum of the first n odd numbers:

$$1 + 3 + 5 + \cdots + (2n - 1) \quad = \quad n^2.$$

   A nice way to do this is by induction. Let $S(n)$ be the statement above. An inductive proof would have the following steps:
   - Show that $S(1)$ is true.
   - Show that if $S(1), \ldots, S(k)$ are true, then so is $S(k + 1)$.

   Can you fill in the details?

3. *Practice with big-O and Ω.* For some fixed positive integer c, consider the summation

$$S(n) = 1^c + 2^c + 3^c + \cdots + n^c.$$

   (a) Show that $S(n)$ is $O(n^{c+1})$. *Hint:* There are n terms in the series, and each is at most ...?
   (b) Show that $S(n)$ is $\Omega(n^{c+1})$. *Hint:* Look just at the second half of the series.

4. *Logarithms base two.* Recall the definition of logarithm base two: saying $p = \log_2 m$ is the same as saying $m = 2^p$. In this class, we will typically write log to mean $\log_2$.
   (a) How many bits are needed to write down a positive integer n? Give your answer in big-O notation, as a function of n. This is *the length of* n.
   (b) How many times does the following piece of code print "hello"? Assume n is an integer, and that division rounds down to the nearest integer. Give your answer in big-O form, as a function of n.

```
while n > 1:
   print ''hello''
   n := n/2
```

(c) In the following code, subroutine $A(n)$ takes time $O(n^3)$. What is the overall running time of the loop, in big-O notation as a function of n? Assume that division by two takes linear time.

```
while n > 1:
   A(n)
   n := n/2
```

5. *Logarithms base b.* Now we consider logarithms to base $b > 1$: saying $p = \log_b m$ is the same as saying $m = b^p$. The following transformation rule is helpful whenever switching between different bases:

$$\log_a m \;=\; (\log_a b)(\log_b m).$$

(a) True or false: $\log_2 n$ is $O(\log_3 n)$?

(b) True or false: $2^{\log_2 n}$ is $O(2^{\log_3 n})$?

(c) True or false: $(\log_2 n)^2$ is $O((\log_3 n)^2)$?

6. *Minimal big-O notation.* The following statements are all true:

$$\begin{aligned}
24n^2 + 10n + 20 &= O(24n^2 + 10n + 20) \\
24n^2 + 10n + 20 &= O(24n^2) \\
24n^2 + 10n + 20 &= O(n^{10}) \\
24n^2 + 10n + 20 &= O(n^2)
\end{aligned}$$

However, the last one is the simplest, cleanest, and tightest of them, and we will refer to it as the *minimal* big-O form. Write the following expressions in minimal big-O notation.

(a) $100n^3 + 3^n$.

(b) $200n \log(200n)$.

(c) $100n^2 2^n + 3^n$.

(d) $100n \log n + 20n^3 + \sqrt{n}$.

(e) $1^3 + 2^3 + \cdots + n^3$.

In the examples above, the minimal form is unambiguous, but sometimes it is a matter of judgement. For instance, $5^{\log_2 n} = n^{\log_2 5}$ can both reasonably be considered minimal (although we will tend to prefer the latter since it emphasizes that the function is polynomial).

7. A *d-ary tree* is a rooted tree in which each node has at most d children.

(a) We can number the levels of the tree as $0, 1, \ldots,$ with level 0 consisting just of the root, level 1 consisting of its children, and so on. The largest level is the *depth* of the tree. Give a formula for the maximum possible number of nodes at level j of the tree, in terms of j and d.

(b) Suppose the tree has depth k. Give a formula for the maximum possible number of nodes in the tree, in terms of k and d. You can leave your answer in big-O notation.

(c) Suppose the tree has n nodes. What is the minimum the depth could possibly be, in terms of n and d? You can leave your answer in big-O format.