CIT 596 Online **Summer 2020**

Module 1-2

Name: Solutions

1. Rank the following functions in increasing order of asymptotic complexity. So if your order has two functions f and g occurring consecutively, you will be asserting that $f \in O(g)$. However, if in your order there are two successive functions that are Θ of each other, indicate this in your solution as well.

   (a) n
   (b) $n^{1.5}$
   (c) 3
   (d) $\sqrt{n}$
   (e) $n \log n$
   (f) $n \log \log n$
   (g) $n \log^2 n$
   (h) $\ln n$
   (i) $\log n^5$
   (j) $(\log n)^5$
   (k) 15000
   (l) 2^n
   (m) 2^{2n}
   (n) 3^n
   (o) $\log(n!)$
   (p) $\log(2^n)$
   (q) 2^{n+1}

   Solution:

   $[3 = 15000], [\ln n = \log n^5], (\log n)^5, \sqrt{n}, [n = \log(2^n)], n \log \log n, [n \log n = \log(n!)], n \log^2 n, n^{1.5}, [2^n = 2^{n+1}], 3^n, 2^{2n}$

   With the letters:

   [(c) = (k)], [(h) = (i)], (j), (d), [(a) = (p)], (f), [(e) = (o)], (g), (b), [(l) = (q)], (n), (m)

2. Give the asymptotic runtime (big-Θ) of the following code snippets.

   (a)
   ```
   int sum = 0;
   for (int i = 0; i < n; ++i)
       for (int j = 0; j < i; ++j)
           ++sum;
   ```
   (b)
   ```
   int sum = 0;
   for (int i = 1; i <= n; ++i)
       for (int j = i; j <= n; ++j)
           for (int k = i; k <= j; ++k)
               ++sum;
   ```
   (c)
   ```
   int sum = 0;
   for (int i = 2; i <= n; ++i)
       for (int j = 1; j <= n; ++j)
           ++sum;
   ```

   Solution:

   (a) $\Theta(n^2)$
   (b) $\Theta(n^3)$
   (c) $\Theta(n^2)$

3. Use the formal definitions of the big-O and big-Ω to prove the following statements.

   (a) $5n + 7 \in O(n)$
   (b) $5n + 7 \in \Omega(n)$

Solution:

(a) Let $c = 6$ and $n_0 = 10$. Observe that for any $n \geq n_0$,

$$5n + 7 \leq 5n + n = cn$$

(b) Let $c = 1$ and $n_0 = 1$. Observe that for any $n \geq n_0$,

$$5n + 7 \geq 5n \geq cn$$

4. Suppose the input to the maximum subinterval problem is an array A. For an index i, if the best sum among all the subintervals ending in i is negative, prove that the optimum subinterval does not include i.

   Solution:

   Assume towards a contradiction that an optimal subinterval I includes the element at index i. Let S denote the sum of the elements in this subinterval. Let S_L denote the sum of the elements in I whose index is less than or equal to i. Then the (possibly empty) subinterval beginning at $i+1$ and ending where I ends has sum $S - S_L$, which is greater than S since S_L is negative (given). Contradiction that I is optimal.

5. Find the recurrence relation for the number of different ways there are to climb n ($n \geq 0$) stairs, if you can either step up one stair or hop up two at a time. Be sure to state the base case(s) for the recurrence as well. *Note*: a "way" to climb n stairs can be viewed as a (ordered) sequence of 1s (1-stair steps) and 2s (2-stair hops).

   Solution:

$$T(n) = \begin{cases} 1 & n = 0, n = 1 \\ T(n-1) + T(n-2) & n \geq 2 \end{cases}$$

6. Find a simple answer to the series $\sum_{k=1}^{n} (2k - 1)$.

   Solution:

$$\begin{aligned} \sum_{k=1}^{n}(2k-1) &= \left(\sum_{k=1}^{n} 2k\right) - n \\ &= 2\left(\sum_{k=1}^{n} k\right) - n \\ &= 2\left(\frac{n(n+1)}{2}\right) - n \\ &= n^2 \end{aligned}$$

7. Prove that given a positive integer a and a non-negative integer b, the number d that Euclid's algorithm outputs is in fact $\gcd(a, b)$. You may use the version of Euclid's algorithm presented in lecture or the (equivalent) one below:

```
euclid(a, b):
    if b == 0:
        return a
    return euclid(b, a % b)
```

Solution:

We will use the algorithm as presented in the problem. We will also assume without loss of generality that $a \geq b$, because of Euclid's algorithm is given an input with $a < b$, then the remainder r when dividing a by b will be equal to a, and in the recursive call the algorithm will be invoked with $(b, r) = (b, a)$. Thus, if the inputs are in the wrong order, the next recursive call will have them in the right order.

