2013 - 2

a) If perms is passed an empty list it returns a list containing the one empty permutation of an empty list

If it is passed a ~~list~~ non-empty list, then perms1 is declared, which takes two lists as arguments, the second being an accumulator. If argument 1 is an empty list an empty list is returned, otherwise the mapping of cons xs is applied to a recursive call. Where cons xs is a partial function application of cons xs ys and returns a function which will cons xs onto all elements of the list it is mapped to. The recursive call is an append of ys and xs reversed given as an argument to perms appended to a call of the tail of x:ixs and xs appended to the accumulator. This means it will ~~cons~~ create permutations beginning with each element of the list, each time adding to these permutations with the permutations of each list that is the tail of a list minus the element used. This then creates all permutations of a list.

perms [1,2,3] = [[1,2,3], [1,3,2], [2,1,3], [2,3,1], [3,1,2], [3,2,1]]