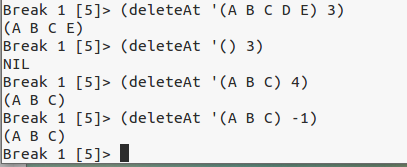
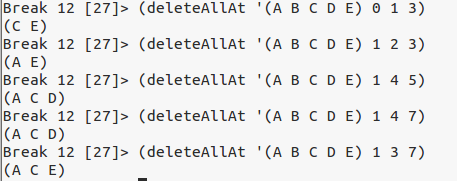
Neal Friedman

Outputs

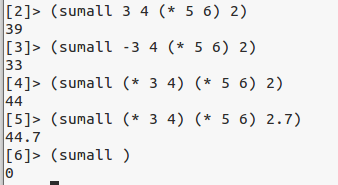
1. Write a LISP function deleteAt which takes two arguments: a list and an integer and returns the list with the high-level element at that location (zero-based indexing) deleted.



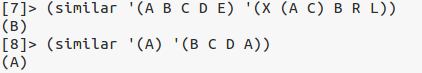
1. Write a LISP function deleteAllAt which takes as arguments a list and one or more integers and returns the list with the high-level elements at all the locations (zero-based indexing) deleted. You can assume the integers are in ascending order.



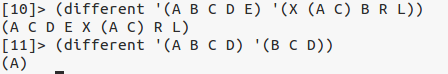
1. Write a LISP function sumAll that will return the sum of the values of its arguments.



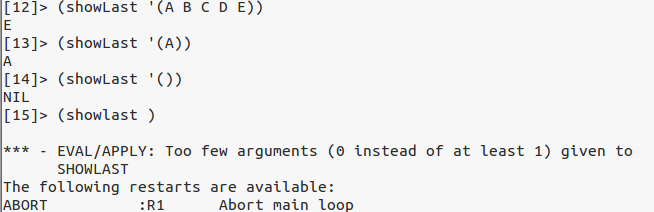
4. Write the LISP function similar that takes two lists as arguments and returns a list of the high-level elements that are in both lists.



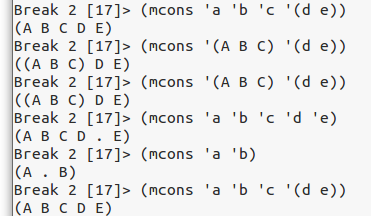
5. Write the LISP function alone that takes two lists as arguments and returns a list of the high-level elements that are in one or the other but not in both lists.



6. Write a function that takes a list as an argument and returns the last element of the list.



7. Write a version of **cons** called **mcons** that takesany number of arguments. The penultimate argument should be **cons**ed into the last; the one before that should be **cons**ed into the resulting value and so on.



8. Write a function **nth** which takes two arguments

* a list of lists
* and an int

The function nth should use the int as an index into every sublist and return a list of the elements in the sublists at that position.

