# A Larger Example

# Evaluating an Expression

- Consider a *string* containing a simple (and restricted) mathematical expression.
- We allow
  - integers
  - names with values stored in a key-value store
  - the operators `+` and `*`
- Examples of input
  - `"43" => 43`
  - `"12+7*9+x"` (with `x=12`) `=> 87`
  - `"2*3+x*y*4+7*x"` (with `x=9, y=6`) `=> 285`
- The examples can serve as *test cases*
  - We want a way to test code automatically
  - HUnit to the rescue

# Solving the problem

- Be warned: this is not a general solution
- Split the input string at occurrences of +
  - `“12+2*3+x*y*4+7*x”`
    - `[“12”, “2*3”, “x*y*4”, “7*x”]`
- If we know the value to each multiplication, we get the end result of by adding them.
- Split each term at occurrences of *
  - `“x*y*4”`
    - `[“x”, “y”, “4”]`
- If we know the value of each factor, we get the end result by multiplying them
- Two cases:
  - only digits in the string => convert to number
  - otherwise => look up value of name

# Functions needed

- Split a list at a given element, result can be several lists
  - Use a helper that just splits at the *first* occurrence
- Function to add all elements in a list
- Function to multiply all elements in a list
- Function to convert a string to a number
  - `read :: Read a => String -> a`
  - `read “4711” :: Int`
- Function to look up the value of a name
  - `getKey :: key -> KVS key value -> value`
  - `setKeys :: [(key, value)] -> KVS key value`
  - These will be provided