Mid-term study guide

What to expect

- The final exam will last 160 minutes, online and in two parts. There will be a HackerRank part (for coding questions) which will be *closed-book* and take maybe 50 minutes. The other part will be on Blackboard and will be *open-book.* No communications devices/apps/smoke signals, etc. will be allowed—having one of these will result in a zero on the exam. [Sorry to have to impose all these rules!]
- Expect quite a few multiple choice questions, true-false, and other similar types of question.
- The coding questions on Blackboard will require *pseudo-code* (or real code if you prefer). On HackerRank, actual compiling/running code will be required.
- There will probably be some "Short Answer" questions. DO NOT write an essay! I just need to see that you have mentioned the salient points. Spelling and grammar are not important. Exact syntax (or names of methods) is nice but not absolutely required. If you don't know the name of a method you need—invent one.
- There may be more questions than you have time for. Don't worry, but try to get as many reasonable answers as possible: there will usually be <u>some</u> credit at least for attempting a question.

Topics to expect

- Major points from the first half of the semester are always likely. So, pretty much any of the following mid-term exam topics are possible:
  - Language fundamentals: what are the principal features of Scala;
  - Basic syntax: use of “_”, “=>”, “<-“, etc.;
  - Significance of val, def, lazy; difference between call-by-value and call-by-name; functions vs. methods;
  - For comprehensions, *map* and *flatMap;* syntactic sugar;
  - *Option, Try, Future, List, Stream, Tuple2*;
  - Basic rules of <u>implicits</u>;
  - Functional composition: *map2, lift, curried, compose,* etc.;
  - Recursion (especially tail-recursion).

Additional Topics: Final Exam

- Scala API (collections, etc.):
  - exact names of methods not required, but an understanding of the various kinds of signatures is.
- Implicits, including type-classes and context bounds.
- Simple parsers (from parser-combinator library).
- Spark
  - How RDDs work; transformations and actions; Datasets.
- Testing
- Actors