We are going to use strong induction for this question, but we will need to be careful because the inputs to the algorithm are pairs of integers. Given two pairs of integers (a_1, b_1) and (a_2, b_2), let's call (a_1, b_1) 'less than' (a_2, b_2), if $a_1 \leq a_2$ and $b_1 \leq b_2$, but at least one of these inequalities is strict.

When proving that Euclid's algorithm is correct on an input (a, b), we will use the strong inductive hypothesis that it is correct for all pairs 'less than' (a, b). Now for the details of the proof:

Base Case: Consider any input pair (a, b) with $b = 0$. We know a is positive and $\gcd(a, 0) = a$, because a is a factor of a ($a = 1 \times a$) and of 0 ($0 = 0 \times a$), and clearly there is no factor of a that is greater than a. Thus, the algorithm is correct in returning a.

Induction Hypothesis: Assume that for all pairs 'less than' the current pair (a, b), Euclid's algorithm is correct.

Induction Step: We wish to show that Euclid's algorithm returns the correct answer on the current pair (a, b) (with $b \leq a$). First, we showed in lecture that $\gcd(a, b) = \gcd(b, a\%b)$. Let us let $r = a\%b$. If we can show that Euclid's algorithm correctly computes $\gcd(b, a\%b)$, then by the above equality we can conclude that it correctly computes $\gcd(a, b)$. But note that $b \leq a$ and $r < b$. Thus by our definition of less than (b, r) is less than (a, b), and therefore the inductive hypothesis implies that Euclid's algorithm is correct on (b, r). Thus it is correct on (a, b).

8. Show the output that the Extended Euclid's Algorithm produces on the inputs $(180, 105)$. Also show the intermediate steps.

   Solution:

   We compute the sequence of remainders following the algorithm. Let $a = 180$ and $b = 105$.

$$
\begin{aligned}
r_1 &= 180 \bmod 105 = 75 \\
&= a - b \\
r_2 &= 105 \bmod 75 = 30 = (1)(105) + (-1)(75) \\
&= b + (-1)(a - b) \\
&= -a + 2b \\
r_3 &= 75 \bmod 30 = 15 = (1)(75) + (-2)(30) \\
&= (1)(a - b) + (-2)(-a + 2b) \\
&= 3a - 5b \\
r_4 &= 30 \bmod 15 = 0
\end{aligned}
$$

   Thus, $\gcd(180, 105) = 15 = 3(180) - 5(105)$.

9. The Greek philosopher Zeno posed the following paradox involving Achilles, a Greek hero who was a very fast runner. Suppose Achilles is in a 100-meter race with a tortoise, a pretty slow mover. For concreteness let us assume that Achilles runs at 10 meters/sec and the tortoise lumbers along at 1 meter/sec. In order to give the tortoise some chance, they agree that it will start 10 meters ahead of Achilles.

   Zeno's argument goes: By the time Achilles runs the 10 meters to get to the tortoise's initial position, it will have moved 1 meter ahead (remember that it runs at one tenth the speed of Achilles). By the time Achilles runs this 1 meter, it will have moved ahead another .1 meter, and so on, *ad infinitum* so that it appears that Achilles will never catch the tortoise.

But it is easy to do the math and see that the tortoise has to run 9/10 the distance Achilles runs at 1/10 the speed, and hence it will take 9 times as long as Achilles to reach the finish line.So obviously our (or Zeno's) argument in the previous paragraph must be incorrect.

Explain what is wrong with this argument using your understanding of summing geometric series. Specifically, show that the infinite sequence of times it takes Achilles to catch up with the tortoise's previous positions add up to a finite number. Complete the argument showing why Achilles will win this race.

Solution:

The total time Achilles takes to run the entire distance is 10 seconds. Now let us look at the time it takes Achilles to catch up with the tortoise. We divide this time into intervals, where in the first interval, Achilles is catching up on the tortoise's initial lead. In the i^{th} interval Achilles covers the distance that the tortoise has covered in the $(i-1)^{th}$ interval. Then the first interval is 1 second long (this is how long it takes Achilles to run 10 meters), the second interval is .1 seconds long for him to run the 1 meter that the tortoise has gained on him in the previous interval, and so on. The i^{th} interval takes $\frac{1}{10^{i-1}}$ seconds. Note that at the end of this infinite sequence of time intervals, Achilles will have caught up with the tortoise. The sum of the lengths of these intervals is

$$\sum_{i=1}^{\infty} \frac{1}{10^{i-1}} = \frac{1}{1-.1} = \frac{10}{9}$$

which is a little over one second. So Achilles will have caught the tortoise in a little over a second, and since both runners take much longer than that to finish the race, Achilles will win.

Aside on terminology: Although this is traditionally called a paradox, this is not a paradox at all. In logic, a paradox is a statement that can neither be true or false because either assumption leads to a contradiction. The simplest such paradox is the statement, THIS STATEMENT IS FALSE., which you can see can neither be true or false. Zeno's paradox, "that Achilles will never catch the tortoise", is simply false.