CSE 8B discussion: week 3

HW2, HW3, tips for success on programming assignments
Agenda

1. HW2 recap
2. How to succeed at programming assignments
3. HW2: my solution
4. HW3 overview and hints
HW2 recap

● Congratulations! That was a hard assignment.

● Tons of new concepts:
  ○ hexadecimal representation of numbers
  ○ bytes
  ○ using InputStreams and OutputStreams
  ○ exceptions (tons of them)
  ○ piecing together few clues to solve a problem

● Logic can get messy
  ○ I saw lots of nested loops, lots of if statements that compare all bytes individually, etc.

● This can be minimized with experience (just takes time), being critical of your own code, and actively seeking out better ways to do things (more on this later).
How to succeed at programming assignments

- **Start early** -- you never know how long/short things like this will take
- Read the assignment **thoroughly**
- Complete the assignment in the order of the directions
  - Often concepts build on each other throughout the assignment
- Working with unfamiliar classes? Read their docs, know what methods are available to you
- Google often (StackOverflow is your friend). ex.: “java check if two arrays are equal”
- Discuss approaches to the problem with your classmates
- Read all the questions on Piazza
- **Start early**
Agenda

1. HW2 recap
2. How to succeed at programming assignments
3. HW2: my solution
4. HW3 overview and hints
HW2: my solution

- [go over solution]
Agenda

1. HW2 recap
2. How to succeed at programming assignments
3. HW2: my solution
4. HW3 overview and hints
HW3 overview and hints

- **Problem:** Scan a text file, keeping track of the words that occur and the number of times that each word occurs. Print the $n$ most frequently-occurring words, as well as the number of times that each occurred.
- Java has a broad set of data structures called **Collections** in which you can store things
  - List: a sequence of elements, accessible by index; allows duplicate elements
  - Map: a set of key-value pairs; keys and values can be any kind of object
  - Set: an unordered collection of elements; no duplicates
  - Queues and Stacks: like lists, but with restrictions on element insertion and retrieval
- Which one(s) are of use to us?
- **Always choose the most appropriate type for the problem!**

https://docs.oracle.com/javase/tutorial/collections/interfaces/index.html
HW3 overview and hints: what data structures?

- It seems like words should be grouped with numbers in some way.
- Maybe a `Map<String, Integer>` mapping words to occurrences?
  - How would you access these to sort them?
So a Map<String, Integer> doesn’t quite work for the whole problem.

What about keying on occurrence count?

- Map<Integer, String> ? What could that represent?
- Map<Integer, Set<String>> ? What could that represent?

Check out `SortedMap` and its methods.

One of the methods, called in a loop, could fetch you the $n$ most frequently-occurring words.
A word on using Collections

- List, Map, SortedMap, etc., are all interfaces
- Instead of instantiating a Map, you actually create an instance of a class that implements the Map interface
  - e.g., Map<String, String> thing = new HashMap<String, String>();
- When looking at the docs for an interface, the area that says “All known implementing classes” tells you what classes you could use to get access to the interface’s methods
- More on interfaces later in the course
If you want to have a fast program, try implementing all the ways of solving this problem you can think of, and seeing how fast they run with the `time` command:

```
$ time java Recover
```

- `real` 0m0.711s
- `user` 0m0.251s
- `sys` 0m0.092s

Using data structures and things incurs a performance hit. Maybe there’s some clever way to do it without expensive Maps and Sets and things?
Agenda

1. HW2 recap
2. How to succeed at programming assignments
3. HW2: my solution
4. HW3 overview and hints
Closing

- Questions?