Cheatsheet

Control flow

- if Standard if statement.
- and Returns true if all arguments to it are true.
- or Returns true if any arguments to it are true.
- for Looping structure. A single for can represent many nested loops.
- cond Case statement.

Collections

- reduce Will run a function over a collection. The function will be passed the result of the last function call and each item in the collection.
- filter Takes test function and a collection. Returns a new collection with only the items from the first collection that caused the test to pass.
- map Takes a function and a collection. Returns a new collection with the result of running the function over each item in the collection. Be careful. It is lazy.
- mapv Like map but not lazy. Do not use with infinite sequences.
- concat Joins two or more sequences.
- set Turns a sequence into a set.
- first Returns first item of a sequence.
- rest Returns all but the first item of a sequence.
- take Pulls the first n items out of a sequence.
- take-while Pulls items out of a sequence until the test function returns false. No further items will be checked.
- range Creates a new sequence of numbers. Can be given a lower bound, upper bound and amount to increment by.
- iterate Creates a sequence by running a function over a value. Then a function over the return value of the first call, and so on.

Predicates

- = Equality check.
- + Addition.
- - Subtraction.
- ≥ Greater than.
- ≤ Less than.
- mod Modulo.
- rem Remainder.
- even? Is number even?
- odd? Is number odd?
- every? Runs a test function over a collection. Returns true if test passes for each item in collection.

Functions

- defn Define a function.
- fn Create a function.

Strings

- str Convert to string.

Naming

- let Allows you to name things. Like local variables.
- def Give something a global name.

Output

- println Output a string with a newline.