

Homework One, for Wed 9/28

CSE 101

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

1. *Geometric series.* Problem 0.2 from the book.
2. *Proving by induction.* We'd like to show that

$$2 + 4 + 6 + \dots + 2n = n(n + 1).$$

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 $S(1), \dots, S(k)$ are true, then so is $S(k + 1)$.

Can you fill in the details?

3. *Proving by induction, again.* It isn't always trivial to pick suitable base case(s) for induction. For example, try problem 0.3(a) from the book.
4. *Practice with big- O and Ω .* Consider the summation

$$S(n) = 1^c + 2^c + 3^c + \dots + n^c,$$

where c is some fixed positive integer.

- (a) Show that $S(n)$ is $O(n^{c+1})$. *Hint:* This is a one-liner.
 - (b) Show that $S(n)$ is $\Omega(n^{c+1})$. *Hint:* Look just at the second half of the series.
5. *Logarithms base two.* Recall the equivalence:

$$m = 2^k \Leftrightarrow k = \log_2 m.$$

In this class, we will typically write \log to mean \log_2 .

- (a) Consider the sequence:

$$a_1 = 1, a_{k+1} = 2a_k$$

What is the smallest k for which $a_k \geq n$? You can leave your answer in big- O notation.

- (b) Consider the sequence:

$$a_1 = 2, a_{k+1} = a_k^2$$

What is the smallest k for which $a_k \geq n$? You can leave your answer in big- O notation.

6. *Logarithms base b .* Recall the equivalence:

$$m = b^k \Leftrightarrow k = \log_b m$$

as well as the base transformation rule:

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

- (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 $O((\log_3 n)^2)$?

7. A *d*-ary tree is a rooted tree in which each node has at most *d* children.
- We can number the levels of the tree as $0, 1, \dots$, 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*.
 - 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.
 - 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.
8. Suppose $A(\cdot)$ is a subroutine that takes as input a number in binary, and takes time $O(n^2)$, where *n* is the length (in bits) of the number.

- (a) Consider the following piece of code, which starts with an *n*-bit number *x*.

```

while  $x > 1$ :
    call  $A(x)$ 
     $x = x/2$ 

```

Assume that the division by two takes $O(n)$ time on an *n*-bit number.

- On the first iteration of the inner loop, *x* is *n* bits long. What is the length of *x* during the second iteration? The third iteration?
- How many times is the inner loop executed? Leave your answer in big-*O* form.
- What is the overall running time (in terms of *n*)? Again, use big-*O*.
- Is this a polynomial-time algorithm?

- (b) Now answer the same questions for this alternative piece of code.

```

while  $x > 1$ :
    call  $A(x)$ 
     $x = x - 1$ 

```

Assume that the subtraction operation takes $O(n)$ time on an *n*-bit number.