Mary Alverson
CMSI 402
Assignment 2

Problem 5.1
What's the difference between a component-based architecture and a service-oriented architecture?

A component-based architecture is a system made up of separate components. Each component does something for another component. A service-oriented architecture also is made up of separate pieces, but each piece is more separate and is implemented as a service. This allows each piece to run in different locations on different computers and communicate through a network.

Problem 5.2
Suppose you're building a phone application that lets you play tic-tac-toe against a simple computer opponent. It will display high scores stored on the phone, not in an external database. Which architectures would be most appropriate and why?

Since the application is simple and doesn't store the scores in an external database, there isn't really a need for anything other than a monolithic architecture. This application doesn't need to make any API calls or communicate with other servers. The application should be data-centric/rule based since the amount of moves in tic tac toe is not large, so the computer doesn't need complicated algorithms in order to play. The UI should be event-driven because things should only happen after a player takes a turn.

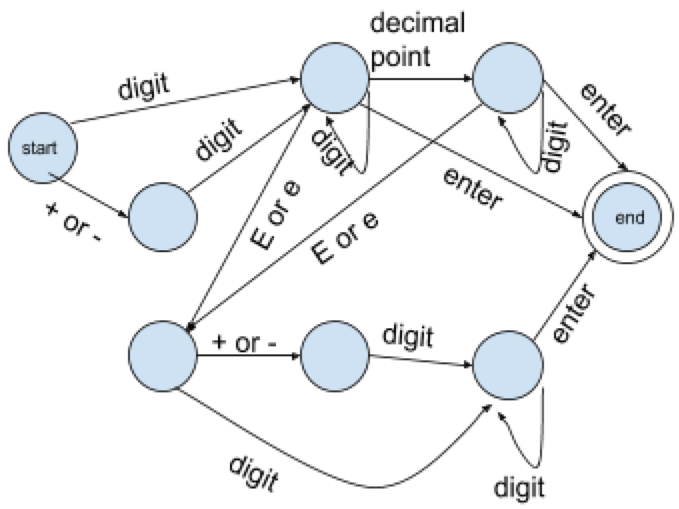
Problem 5.4
Repeat question 3 [after thinking about it; it repeats question 2 for a chess game] assuming the chess program lets two users play against each other over an Internet connection.

Other than needing to send and receive information over a network, chess is still a simple game (especially if the opponent is a human and not a computer) so a simple monolithic architecture should be fine. An external database still wouldn't be necessary. The application should also still be rule-based/data-centric because rules will allows the app to detect when someone wins, for example.

Problem 5.6
What kind of database structure and maintenance should the ClassyDraw application use?

ClassyDraw needs some way to store drawings, but is able to to store them each as a file on the operating system. The files don't need to be shared with other users so they can be stored locally instead of in an external database. This also is a good option because it allows the user to copy, move, and delete the drawings as they please. As far as maintenance goes, if the ClassyDraw application crashes, the drawing should be kept in a temporary file that can be used when the application is re-launched so that the user doesn't lose any of their work.

