Layla Oesper, Jeff Ondich

There is no check point for this problem set. Your full answers to all questions are due by **5:00PM Friday, May 10**. Each solution will be graded for correctness and clarity (4 points per question). Read the course information page/syllabus for further info about this 4-point scale.

0. Estimate the amount of time you spent on each question and include it at the top of your solution. Also, list your collaborators for each question at the top of your solution.

1. For each of the following recurrences, determine the asymptotic running time by either applying the Master Theorem or making a conjecture on a closed form and proving it correct using induction.

   (a) $T(n) = 4T(n/3) + n^2$, $T(1) = 1$

   (b) $T(n) = 4T(n/4) + n$, $T(1) = 1$

   (c) $T(n) = T(n-1) + 2n - 1$, $T(1) = 1$

   (d) $T(n) = 4T(n/2) + 1$, $T(1) = 1$

2. (a) Write a recurrence relation $T(n)$ that has an exact running time of $\Theta(n^{\log_2 3})$ and make an argument about why it has this running time.

   (b) Write a recurrence relation $T(n)$ that has an exact running time of $\Theta(n^2 \log n)$ and make an argument about why it has this running time.

   (c) Write a recurrence relation $T(n)$ that describes the number of basic operations done by the function `foo` that takes in an array of size n. Make sure you also include any base cases for your recurrence.

```
foo(A[1, ..., n]) // a function that takes as input an array A of length n
if n == 0:
    return 0
else if A[1] < 0 :
    return 1 + foo(A[2, ..., n])
else:
    return foo(A[2, ..., n])
```

   (d) Write a recurrence relation $T(n)$ that describes the number of basic operations done by the function `bar` that takes in an array of size n. Make sure you also include any base cases for your recurrence.

```
bar(A[1, ..., n]) // a function that takes as input an array A of length n
if n == 0 or (n == 1 and A[1] >= 0)
    return 0
else if n == 1 and A[1] < 0 :
    return  1
else:
    count = 0
    count = count + bar(A[1, ..., n/2])
    count = count + bar(A[n/2 + 1, .., n])
return count
```