Functions with multiple return values

- We generally assume a function returns a single value
- We can actually make it return multiple separate values (without putting them in a list)
- Requires use of “values” function to do the return
- By default just the first returned value is captured
- Requires use of either “nth-value” or “multiple-value-bind” to capture the extra returned values

Values and nth-value

- We can return multiple values using values function, and capture them one at a time with nth-value, e.g.

```
(defun f ( x )
    ; if x is numeric return sqrt x, -x, and x^2
    (if (numberp x)) (values (sqrt x) (- x) (* x x))))

(defvar first (f 3))  ; just captures the sqrt of x
(defvar second (nth-value 1 (f 3))) ; captures the -x
(defvar third (nth-value 2 (f 3))) ; captures the x^2
```

multiple-value-bind

- Multiple-value-bind lets us create a local block, with a list of variables that we'll initialize from the return values
- We can then use the values in the rest of the block

```
(multiple-value-bind
    (first second third) (f 3)
    (format t "sqrt is ~A~%" first)
    (format t "negative is ~A~%" second)
    (format t "square is ~A~%" third))
; note: if we ask for more values than are returned
;       then the extras just default to nil
```