Assignment 1 CS20B (Questions Portion)

Jonathan Reznik

Student ID#1654919

1. *Two examples of the object and class relationship are* a (a) chair object from the Furniture class, and (b) dog object from the class for Animal. *The differences in* terms of methods are that objects make use of instance methods while classes usually have static methods. Objects are generated at runtime and added to a heap in memory, so that objects are referenced and aliased. But a class gets stored in a .java file and provides the organizational view of the code. Also naming conventions differ among the two.
2. *Graph Structures*
   1. Flights
      1. Node: Destination
      2. Edge: Travel time or distance
   2. Borders
      1. Node: Single Country Name,
      2. Edge: Pairs of countries e.g. ‘US/Mexico border’, or ‘South US with North Mexico’.
   3. Research Collection
      1. Node: Title or Author
      2. Edge: Mention of certain words, common references or citations
   4. Actors
      1. Node: Actors names
      2. Edge: Movies in common with other actors
   5. Computer Network
      1. Node: Host
      2. Edge: Connections or Services (e.g. shared files, printing or transfer of data)
   6. Labyrinth
      1. Node: Intersections of paths
      2. Edge: Connections to other intersections, with distance or direction in weighted values
   7. WWW
      1. Node: Hosts
      2. Edge: Ports or Addresses or weighted values of traffic over either one
   8. Social Media
      1. Node: Individuals account
      2. Edge: Number of friends or shares (likes)
3. ArrayList inherits from AbstractList, which inherits from AbstractCollection. The ArrayList class has 3 direct subclasses: AttributeList, RoleList, and RoleUnresolvedList. (Not sure about implement methods, but the implementation Classes are List, RandomAccess, Cloneable, and Serializable). A count of all methods include 3 constructor methods, 31 instance/concrete methods, and another 16 methods inherited from the three inherited classes. And toString() is inherited from AbstractCollection in ArrayList.
4. Complexity in **Big-O**
   1. O(N) or linear complexity is derived from the N + 1 count of operations with 1 variable declaration and N linear processing operations inside a for loop.
   2. O(N2) or time-squared complexity is derived from N2 + 1 operations, with 1 variable declaration and N2 or N \* N from single nested operation inside of the two nested for loops.
   3. O(N) or linear complexity is derived from 2N + 1 operations, with 1 for variable declaration, and 2N or N + N in each subsequent for loop, which makes N + N + 1 or 2N + 1.
   4. O(N2) or time-squared complexity is the same as it was in part (b) despite counting the number in quarter overall as derived by N2/4 +1 count of operations the complexity O(N2).
   5. O(N) or linear time complexity is derived via N + 4 or N + 3 depending on if counting the return statement.
5. Design Problem
   1. Banking – Modeling customer deposit activity with child classes ATM, Online, and Teller for the superclass BankTransaction. Another possible design for Teller process of counting with superclass for Denominations, and subclasses Hundreds, Fifties, Twenty, Ten, Fives and Ones, Quarters, Dimes and Nickels as well. Ok, pennies too!
   2. Gaming – Many models exist for this one, but inheritance can be used to overlay sounds into a virtual environment of the game, with parent class Sound extended to subclasses for in game actions that would trigger sound such as SndNewGame, SndMakeMove, and SndQuitGame. By using inherited members and methods from super class all sounds in the game will have certain behavior like playSound() or locateSound(). *Another example of a game inheritance of classes could be many character types within the Character superclass.*
   3. Travel – ModeOfTransport parent class for the subclasses Plane, Train, Bus. The inherited properties of the superlass can include things like time spent in travel, time of departure, and ETA (from the superclass). *As another example (possible interface class) a Fare class, with subclasses OneWay and Roundtrip.*
6. Complexity – NOTE: I tried to make some explanations that involved more than identifying the highest order of polynomial. Please correct faulty logic wherever possible...
   1. N2+3N = O(N2), because evaluating each term with x2 gives 4x2 which is complexity O(N2).
   2. 3N2+N = O(N2) basically for exact same reason is 4x2 which is O(N2).
   3. N\*(N-1)/2 = O(N2) simplifying mathematically 0.5N2 – 0.5N, looking at the highest order to determine this is of complexity O(N2).