$Id: study-guide-2021-q1winter.mm,v 1.39 2021-03-10 10:37:54-08 - - $
PWD: /afs/cats.ucsc.edu/courses/cse112-wm/Syllabus/study-guide-2021-q1winter.d
URL: https://www2.ucsc.edu/courses/cse112-wm/:/Syllabus/study-guide-2021-q1winter.d/

1. Week 1 — January 5

(a) Syllabus, pair programming, course overview.

(b) Lab0 intro unix, and review of Data Structures labs.

(c) Lecture notes : **`scheme-1-language.pdf`** (p. 1–4).

(d) **`Languages/scheme/Examples`** — Simple introductory Scheme programs : **`hello.scm`**, **`argv.scm`**, **`false.scm`**, **`hashbang.scm`**, **`commandline.scm`**, **`factorial.scm`**, **`fibonacci.scm`**, **`divmod.scm`**, **`complex.scm`**.

2. Week 1 — January 7

(a) **`asg1-scheme-mbir`** — Programming project : interpreter for Minibasic written in scheme. Program specifications.

(b) Example **`mbir`** programs to be interpreted : syntax and semantics.

(c) **`misc-cons-lists.d`** — pictures and diagrams of Scheme lists with **`cons`**.

(d) **`code/mbir.scm`** — starter code for the interpreter, begin general dissection and details of the code.

3. Week 2 — January 12

(a) Continued dissection and examination of starter code **`code/mbir.scm`**.

(b) Somewhat more complex Scheme programs, showing data structures and expression evaluation : **`readnums.scm`**, **`simple-eval.scm`**, **`evalexpr.scm`**, **`hashexample.scm`**, **`labelhash.scm`**, **`symbols.scm`**, **`mergesort.scm`**.

4. Week 2 — January 14

(a) Lecture notes : **`scheme-1-language.pdf`** (p. 5–end).

(b) Mathematical derivation of factorial : non tail recursive and tail recursive versions, showing call stack depth.

(c) Scheme tracing on factorial and Fibonacci functions.

5. Week 3 — January 19

(a) Lecture notes : **`scheme-2-higherorder.pdf`** (p. 1–12) — higher order functions in Scheme.

6. Week 3 — January 21

(a) Lecture notes : **`scheme-2-higherorder.pdf`** (p. 12–19) — higher order functions in Scheme.

(b) **`testrun.sh`** — Testing scripts and test data, how the graders will do the grading of submitted programs.

(c) Mention as an example a book that uses Ocaml as a Unix Systems programming language.

(d) Lecture notes : **`ocaml-1-notes.pdf`** (p. 1–9) — introducing Ocaml, another functional lanuage, but one that compiles to native machine code.

7. Week 4 — January 26

(a) Lecture notes : **`ocaml-1-notes.pdf`** (p. 9–19).

(b) Minibasic language specification : Language description.
Sample programs : syntax and semantics.

(c) **`asg2-ocaml-interp`** — Programming project : interpreter for Minibasic written in Ocaml.

(d) Begin dissection of Ocaml starter code, general overview.

8. Week 4 — January 28

(a) Finish dissection of starter code for Minibasic interpreter in Ocaml. **`.mli`** files are interface specifications, and **`.ml`** files are implementations.

(1) **`absyn.mli`** — abstract syntax definitions for the interpreter.
(2) **`dumper.mli`**, **`dumper.ml`** — formatting and printing abstract syntax for debugging.
(3) **`etc.mli`**, **`etc.ml`** — miscellaneous functions.
(4) **`interp.mli`**, **`interp.ml`** — functions performing the interpretation of statements and expressions, and supervisory functions calling these.
(5) **`main.ml`** — main program and options analysis.
(6) **`tables.mli`**, **`tables.ml`** — dispatch tables for labels, arrays, variables, and functions.
(7) **`scanner.mll`**, **`parser.mly`** — complete scanner and parser provided, not student-edited, in lex-like and yacc-like format.
(8) **`Makefile`**.

(b) **`Languages/ocaml/Examples/a-list`** — Simple introductory Scheme programs : **`hello.ml`**, **`helloworld.ml`**, **`argv.ml`**, **`length.ml`**, **`factorial.ml`**, **`fibonacci.ml`**.

(c) **`Languages/ocaml/Examples/b-list`** — Ocaml examples specifically relevant to the programming assignment :
`eval1-simple.ml`, **`eval2-symbols.ml`**, **`hashexample.ml`**, **`readnumber.ml`**.

9. Week 5 — February 2

(a) Lecture notes : **`ocaml-2-higherorder.pdf`** (p. 1–9) — higher order functions in Ocaml.

(b) Frivolous : EWD-714 : E.W.Dijkstra, trip report to Santa Cruz (UCSC), 1979.
EWD-498 : E.W.Dijkstra, How do we tell truths that might hurt ?

