Practice: Study Questions on Material Before Break

This is a collection of review questions on material that we did before break. You don’t have to turn these in. They are just a guide to help you review essential concepts.

1 Interfaces and Traits

1. What does an interface (ala Java) achieve in a program?

2. What do interfaces do that (abstract) classes do not?

3. What is the difference between Java-style interfaces and Scala-style traits?

4. When should you create an interface/a trait in a programming project?

2 Mutable and Immutable Data

5. What does it mean for a piece of data to be immutable? What specifically can and cannot change about the data?

6. Describe two concrete situations (examples) where you would want your data to be mutable (e.g., “account balances in a bank need to be mutable”)

7. Describe two concrete situations (examples) where you would want your data to be immutable.

8. Consider the following piece of code:

```scala
class Contact(val name: String, var number: Int) {...}
val c1 = new Contact(``Kathi'', 4018637607)
var c2 = new Contact(``Brown CS'', 4018637600)
```

9. Which names and fields can have their values changed with this code?

10. Does marking a class field private make it immutable?

3 Lists vs Arrays

11. Make a chart listing the advantages and disadvantages of each of Linked Lists and Arrays.

12. What questions should you ask yourself when deciding whether to use a LinkedList or an Array when choosing data structures for a program?
4 Encapsulation

The term *encapsulation* refers to the principle that “computations should live with their data” (i.e., put computations in the classes that contain the data).

13. What benefit does encapsulated code provide that non-encapsulated code does not?
14. In what way(s) are interfaces part of achieving encapsulation?
15. In what way(s) are access modifiers (public/private/etc) part of achieving encapsulation?

5 HashCodes

16. What is a hashcode?
17. What have we used hashcodes for so far in CS18?
18. In general, what can having a hashcode help you do with a piece of data (think broader than hashmaps to a more general principle)

6 Hashmaps

19. How are Hashmaps different from Arrays?
20. What role do hashcodes play in hashmaps?
21. What does the term “collision” mean in a hashmap?
22. How does a Hashmap implementation guarantee that all the items don’t end up in the same bucket/slot of the hashmap?
23. When should you choose a hashmap as opposed to a list, array, or binary tree when setting up data for a problem?

Please let us know if you find any mistakes, inconsistencies, or confusing language in this or any other CS18 document by filling out the anonymous feedback form: [https://cs.brown.edu/courses/cs018/feedback](https://cs.brown.edu/courses/cs018/feedback)