1. (1point) Write a precondition or requires clause for the method removeDuplicates, so all duplicates from List `lst` are removed.

```
public static void removeDuplicates(List lst) {
   if (lst == null || lst.size() == 0) return;
   List copy = new ArrayList(lst);
   Iterator elements = copy.iterator();
   Object pre = elements.next();
   while(elements.hasNext()) {
      Object nex = elements.next();
      if (pre.equals(nex)) lst.remove(nex);
      else pre = nex;
   }
}
```

2. (1point) Write a return statement in place of the line marked TODO to make `hashCode()` consistent with `equals()`.

```
class Person {
  private String firstName;
  private String lastName;
  ...

  public boolean equals(Object obj) {
      if (!(obj instanceof Person)) return false;
      Person that = (Person) obj;
      return this.lastName.toUpperCase().equals(that.lastName.toUpperCase());
  }

  public int hashCode() {
      // TODO
  }
}
```

3. (2 points) Write a `prettyPrintApple` method that takes a `List` of `Apple`s and that can be parameterized with multiple ways to generate a `String` output from an apple (a bit like multiple customized `toString` methods). For example, you could tell your `pretty-PrintApple` method to print only the weight of each apple. In addition, you could tell your `prettyPrintApple` method to print each apple individually and mention whether it's heavy or light. The solution is similar to the filtering examples we've explored in class. To help you get started, here is a rough skeleton of the `pretty-PrintApple` method:

```
public static void prettyPrintApple(List<Apple>
inventory, ???){
    for(Apple apple: inventory) {
        String output = ???.???(apple);
        System.out.println(output);
    }
}
```

4. (1 point) What is a functional interface? Why is a functional interface required for a lambda expression?

5. (1 point) What is a function descriptor? Give two examples for function descriptors.