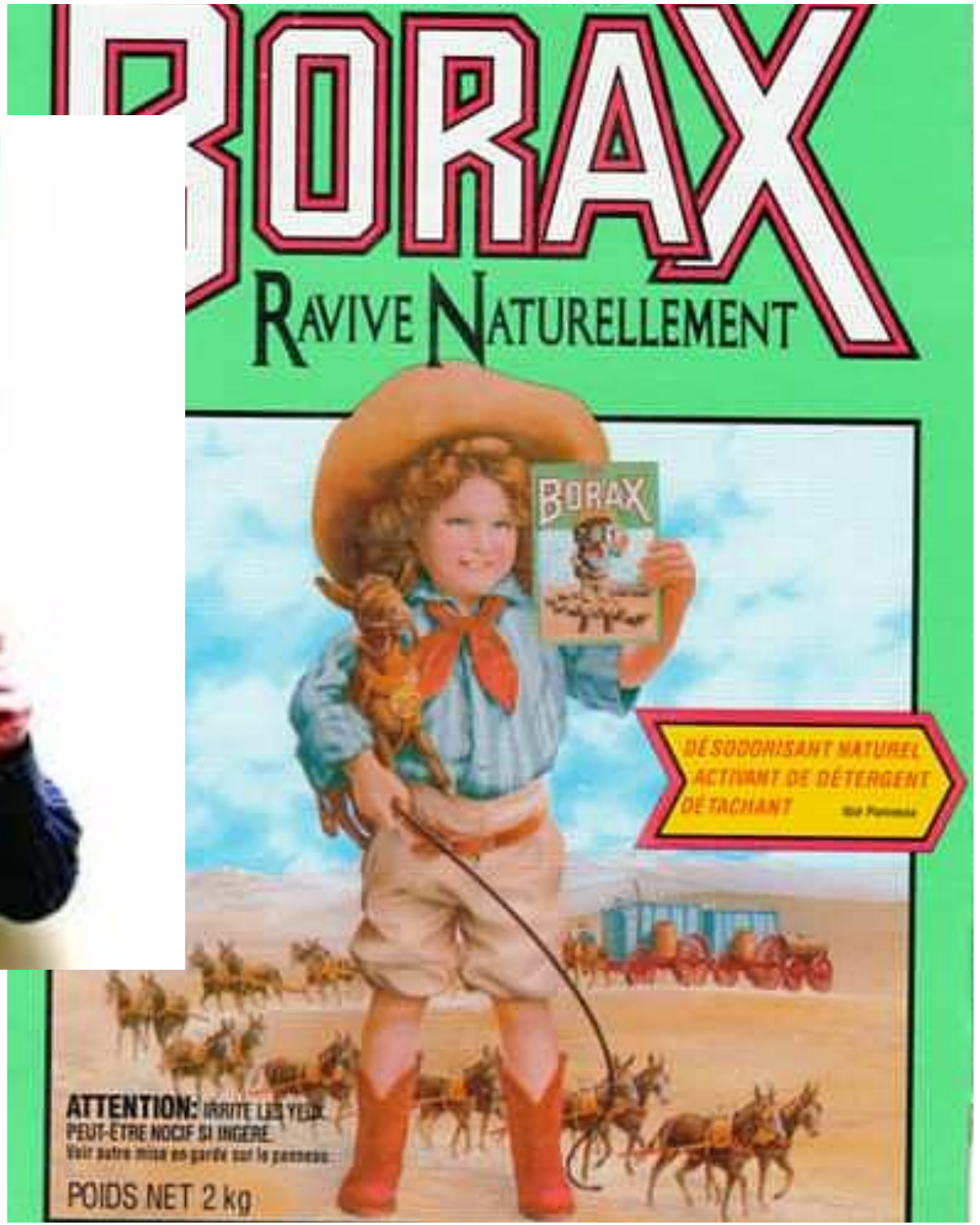
Recursion continued…


BORAX
RAVIVE NATURELLEMENT
BORAX
DÉSODORISANT NATUREL
ACTIVANT DE DÉTERGENT
DÉTACHANT
ATTENTION: IRRITE LES YEUX.
PEUT-ÊTRE NOCIF SI INGÉRÉ.
POIDS NET 2 kg

recall template for a recursive algorithm

```
recursiveFunction(data)
    if (smallestPossibleProblem? data)
        the simple answer
        return ...
    else
        first part of data ...
        recursiveFunction(smallerProblem(data))
        return ...
```

Basecase

Deal with one piece of the data

Recursive call

Exercise

- Write the pseudocode for a recursive algorithm that will take take a array of numbers and the length of that array and returns the product of the values in that array.
- Recall:
  -the product of numbers (n1 and n2) is: n1 * n2
  - the result of multiplying no numbers is 1

Process:

- Input/output?
- Examples:
  - Simplest example of calling the function
  - A more complex example
- Edit the template
  - Rename function & data
  - Edit the basecase
  - Deal with one data piece
  - Update to smaller problem for recursive call

input – array, length
output – product of numbers in array of specified length

```
productArray(array, length)
    if (length == 0)
        return 1
    else
        result = array[length-1] * productArray(array, length-1)
        return result
```

Exercise

- Write the pseudocode for a recursive linear search algorithm.
- It will take take an array of numbers, the length of that array and the value being searched for. It will return the index the value is found at or -1 if it is not found.
- Recall, linear search visits each element one after another. TIP: You can search from back to front of the array.

Process:

- Input/output?
- Examples:
  - Simplest example of calling the function
  - A more complex example
- Edit the template
  - Rename function & data
  - Edit the basecase
  - Deal with one data piece
  - Update to smaller problem for recursive call

input – array, length, value
output – product of numbers in array of specified length

```
search(array, length, value)
   if (length == 0)
      return -1
   else
      if (array[length-1] == value)
            return(length-1)
       else
            return search(array, length-1, value)
```

Exercise

- Write the pseudocode for a recursive binary search algorithm.
- It will take take an array of numbers, the minIndex and the maxIndex to search in the array and the value being searched for. It will return the index the value is found at or -1 if it is not found.
- Recall binary search begins looking in the middle of the array and splits the search space in half with each check.

Process:

- Input/output?
- Examples:
  - Simplest example of calling the function
  - A more complex example
- Edit the template
  - Rename function & data
  - Edit the basecase
  - Deal with one data piece
  - Update to smaller problem for recursive call

recall the iterative binarySearch…

```
binarySearch(list, value)

    N = number of elements in list
    minIndex =0
    maxIndex = N-1
    while (minIndex <= maxIndex)
        middle = (minIndex + maxIndex)/2
        currentItem = list[middle]
        if (currentItem == value)
            return middle
        else if (currentItem > value)
            maxIndex = middle-1
        else
            minIndex = middle+1

    return -1 // not found
```

input – array, minIndex, maxIndex, value
output – product of numbers in array of specified length

```
binarySearch(array, minIndex, maxIndex, value)
   if (minIndex>maxIndex)
      return -1
   else
      middle = (minIndex+maxIndex)/2
      if (array[middle] == value)
            return middle
      else (value<array[middle])
            return search(array, minIndex, middle-1, value)
      else
            return search(array, minIndex+1, maxIndex, value)
```