UCLA Computer Science 131 midterm - Fall 2020

Name:________________________________

Student ID:__________________________

100 minutes total, 1 minute = 1 point. Open book, open notes, open computer. Answer all questions yourself, without assistance from others.

The exam is not easy, and you are not expected to answer all the questions completely. In your answers, overall approach and intuition will count more than trivial detail. Budget your time while taking the exam. It may help to skip questions that are harder than their point count would suggest.

You can print the exam, read the first page (if you haven't read the instructions already, which I sent you via email yesterday), then write your starting time on the first page. Then take at most 100 minutes to answer the questions and write your answers on the exam. (CAE students with x% extra time should add to the 100 minutes accordingly, getting 100+x minutes.) When you're done, write your finishing time on the first page, sign the first page, scan the completed exam, and upload your scans to CCLE Gradescope as quickly as you can. If you lack a scanner, carefully photograph the sheets of paper with your cell phone and upload the photographs. Save your filled-out exam until the class is over, and do not give or show it to anybody other than an instructor or TA.

You can use other technology to take the exam, e.g., by using a tablet to write over the PDF. All we need is a PDF, preferably using the same layout.

You can pick the starting time for the exam. We will give you an extra 20 minutes for the overhead of downloading, printing, and/or scanning. Don't abuse this extra time: limit your own exam-taking time to 100 minutes. You must finish the exam by 24 hours after the exam is made available.

If you lack a printer, read the exam on your laptop's screen, write your answers on blank sheets of paper (preferably 8½″×11″) with one page per question, and upload the scanned sheets of paper. At the end of the exam, you should have scanned and uploaded as many photographs as there are questions. If you do not answer a question, scan a blank sheet of paper as the answer. You can type your answers if you like; all we need is a PDF that you can upload on Gradescope.

You can use your laptop to use a search engine for answers, and to run programs designed to help you answer questions. However, do not use your computer or any other method to communicate with other students or outsiders, or anything like that. Communicate only via CCLE and Gradescope to obtain your exam and upload your scanned results, or via Zoom or Piazza with the instructor or TAs. Do not communicate this exam or your answers to anybody other than the professor or the TAs, even after the exam is over.

IMPORTANT Before submitting the exam, certify that you have read and followed the above rules by signing and dating it.

_________________ Date and time (Los Angeles time) you started the exam

_________________ Date and time that you ended the exam

____Signature

1. Consider the quiz given on the first day of class, in which M.D. McIlroy wrote a shell script that computed a concordance of words found in the input, sorted by popularity of occurrence. You want an OCaml function ‘mcilroy’ that does what McIlroy’s script did, but using OCaml objects instead of POSIX character streams. ‘mcilroy’ should take a sequence of characters as an argument, and return a concordance that contains the same information as the concordance output by McIlroy’s shell command. You may assume the existence of functions ‘sort’, ‘tr’, and ‘uniq’ that have the functionality of their POSIX counterparts, but which are OCaml functions instead of being POSIX utilities.

1a (8 minutes). Give the types of the functions ‘mcilroy’, ‘sort’, ‘tr’, and ‘uniq’. Base the types on ‘char’, not ‘string’; for example, if you want a finite sequence of characters, use a list or a tuple of ‘char’, not ‘string’.

1b (8 minutes). Implement ‘mcilroy’ in terms of the other functions.

2 (6 minutes). Give an example of Java code that is not referentially transparent.

3. Suppose you’ve written a curried function f which takes three arguments, but now you notice that you would rather have had a different curried function rf which would act like f except with its arguments in reverse order. That is, (rf c b a) should act like (f a b c). Worse, you’ve done this systematically; you also have a function g with three arguments where you really wanted them in reverse order, another such function h, and so forth. You can’t change f, g, h, ... because so much other code uses them now and you don’t want to change that code; but you’d like to have reverse functions rf, rg, rh for new code.

3a (6 minutes). Write a higher-order function r3 that reverses such functions. For example, you should be able to implement rf this way:

```
   let rf = r3 f
```

and similarly for rg, rh, etc. Or, if it’s not possible to write ‘r3’, explain why not.

