This homework is just a check to make sure you have some necessary background from
an earlier algorithms class (like CS 330). You need to get each question “mostly right”
before you do HW 1. You can do them multiple times if needed. HW 0 will not be
graded and is not part of the final grade. As such, it is a bit informal.
You will get HW 1 in about a week. This will be longer, more formal and will be graded.

1. (i). Put in increasing order: \[O(n^{\log n}), O(n \log n), O(1.1^n), O(2^{\log n}), O((\log n)^2)\].

(ii). Explain why the first function class in your list is less than the second. Be precise.

(iii). Prove that \((1) n^2 \log n\) is not in \(O(n \log(n)^2)\).
By prove I mean state precisely what the two statements mean and then use the precise
statement in your arguments/proofs that the statements are true.

2. (a) Describe (write) an algorithm which takes an undirected graph and two vertices
v1 and v2 in that graph and finds a path from v1 to v2 if such a path exists.
(b) Show how your algorithm works on a graph with about 7 nodes.
(c). Is your algorithm efficient? (Here you need not answer formally, just say if you
think it is an efficient algorithm or not and say why. No proof needed.)

3. A long distance runner is training for a long race by each day taking a series of runs
each of which is either 1 kilometer or 3 kilometers. So for example one day he may run a
total of 7 kilometers by taking 5 runs of 1,1,3,1,1 kilometers or 1,1,1,3,1 kilometers or
1,3,3 etc.
Let \(\text{run}(k)\) be the number of different ways a runner can run \(k\) kilometers in one day.
(a). Write a recursive algorithm which takes as input a number of kilometers \(k\) and
which calculates \(\text{run}(k)\).
(b). What is the value of \(\text{run}(6)\). (This should agree with what your algorithm says.)
(c). Prove that \(\text{run}(k) < 2^k\), for all positive integers \(k\).