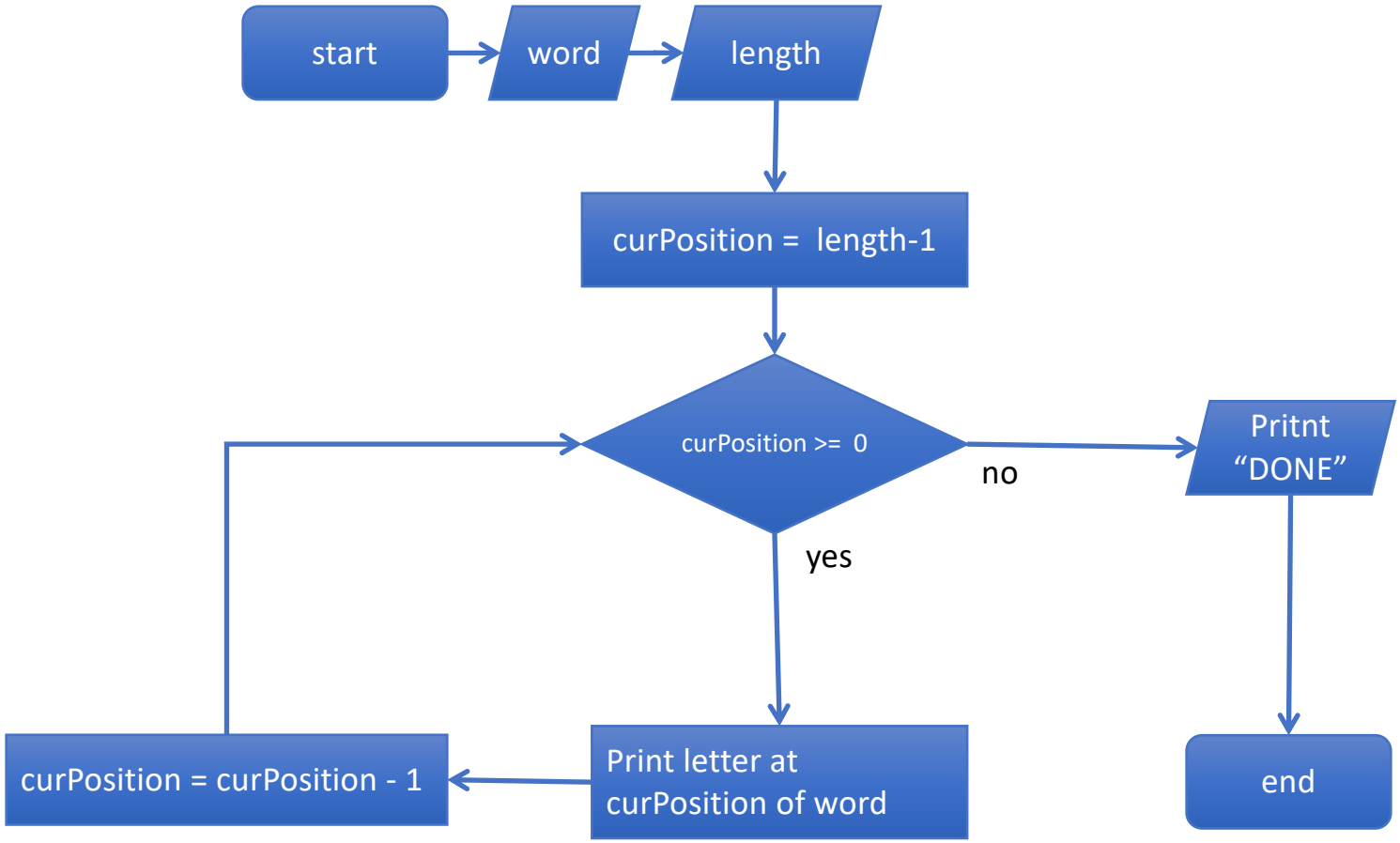
More algorithm practice

exercise

- Create flowchart for an algorithm that
  - prints out each letter (a character) in a word or a sentence (a string) from back to front
  - Your algorithm should print “DONE” at the end
  - NOTE: the first letter in a word is said to be a position 0 and the last letter is at position one less than the length of the word
- Write the corresponding pseudocode for your flowchart

