COMP 210: Object-Oriented Programming **May 02, 2018**

Exam 6 (Practice)

Instructions:

- Partial answers will receive partial credit. Make every effort to at least put something for each question. If you don't know the answer, you can get plenty of partial credit by giving a partial answer and explaining where it is wrong or where you got stuck.
- If you believe a problem is incorrectly or incompletely specified, make a reasonable assumption and solve the problem. The assumption should not result in a trivial solution.
- In all cases, clearly state any assumptions you make in your answers.
- Design questions can sometimes have multiple answers. Be sure to give thorough explanations so that I can decide if your answer is a reasonable one I hadn't thought of.

Name	

Question	Description	**Points Possible**	**Grade**
1	Huffman Trees	15	
2	Heaps	15	
3	Heaps	5	
4	Design Principles	20	
5	Design Patterns	30	
Total		85	

1. Create a Huffman tree (using our algorithm from class) for the message

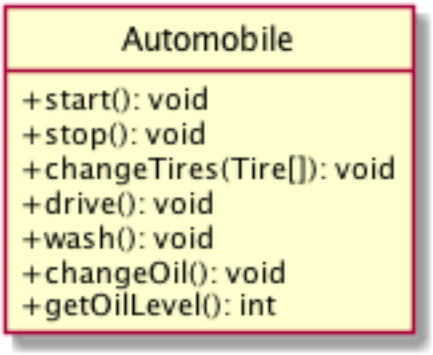
abacadab

2. You are given the following array which represents a min-heap. Assume `deleteMin` is called on it. Show the resulting array.

1	2	17	19	4	42	36	25

3. If I start with an empty min-heap and perform n inserts, what is the *total* worst-case running time (in Big-O)?

4. You just got a job at Tesla working on autonomous cars. Your boss shows you the following class and says it has a problem, but walks away before you can ask about it.



(a) What SOLID design principle is most violated in this code?

(b) How might you fix the problem?

5. Consider the simple class hierarchy concerning pizzas in Listing 1.

```
interface Pizza {
  boolean isBlah();
}

class Crust implements Pizza {
  boolean isBlah() { return true; }
}

class Cheese implements Pizza {
  private Pizza p;
  Cheese(Pizza _p) { p = _p; }
  boolean isBlah() { return p.isBlah(); }
}

class Olive implements Pizza {
  private Pizza p;
  Olive(Pizza _p) { p = _p; }
  boolean isBlah() { return p.isBlah(); }
}

class Anchovy implements Pizza {
  private Pizza p;
  Anchovy(Pizza _p) { p = _p; }
  boolean isBlah() { return false; }
}

class Sausage implements Pizza {
  private Pizza p;
  Sausage(Pizza _p) { p = _p; }
  boolean isBlah() { return false; }
}
```

Listing 1: Simple pizza class hierarchy.

For example, we can make a new anchovy and olive pizza with:

```
Pizza gross = new Anchovy(new Olive(new Cheese(new Crust())));
```

(a) What name makes more sense for `isBlah()`?

(b) The `isBlah()` method is a bit annoying because it is split between all the subclasses of `Pizza`. What design pattern would allow you to put all the `isBlah` code in one class?

(c) Rewrite the code to use the pattern. You will need new class(es) and new method(s).