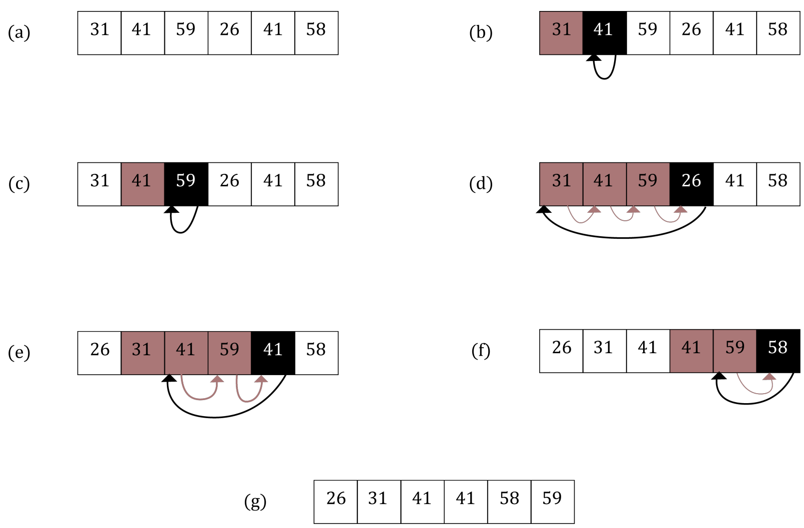
Chapter 2 Part 1 Exercises

2.1-1

The steps are the following:



2.1-2

The new instertion-sort will be the following:

```
Reverse-nsertion-Sort(A)
for j = 2 to A.lenght
        do
            key = A[j]
            i = j - 1
            while i > 0 and A[i] < key
                do
                    A[i + 1] = A[i]
                    i = i - 1
            A[i + 1] = key
```

2.1-3

The linear search algorithm will be:

```
Linear-Search(A, v)
while i <= A.length
        do
            if A[i] = v
                then
                    return i
            i = i + 1
return NIL
```

If the linear search finishes, then v has not been found in the array, so we return NIL. Otherwise, the loop stops when the first occurance of v is found.

The **loop invariant** of the algorithm is:

At the start of each iteration of the loop, the subarray $A[1..i-1]$ does not hold the value v.

Let us see now how the loop invariant properties hold now.

Initialization: We start by showing that the loop invariant holds before the first loop operation, when $i = 1$. The subarray of A is empty, so by definition it does not contain v.

Maintenance: Informally, the body of the while loop compares v with $A[i]$ and updates exits the lopp if they are equal. The subarray $A[1..i]$ consists of elements that are not equal to v, as otherwise the loop would have ended.

Termination: We examine what happens when the loop terminates. When the loop terminates the value of $i = A.lenght + 1 = n + 1$. Then, the whole array A is the left subaray $A[1...i]$, so we have gone through the whole array and not found the value v, so we return NIL.

2.1-4

Stating the problem formally:

Input: Two arrays A, B of size n containing binary digits.

Output: An array C, which is of size $n + 1$ and contains the binary sum of A and B.

Code:

```
BINARY-ADDITION(A, B, C)
carry = 0
for i = A.length downto 1
        do
            C[i + 1] = A[i] + B[i] + carry
            if C[i + 1] = 2
                then
                    C[i + 1] = 0
                    carry = 1
            elseif C[i + 1] = 3
                then
                    C[i + 1] = 1
                    carry = 1
                else
                    carry = 0
C[1] = carry
```