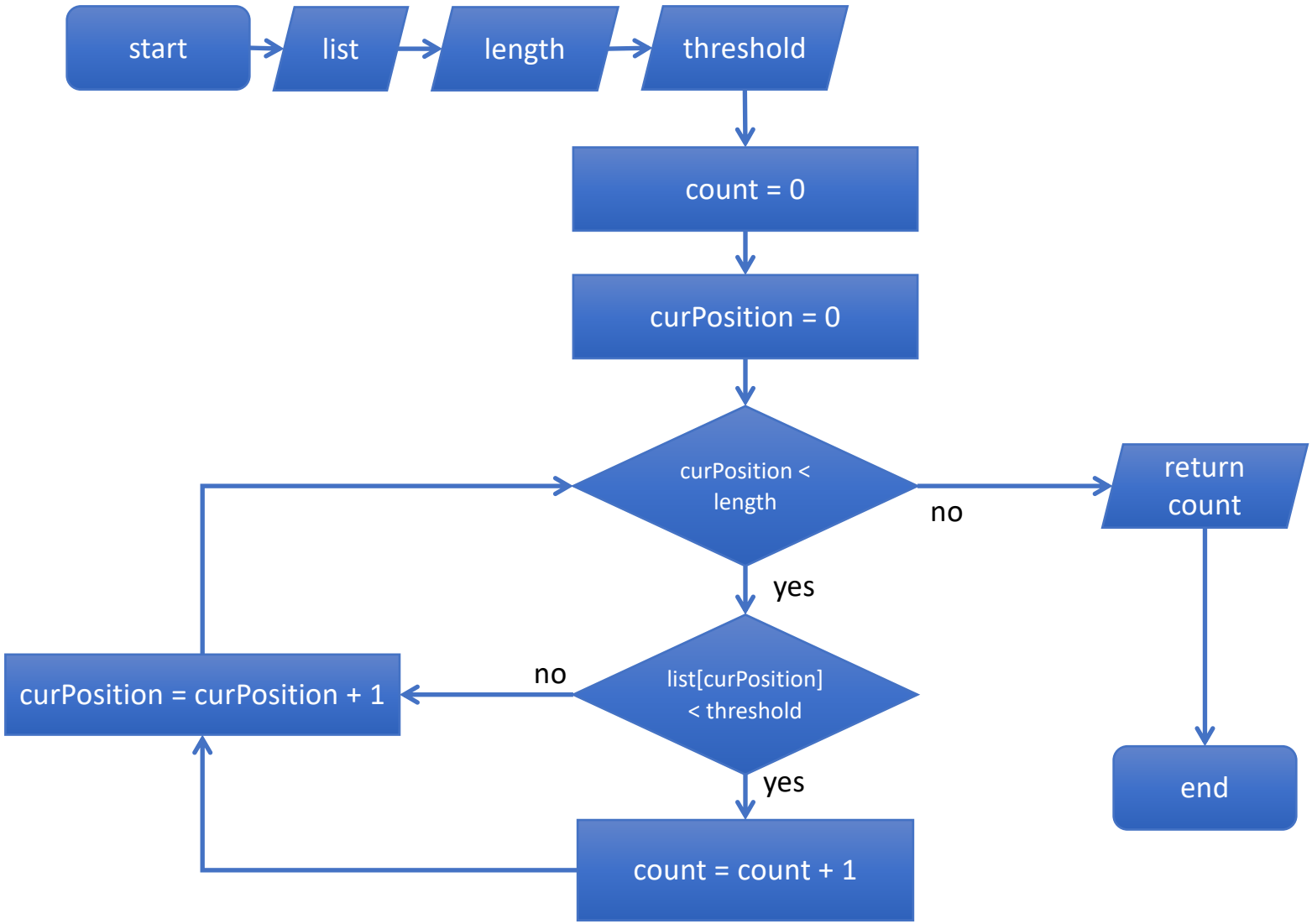

start
word
length
curPosition = length-1
curPosition >= 0
no
yes
Pritnt
“DONE”
curPosition = curPosition - 1
Print letter at
curPosition of word
end

pseudocode

```
reverse(word, length)
        curPosition = length - 1
        for curPosition down to 0 (inclusive)
                print word[curPosition]
        print “DONE”
```

exercise

- Create flowchart for an algorithm that takes a list of numbers, the length of the list and another number as a threshold (limit). The algorithm should return a count of the number of values in the list that are less than that threshold.
- Write the corresponding pseudocode for your flowchart


start
list
length
threshold
count = 0
curPosition = 0
curPosition < length
no
return count
yes
list[curPosition] < threshold
no
curPosition = curPosition + 1
yes
count = count + 1
end

