Useful functions

Push

- push ➔ add values to an array
- `push @array 5;`
  - adds 5 to end of @array
- `push @foo (4, 3, 2);`
  - adds 4, 3, and 2, to the end of @foo
- `@a = (1, 2, 3); @b = (10, 11, 12);`
- `push @a @b;`
  - @a now ➔ (1, 2, 3, 10, 11, 12)

Pop

- removes last element of array, and returns it
- `@array = (1, 5, 10, 20);`
- `$last = pop @array;`
  - $last contains value 20
  - @array contains (1, 5, 10)
- `@empty = ();`
- `$value = pop @empty;`
  - $value gets `undef`.

shift

- analogous to popping from beginning
- `@array = (1, 5, 10, 20);`
- `$first = shift @array;`
  - $first contains value 1
  - @array contains (5, 10, 20);
- `@empty = ();`
- `$value = shift @empty;`
  - $value gets `undef`

unshift

- analogous to pushing at beginning
- `unshift @array 5;`
  - adds 5 to front of @array
- `unshift @foo (4, 3, 2);`
  - adds 4, 3, and 2, to the front of @foo
- `@a = (1, 2, 3); @b = (10, 11, 12);`
- `unshift @a @b;`
  - @a now ➔ (10, 11, 12, 1, 2, 3)

splice

- all functionality of push, pop, shift, unshift
  - (plus a little bit more)
- Formally:
  - `splice ARRAY, OFFSET, LENGTH, LIST`
- remove LENGTH elements from ARRAY, starting at OFFSET, and replace them with LIST.
- In scalar context, return last element removed
- In list context, return elements removed

splice w/o some arguments

- `splice ARRAY, OFFSET, LENGTH, LIST`
- Omit LIST: remove elements, don't replace
- Omit LENGTH: remove all elements starting at OFFSET
- Omit OFFSET: clear entire ARRAY as it's being read

splice equivalencies

- `splice ARRAY, OFFSET, LENGTH, LIST`
- `push @a ($x, $y);`
  - `splice (@a, @a, 0, $x, $y);`
- `pop @a;`
  - `splice (@a, $#a);`
  - `splice (@a, -1);`
- `shift @a;`
  - `splice (@a, 0, 1);`
- `unshift @a ($x, $y);`
  - `splice (@a, 0, 0, $x, $y);`
- `$a[$x] = $y;`
  - `splice (@a, $x, 1, $y);`

keys, values

- keys ➔ get list of all keys from a hash
  - seemingly random order (kind of)
- values ➔ get list of all values from a hash
  - same 'random' order as keys produces
- `%months = ('Jan' => 'January, 'Feb' => 'February', 'Mar' => 'March', …);`
- keys (%months) == ('Jan', 'Feb', 'Mar', …)
- values (%months) == ('January', 'February', 'March', …)
  - NOT necessarily in that order.