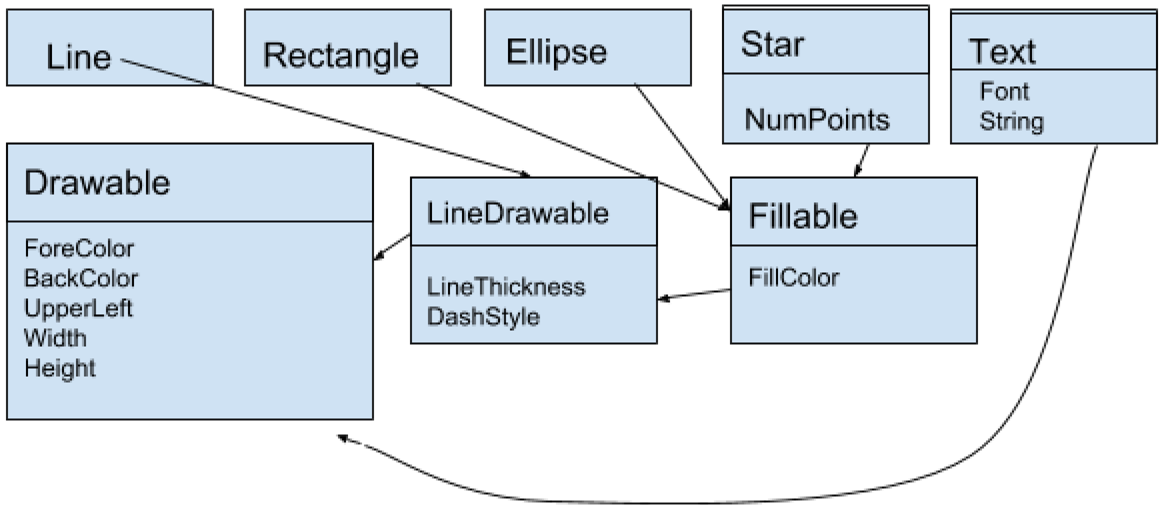
Problem 5.8
Draw a state machine diagram to let a program read floating point numbers in scientific notation as in +37 or -12.3e+17 (which means -12.3 x 1017). Allow both E and e for the exponent symbol. [Jeez, is this like Dr. Dorin's DFAs, or *what*???]



Problem 6.1
Consider the ClassyDraw classes Line, Rectangle, Ellipse, Star, and Text. What properties do these classes all share? What properties do they not share? Are there any properties shared by some classes and not others? Where should the shared and nonshared properties be implemented?

They all share the properties that are needed to draw anything, such as foreground color, background color, position (upper-left corner position), width and height. The classes have some specific needs, however, such as text. Text needs information such as which font and of what string. Star needs how many points to make the star have. Stars can also be filled, but so can the ellipse and the rectangle. All of those shapes can also be drawn with just lines, so line styles are shared by them, as well as by the Line class.

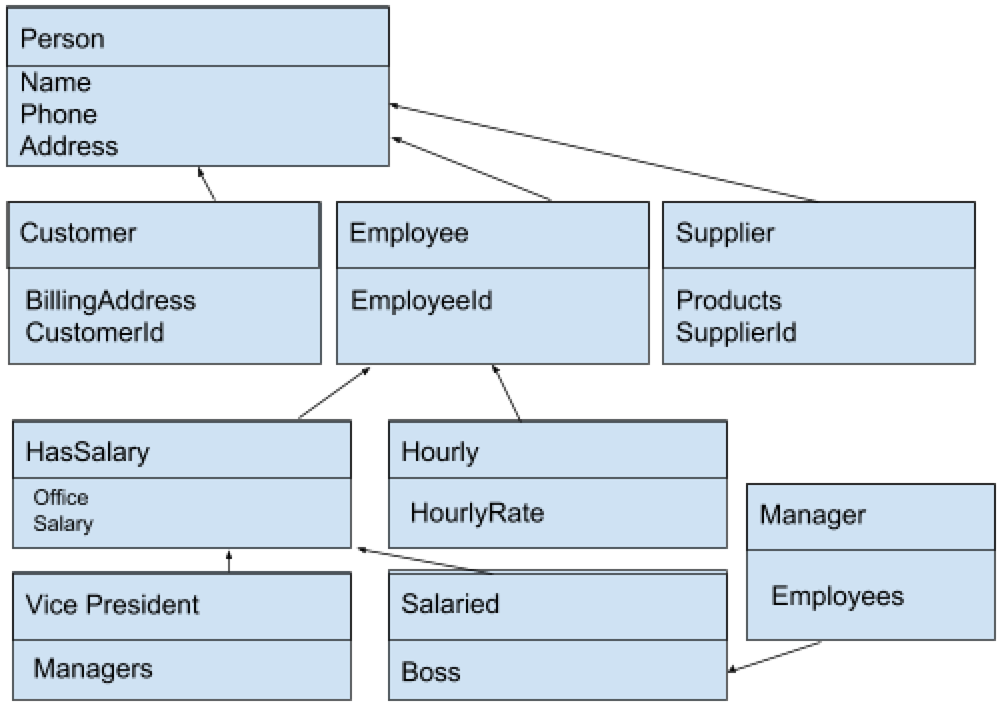
Problem 6.2
Draw an inheritance diagram showing the properties you identified for Exercise 1. (Create parent classes as needed, and don't forget the Drawable class at the top.)