(c) **`Languages/ocaml/Examples/c-list`** — Some more advanced Ocaml examples :
`ackermann.ml`, **`complex-nrs.ml`**, **`exponent.ml`**, **`mergesort.ml`**, **`ncat.ml`**, **`odd-even.ml`**, **`qsort.ml`**.

10. Week 5 — February 4

(a) **`Languages/ocaml/Examples/x86-64-code`** — Examples of constant propagation optimization and tail call elimination in code generated for the x86-64 architecture :
`boolconst.s-opt`, **`boolvar.s-opt`**, **`cfacloop.s-opt`**, **`cfacrec.s-opt`**, **`tailrectest.s-opt`**.

(b) Lecture notes : **`object-oriented.pdf`** — Object-oriented programming. Polymorphism : parametric (universal), inclusion (object oriented), overloading (ad hoc), and conversion (ad hoc).

(c) Lecture notes : **`smalltalk-notes.pdf`** — introduction to Smalltalk.

11. Week 6 — February 9

(a) **`asg3-smalltalk-mbst`** — programming project : interpreter for Minibasic written in Smalltalk. Overview of intermediate code files to be processed by the interpreter.

(b) **`asg3/Examples/`** — examples of simple Smalltalk programs showing general ideas.
  (1) **`a-trivial.d`** — some trivial examples :
  `hello.st`, **`echoargs.st`**, **`arithmetic.st`**, **`cmdline.st`**, **`divide.st`**, **`intsort.st`**, **`dictionary.st`**, **`collatz-block.st`**, **`collatz-class.st`**.
  (2) **`b-simple.d`** — some very simple examples :
  `ashex.st`, **`filein.st`**, **`isgraph.st`**, **`perform.st`**, **`priority.st`**, **`string.st`**, **`terminalecho.st`**.

(c) **`code/mbint.st`** — dissection of starter code for interpreter.

12. Week 6 — February 11

(a) **`code/mbint.st`** — continuation of dissection of starter code for interpreter. Most of lecture.

(b) **`c-involved.d`** — a few more involved examples :
`initarray.st`, **`sorted-names.st`**.

13. Week 7 — February 16

(a) Midterm exam. No lecture.

14. Week 7 — February 18

(a) **`asg3/Examples/c-involved.d`** — more Smalltalk examples : **`binepsilon.st`**, **`catfile.st`**, **`complexx.st`**, **`euler.st`**, **`sorted-names.st`**, **`treeleaf.st`**, **`wordcount.st`**.

(b) **`asg3/misc-evalexpr`** — miscellaneous parallel examples comparing features of Scheme, Ocaml, and Smalltalk.

(1) **`perform.ml`**, **`perform.scm`**, **`perform.st`** — examples comparing performing indirect operators.

(2) **`evalexpr.ml`**, **`evalexpr.scm`**, **`evalexpr.st`** — examples comparing symbolic evaluations of expressions.

15. Week 8 — February 23

(a) reviewed midterm exzm questions and answers.

(b) **`Perl-notes.d`** (1..156) — introduction to Perl.

(c) **`perl/Examples`** — a few example programs : **`hello.perl`**, **`argv.perl`**.

(d) Brief overview of asg5 **`pmake`**.

16. Week 8 — February 25

(a) **`asg4-perl-pmake`** — Perl project to implement a small subset of **`make`**.

(b) **`asg4/misc`** subdirectory with relevant example code.

(1) **`graph.perl`** — creating a graph using a hash with values pointing at arrays.
(2) **`modtime.perl`** — find a file's modification time.
(3) Various C++ programs and a Makefile showing how make reacts to various exit status codes and signal crashes.

(c) **`code/pmake`** — detailed dissection of starter code for the project.

17. Week 9 — March 2

(a) **`Perl-notes.d`** (156..end) — includeing regular expressions.

(b) Some Perl examples : **`wc.perl`**, **`text2html`**.

18. Week 9 — March 4

(a) More Perl examples : **`subst-macros.perl`**, **`nvcat.perl`**, **`switch.perl`**, **`wordfreq.perl`**, **`xref.perl`**.

(b) Notes : delayed evaluation, Haskell.

19. Week 10 — March 9

(a) **`Lecture-notes/lambda-calculus`** — Lambda calculus. abstraction, currying, beta conversion, alpha conversion, eta conversion, strict vs non-struct evaluation.

(b) **`Lecture-notes/data-types`** — static vs dyntamic types, primitive vs declared types, type constructors, type checking, compatibility. Universal and ad-hoc polymorphism.

(c) **`Lecture-notes/procedures-environments`** — procedure activation, parameter passing, stack based runtime and local stack frames, activation records.

20. Week 10 — March 11

(a) Review final exams from previous quarters.