CSE 6140 / CX 4140 Assignment 1
due Sep 4, 2020 at 11:59pm on Canvas

Please type your answers (LaTeX highly recommended) and upload a single PDF including all your answers. If you want to draw a graph by hand you can take a picture of your drawing and insert it to your PDF file. Please make sure that your inserted picture can be clearly read. Do not forget to acknowledge your collaborators.

1 Simple Complexity (11 pts)

1. (6 pts) For each pair of functions f and g, write whether f is in $\mathbb{O}(g)$, $\Omega(g)$, or $\Theta(g)$.
   (a) $f = (n + 1000)^4$, $g = n^4 - 3n^3$
   (b) $f = \log_{1000} n$, $g = \log_2 n$
   (c) $f = n^{1000}$, $g = n^2$
   (d) $f = 2^n$, $g = n!$
   (e) $f = n^n$, $g = n!$
   (f) $f = \log n!$, $g = n \log n$ (hint: Stirling's approximation)
2. (5pts) Determine the Big-O time complexity for the algorithm below (**show your analysis**). Also, very briefly explain (in one or two sentences) what the algorithm outputs (note: the % symbol is the modulo operator):

```
   Data: n
1  i = 1;
2  while i ≤ n do
3  |  j = 0;
4  |  k = i;
5  |  while k % 3 == 0 do
6  |  |  k = k/3 ;
7  |  |  j++ ;
8  |  end
9  |  i++ ;
10 |  print i,j;
11 end
```

2 Algorithm design and complexity (15 pt)

The problem consists of finding the lowest floor of a building from which a box would break when dropping it. The building has n floors, numbered from 1 to n, and we have k boxes. There is only one way to know whether dropping a box from a given floor will break it or not. Go to that floor and throw a box from the window of the building. If the box does not break, it can be collected at the bottom of the building and reused.

The goal is to design an algorithm that returns the index of the lowest floor from which dropping a box will break it. The algorithm returns $n+1$ if a box does not break when thrown from the n-th floor. The cost of the algorithm, to be kept minimal, is expressed as the number of boxes that are thrown (note that re-use is allowed).

1. For $k \geq \lceil \log(n) \rceil$, design an algorithm with $O(\log(n))$ boxes thrown.
2. For $k < \lceil \log(n) \rceil$, design an algorithm with $O\left(k + \frac{n}{2^{k-1}}\right)$ boxes thrown.
3. For $k = 2$, design an algorithm with $O(\sqrt{n})$ boxes thrown.

Please explain your algorithms clearly.

3 Greedy - points on a 2D plane (12pt)

You are given n distinct points and one line l on the plane and some constant $r > 0$. Each of the n points is within distance at most r of line l (as measured along the perpendicular). You are to place disks of radius r centered along line l such that every one of the n points lies within at least one disk. Devise a greedy algorithm that runs in $O(n \log n)$ time and uses a minimum number of disks to cover all n points; prove its optimality.

4 Greedy - Why is the pool always so busy anyway? (12 pt)

Georgia Tech is trying to raise money for the Technology Square Research Building and has decided to host the "Tech Swim Run Bike" (TSRB) Triathalon to start off the fundraising campaign!

Usually in a triathalon all athletes will perform the three events in order (swimming, running, then biking) asynchronously, but unfortunately due to some last minute planning, the race committee was only able to secure the use of one lane of the olympic pool in the campus recreation center. This means that there will be a bottleneck during the first portion (the swimming leg) of the race where only one person can swim at a time. The race committee needs to decide on an ordering of athletes where the first athlete in the order will swim first, then as soon as the first athlete completes the swimming portion the next athlete will start swimming, etc. The race committee, not wanting to wait around for an extremely long time for everyone to finish, wants to come up with a schedule that will minimize the time it takes for everyone to finish the race. Luckily, they have prior knowledge about how fast the athletes will complete each portion of the race, and they have you, an algorithm-ista, to help!

Specifically, for each athlete, i, they have an estimate of how long the athlete will take to complete the swimming portion, s_i, running portion, r_i and biking portion, b_i. A schedule of athletes can be represented as a list of athletes (e.g. $[athlete_7, athlete_4, ...]$) that indicate in which order the athletes will start the race. Using this information, they want you to find an ordering for the athletes to start the race that will minimize the time taken for everyone to finish the race, assuming all athletes perform at their estimated time. More precisely, give an efficient algorithm that produces a schedule of athletes whose completion time is as small as possible and prove that it gives the optimal solution using an exchange argument.

Keep in mind that once an athlete finishes swimming, they can proceed with the running and biking portions of the race, even if other athletes are already running and biking. Also note that all athletes have to swim first (i.e. some athletes won't start off running or biking). For example: if we have a race with 3 athletes scheduled to go in the order $[2, 1, 3]$, and the finishing time for athlete i is represented as a_i, then the finishing

times would be as follows: (with a total finishing time of $\max(a_1, a_2, a_3)$)

$$a_2 = (s_2) + r_2 + b_2$$
$$a_1 = (s_2 + s_1) + r_1 + b_1$$
$$a_3 = (s_2 + s_1 + s_3) + r_3 + b_3$$