1 Written Part (30 points)

1. R-4.8, p. 182, provide explanation
2. R-4.13, p. 182, provide explanation
3. R-4.16, p. 182
4. Consider $f(n) = 4n^2 + 3n - 1$, mathematically show that $f(n)$ is $O(n^2)$, $\Omega(n^2)$, and $\Theta(n^2)$.
5. For finding an item in a sorted array, consider “tertiary search,” which is similar to binary search. It compares array elements at two locations and eliminates 2/3 of the array. To analyze the number of comparisons, the recurrence equations are $T(n) = 2 + T(n/3)$, $T(2) = 2$, and $T(1) = 1$, where $n$ is the size of the array. Explain why the equations characterize “tertiary search” and solve for $T(n)$.
6. To analyze the time complexity of the “brute-force” algorithm in the programming part of this assignment, we would like to count the number of all possible routes.
   (a) Explain the number of all possible routes in terms of $n$ (number of locations).
   (b) Consider a computer that can process 1 billion routes per second and $n$ is 100, explain the number of years needed to process all possible routes.
   (c) If we don’t want the computer to spend more than 1 minute, explain the largest $n$ the computer can process.

2 Programming Part (70 points)

An online retailer has robots that collect items from different locations in a warehouse to fulfill customer orders. Given a set of items and their locations, a robot starts from a packing station, collects all the items, and returns to the packing station quickly. How would you design an algorithm that finds the shortest route?

The goal of the assignment is to find the shortest route for the robot. Design a “brute-force” algorithm that recursively enumerates all possible routes and determines the shortest route. For simplicity, we use non-negative integer x,y coordinates for the locations and assume a robot can travel from one location to another via a straight line (i.e., no stationary or moving obstacles). Sample input and output are on the course website.

**Input:** The command-line argument for HW2.java is the name of the input file, which has:
1. the number of locations on the first line
2. a location name, x coordinate, and y coordinate on each of the remaining lines

**Output:** The program prints the shortest route (from a packing station and back to the packing station) to the standard output (screen). The location name, x, y, and distance from the previous location is on a line and the total distance is on the second last line. All distance values have 2 decimal places. The first and last locations are the packing station, and the first location has a distance value of 0.00 from the previous location. The amount of time spent by your entire program in seconds, with 3 decimal places, is on the last line (Code Fragment 4.1, p. 151, finds the number of milliseconds).

**Extra Credit (10 more points):** Separate submission via HW2Extra.java. Solve the problem without recursion (or using a stack to simulate recursion).

3 Submission

Submit HW2.java that has the main method and other program files. Submissions for Individual and GroupHelp have the same guidelines as HW1.

Submit the written part in PDF format to the Submit Server. Handcopy is also acceptable in the lab. GroupHelp submission is not applicable to the written part.

Note the late penalty on the syllabus if you submit after the due date and time as specified at the top.

For extra credit, submit HW2Extra.java that has the main method and other program files. GroupHelp submission is not applicable to extra credit. Late submission for extra credit is not accepted.