3b (10 minutes). Assume that r3 is possible to write, and similarly for functions like r2 (for 2-ary curried functions), r4 (for 4-ary curried functions), and so forth. Generalize them to a function that takes an integer and an n-ary curried function as an argument. Call your generalized function ‘rn’. ‘rn’ should itself be curried: you pass it an integer n, and it returns a function that will accept any n-ary curried function and return a curried function that wants the arguments in reverse order. That is, you should be able to implement r3 this way:

```
   let r3 = rn 3
```

and similarly for r2, r4, etc.

Specify the type of ‘rn’. Or if it’s not possible to write ‘rn’, explain why not.

4 (9 minutes). In Homework 2, the definition for “alternative list” said, “By convention, an empty alternative list [] is treated as if it were a singleton list [[]] containing the empty symbol string.” Suppose instead you had been assigned a variant Homework 2V in which the definition had said “By convention, an empty alternative list [] means that nothing can be derived from the nonterminal in question; i.e., the nonterminal is a blind alley that can never be used to produce a sentence in the grammar”, and suppose you had written a function “make_parserV” that acted like “make_parser” except it solved Homework 2V instead of Homework 2.

Would it be reasonable to implement make_parser in terms of make_parserV? or make_parserV in terms of make_parser? or both? or neither? Briefly justify your answer.

5. Consider the following grammar for a variant of ISO EBNF, written in ISO EBNF. The start symbol is ‘syntax’, and assume that ‘letter’, ‘character’, and ‘decimal digit’ are defined in the usual way: a letter is any ASCII letter, a decimal digit is any ASCII decimal digit, and a character is any ASCII character.

```
  syntax = syntax rule, {syntax rule};
  syntax rule = meta identifier, '=', definitions list, ';';
  definitions list = single definition, {'|', single definition};
  single definition = primary, {',', primary};
  primary = optional sequence | repeated sequence | special sequence
    | grouped sequence | meta identifier | terminal string | empty;
  empty = ;
  optional sequence = '[', definitions list, ']';
  repeated sequence = '{', definitions list, '}';
  special sequence = '?', {character}, '?';
  grouped sequence = '(', definitions list, ')';
  terminal string = "'", character, {character}, "'"
    | '"', character, {character}, '"';
  meta identifier = letter, {letter | decimal digit};
```

5a (5 minutes). Show that this grammar is ambiguous. Assume the usual way of expanding repetitions like {syntax rule} into plain BNF.

5b (10 minutes). Translate this grammar into syntax diagrams in the style of Webber, circling nonterminals and drawing boxes around terminals. Keep your diagrams simple by eliminating any nonterminal that is used only once (except do not eliminate the start symbol ‘syntax’). You need not diagram ‘letter’, ‘character’, or ‘decimal digit’.

6a (10 minutes). Consider the following Java definitions.

```
   int v = 10;
   boolean flag = false;
   void setem () { v = 20; flag = true; }
   void getem () { if (flag) System.out.println("v = " + v); }
```

Suppose thread 1 executes ‘setem’ at about the time that thread 2 executes ‘getem’, and that these are the only uses of any code that accesses the variables ‘v’ and ‘flag’. Explain whether the output could be "v = 10" under the Java Memory Model.

6b (4 minutes). How would your answer to (a) differ if ‘setem’ and ‘getem’ were declared synchronized instead? Briefly explain.

6c (6 minutes). How would your answer to (a) differ if ‘v’ was declared volatile instead? (Assume ‘synchronized’ is not used.) Briefly explain.

7 (12 minutes). Write a function adjdup that takes a list and returns a list of the same length with the same elements in the same order, except that if there are duplicates in the original list, they are reordered to be just after the first of the duplicates. For example, adjdup [7;6;7;8;8;4;10;4;3;5;1;2;7;7;10;9;8;5] should returns [7;7;7;7;6;8;8;8;4;4;10;10;3;5;5;1;2;9]. Your implementation may use the Stdlib and List modules, but it should use no other modules.

8 (6 minutes). In Java, a class can implement as many interfaces as you like, but it can extend at most one class. Suppose we invented a language Avaj which was just like Java except that in Avaj a class can extend as many classes as you like, but it can implement at most one interface. Explain why Avaj would be more problematic than Java in practice.