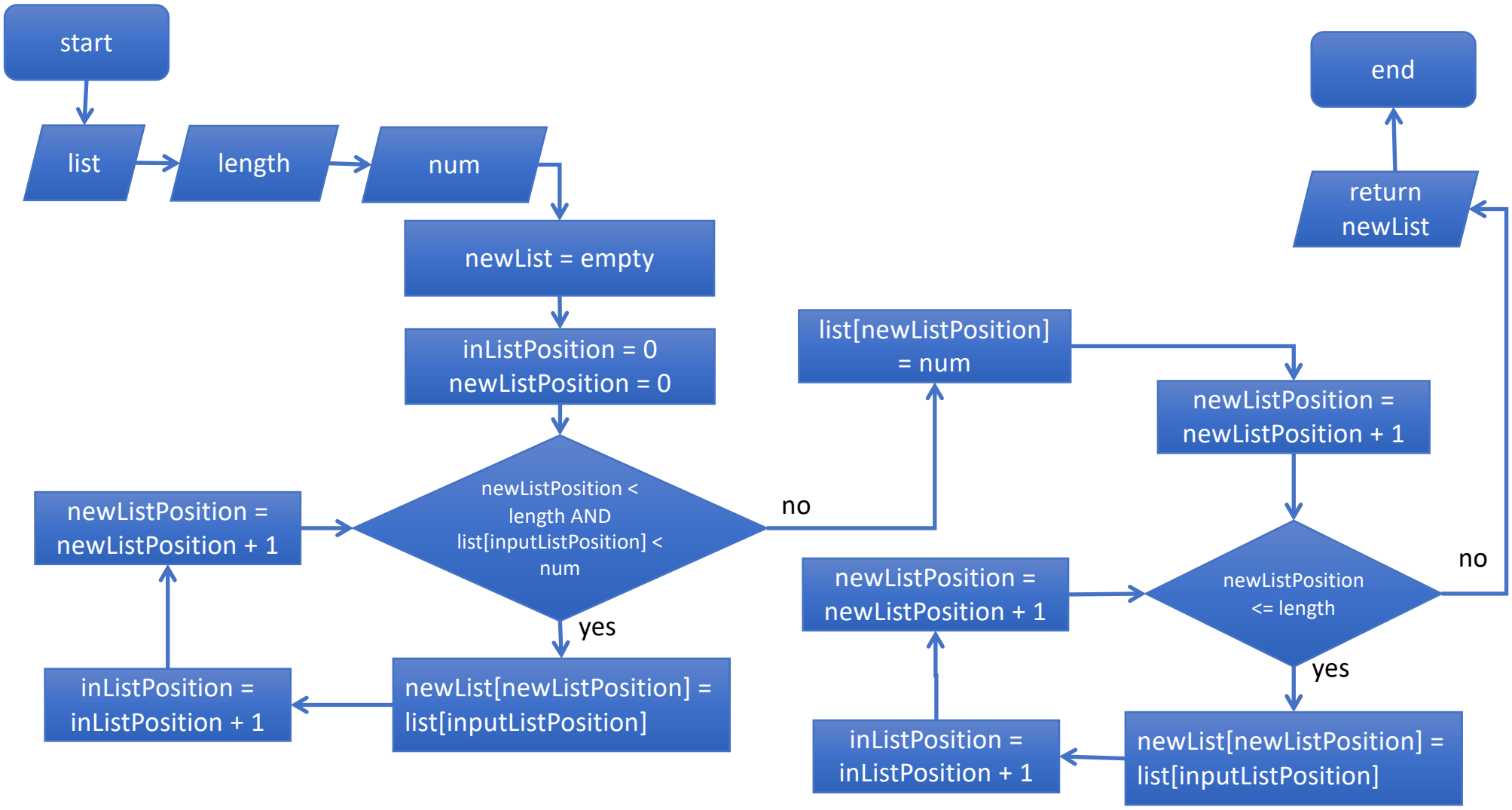
pseudocode

```
countBelow(list, length, threshold)
        curPosition = 0
        for (curPosition up to length-1 (inclusive))
                if (list[curPosition] < threshold)
                        count = count + 1

        return count
```

exercise

- This is a more challenging problem – focus on the previous 2 if you are finding these challenging
- Create a flowchart for an algorithm that takes a list of numbers that is in increasing sorted order, the length of that list and another number to insert. The algorithm should return a new list in increasing sorted order that includes all numbers from the list and the inserted number.
- Write the corresponding pseudocode for your flowchart


start
list
length
num
newList = empty
inListPosition = 0
newListPosition = 0
newListPosition < length AND list[inputListPosition] < num
yes
no
newListPosition = newListPosition + 1
inListPosition = inListPosition + 1
newList[newListPosition] = list[inputListPosition]
list[newListPosition] = num
newListPosition = newListPosition + 1
newListPosition <= length
yes
no
inListPosition = inListPosition + 1
newList[newListPosition] = list[inputListPosition]
return newList
end

```
insert(list, length, num)
        newList = empty
        inputListPosition = 0
        newListPosition = 0

        while ( newListPosition < length AND list[inputListPosition] < num)
                newList[newListPosition]   = list[inputListPosition]
                newListPosition            = newListPosition + 1
                inputListPosition          = inputListPosition + 1

        newList[newListPosition] = num
        newListPosition            = newListPosition + 1

        while (newListPosition <= length)
                newList[newListPosition]   = list[inputListPosition]
                newListPosition            = newListPosition + 1
                inputListPosition          = inputListPosition + 1

        return newList
```