Please enter your name and uID below.

Name: Jason Crandall

uID: u0726408

Submission notes

- **(PS 1 specific) This is a warm-up assignment. Full points will be given for answers that have some correct elements and that clearly been given a legitimate effort.**

- *Solutions must be typeset* using one of the template files. For each problem, your answer must fit in the space provided (e.g. not spill onto the next page) *without* space-saving tricks like font/margin/line spacing changes.
- Upload a PDF version of your completed problem set to Gradescope.
- Teaching staff reserve the right to request original source/tex files during the semester, so please retain the original files.
- Please remember that for problem sets, collaboration with other students must be limited to a high-level discussion of solution strategies. If you do collaborate with other students in this way, you must identify the students and describe the nature of the collaboration. You are not allowed to create a group solution, and all work that you hand in must be written in your own words. Do not base your solution on any other written solution, regardless of the source.

1. (Total Induction)

To prove that the equation $1 + \sum_{k=4n}^{6n} k$ is $O(n^2)$, we can begin by re-writing the summation. From summation rules, we know that $\sum_{k=1}^{4n} k = \frac{4n(4n+1)}{2}$ and $\sum_{k=1}^{6n} k = \frac{6n(6n+1)}{2}$, thus we can conclude that $\sum_{k=4n}^{6n} k = \frac{6n(6n+1)}{2} - \frac{4n(4n+1)}{2}$. Reducing this equation down gives us the final form of $10n^2 + 5n$, thereby meaning that $1 + \sum_{k=4n}^{6n} k = 10n^2 + 5n + 1$. We can show that this is the case by means of induction. We start with a base case of $n = 1$. When plugged into the original formula, this gives us $1 + (4 + 5 + 6) = 16$. We verify this with our derived equation giving us $10(1) + 5(1) + 1 = 16$, proving our base case to be true. Given that, let us assume that this is also that a case $n = j$ is true, giving us $1 + \sum_{k=4j}^{6j} k = 10j^2 + 5j + 1$. We can prove that the next step $(j + 1)$ is true by adding the next progression to the j equation, giving us $1 + \sum_{k=4j}^{6j} k = 10j^2 + 5j + 1 + 20j + 15 = 10j^2 + 25j + 16$. Now we prove (j+1) by solving the following equation and ensure that they are equal: $1 + \sum_{k=4(j+1)}^{6(j+1)} k = 10(j + 1)^2 + 5(j + 1) + 1 = 10j^2 + 20j + 10 + 5j + 5 + 1 = 10j^2 + 25j + 16$. As the equations are equal, we see that this is the polynomial we are looking for. Now, to show that it is in fact $O(n^2)$, we must show that there exists a $c, k\ for\ 10n^2 + 5n + 1 \le cn^2\ for\ all\ n > k$. Here we must find a constant c that proves this, and we modify the right hand side of the equation to give us this value as follows: $10n^2 + 5n + 1 \le 10n^2 + n^2 + n^2$ as this guarantees the right hand side to be greater than or equal to the left. Reducing the right hand side further gives us $10n^2 + 5n + 1 \le 12n^2$, thus our $c = 12$, and as this number does exist, we have shown that $1 + \sum_{k=4n}^{6n} k$ is $O(n^2)$.

2. (CattleSorting)

In order to properly sort these cows using only equalities, we start by marking the first cow in the first stall we see as the type we search for first. Next we have someone move to the next cow and check if that cow equals the first. If it does, continue to the next, otherwise take note of this cow's position. Move to the next cow and see if it is equivalent to the first cow, if it is, swap it with the position you previously marked, otherwise, continue on until you find one that is equal to the first cow. Repeat this process until either of these conditions are met: if you've made it to the last stall without marking a cow to be swapped, you are done. If you have marked a cow and are searching for a swap but make it to the end of the stalls, then you have finished with that cow type, and can start with the next cow. Treat your marked cow as the next cow type and repeat the algorithm above. This is performed in O(nk) time as you will never have to pass through the array more than there are types of cows, as you sort the array as you go. If there is only 1 breed, then you only go through the array once, if there are 2, then you go through at most 2 times, etc.

3. (Vacations)

To prove that no remote employee is given a vacation before an in person employee, let us assume that a remote employee is currently ahead of an in person employee in the priority array. We must find a case where this is broken up in the algorithm so that the remote employee is no longer ahead of the in person employee. As the algorithm progresses through the entirety of the priority array, looking at line 4 of the algorithm, we see that if the current person is remote, and the person afterwards is in person, then these two must swap places. In our example of a remote employee being ahead of an in person employee, the logic in line 4 forces these 2 to swap places, meaning our example cannot hold. As our example of a remote employee being in front of an in person employee does not hold, the algorithm shows that it does give in person employees priority in the array.