OCaml

Getting Started Reference Sheet

Administrivia

- ◇ Command line interpreter, REPL, begun with `ocaml` and exited with `exit 0;;`
    - ○ All expressions in REPL must be terminated with `;;`, not so in a script.

        ```
        # let rec x = 4 and woah (a, b) c = print_endline a ; print_endline b; x + c ;;
        val x : int = 4
        val woah : string * string -> int -> int = <fun>

        # woah ("currying", "or not") 12 ;
        currying
        or not
        - : int = 16
        ```

        Note the keywords for mutual recursion: `let rec ...  and ....`
    - ○ For a source file: `ocamlc -i path/to/file.ml`. This prints all names and type signatures as well.
- ◇ Only multi-line, nestable, comments: `(* ...  *)`.
- ◇ All declarations are preceded by `let` or `let rec` for recursive ones.
- ◇ Sequencing is via `;` and the result is the value of the final expression.
- ◇ Anonymous functions use the syntax: `fun x ...  x -> ...`
    - ○ `function` in-place of `fun` also works.
    - ○ Infix functions can be used in prefix by enclosing them in parens; e.g., `(+) 1 2.`

Lists

- ◇ Syntax: `[x;  ...; x]`
    - ○ Tuples are optionally enclosed in parens; hence `[x,  ..., x]` is a singleton list consisting of only one tuple!
- ◇ Expected functionals: `List.nth` for list lookup, `List.map`, `List.filter`, etc.
- ◇ Cons operation is denoted `::`.
- ◇ Arrays have syntax `[|x;  ...; x|]` with 0-indexing lookup `arr.(n)`.