6 Malicious Mallory (5 points)

Eve and Mallory are lab partners in CS61BL. Unfortunately for Eve, Mallory doesn't get along with anyone. One day, when Eve isn't looking, Mallory codes the following method and adds calls to it in Eve's code:

```
1  import java.util.ArrayList;
2
3  public void method() {
4      ArrayList a = new ArrayList();
5      String msg1 = "Code not working? Try turning your computer off and on again!";
6      String msg2 = "Is your code running? Then you better go catch it!";
7      String msg3 = "You might have forgotten a semicolon somewhere.";
8      a.add(new IllegalArgumentException(msg1));
9      a.add(new ArrayIndexOutOfBoundsException(msg2));
10     a.add(new NumberFormatException(msg3));
11     int index = (int) (Math.random() * 10);
12     if (index >= 3) return; throw a.get(index);
13 }
```

This code is meant to do nothing 70% of the time, and throw one of three random Exceptions the other 30% of the time. Note: `Math.random()` returns a random `double` in the range $[0, 1)$.

(a) There is an error with this code. Explain what the error is, and whether it is a compile time error, a runtime error, or a logic error.

Solution: `ArrayList` isn't given a generic type, so `a.get(index)` will give an object of static type `Object` when `throw` is expecting an object of static type `Throwable` (or any of its subclasses). This is a compile time error.

(b) Cross out one line of code above. Rewrite this line of code below so that the code compiles, runs, and does what Mallory intended.

Solution:
Cross out line 12:
`if (index >= 3) return; throw (RuntimeException) a.get(index);`

Alternatively, you could have crossed out line 4:
`ArrayList<RuntimeException> a = new ArrayList<RuntimeException>();`

Comments: You have to cast as a `RuntimeException` (unchecked exception) and not just an `Exception`. Throwing a potentially checked exception would also require the method header to say `throws Exception`.