Problem 6.3
The following list gives the properties of several business-oriented classes.

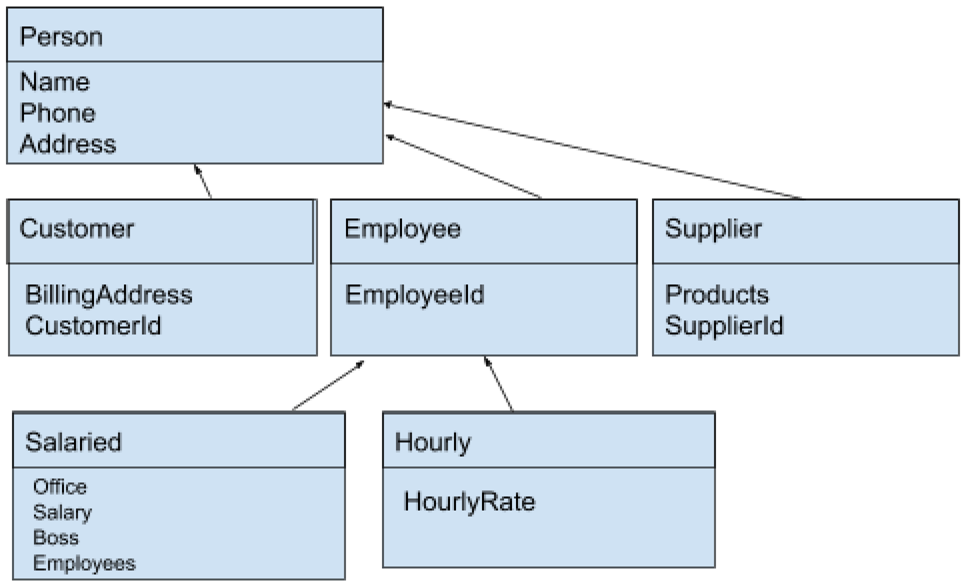
- Customer — Name, Phone, Address, BillingAddress, CustomerID
- Hourly — Name, Phone, Address, EmployeeID, HourlyRate
- Manager — Name, Phone, Address, EmployeeID, Office, Salary, Boss, Employees
- Salaried — Name, Phone, Address, EmployeeID, Office, Salary, Boss
- Supplier — Name, Phone, Address, Products, SupplierID
- VicePresident — Name, Phone, Address, EmployeeID, Office, Salary, Managers

Assuming a Supplier is someone who supplies products for your business, draw an inheritance diagram showing the relationships among these classes. (Hint: Add extra classes if necessary.)



Problem 6.6
Suppose your company has many managerial types such as department namager, project manager, and division manager. You also have multiple levels of vice president, some of whom reprt to other manager types. How could you combine the Salaried, Manager, and VicePresident types you used in Exercise 3? Draw the new inheritance hierarchy.

You could combine Salaried, Manager, and VicePresident types by putting them all in a a Salaried class that has the fields: office, salary, boss, and employees. Vice presidents would just leave the boss category open if they don't have a boss. The lower level employees who are not bosses would just have an empty employees field.


Person
Name
Phone
Address
Customer
BillingAddress
CustomerId
Employee
EmployeeId
Supplier
Products
SupplierId
Salaried
Office
Salary
Boss
Employees
Hourly
HourlyRate