Cormen Exercises I.2.1

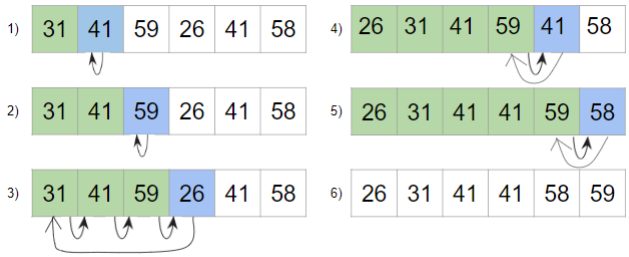
September 2018

José Daniel Organista Calderón

Exercises

2.1-1

Using Figure 2.2 as a model, illustrate the operation of INSERTION-SORT on the array A = {31, 41, 59, 26, 41, 58}.



2.1-2

Rewrite the INSERTION-SORT procedure to sort into nonincreasing instead of nondecreasing order.

```
INSERTION-SORT(A)
1 for j = 2 to A.length
2    key = A[j]
3    i = j - 1
4    while i >0 and A[i] >key
5         A[i + 1] = A[i]
6         i = i - 1
7    A[i + 1] = key
```

2.1-3

Consider the ***searching problem:***

Input: A sequence of n numbers A = {a_1, a_2, ... , a_n} and a value .

Output: An index i such that v = A[j] or the special value NIL if does not appear in A.

Write pseudocode for linear search, which scans through the sequence, looking for . Using a loop invariant, prove that your algorithm is correct. Make sure that your loop invariant fulfills the three necessary properties.

Pseudocódigo

```
LinearSearch(A,v)
1 i = NIL
2 for j = 1 to A.length
3   if A[j] = v then
4       i = j
5       break
6 return i
```

Tenemos como invariante que no existe un indice k menor a j tal que A[k]=v por cada iteracion en el loop. Si en un caso se llega a cumplir que A[j] = v el ciclo retorna i y se cumple la invariante dado que j es mayor a k y no existe un indice k menor a j tal que A[k]=v.En caso contrario se realizan todas las comparaciones entre A[j] con v se termina el loop y se retorna i que ha sido inicializado con NIL. La invariante se conserva, dado que en ningún punto existió un k menor a j tal que A[k] = v

2.1-4

Consider the problem of adding two n-bit binary integers, stored in two n-element arrays A and B. The sum of the two integers should be stored in binary form in an (n+1) element array C. State the problem formally and write pseudocode for adding the two integers.

Pseudocódigo

Input: A,B arreglos de tamaño n
Output: C, un arreglo de tamaño (n + 1) con la suma de A y B

```
addingBinary(A,B)
1   C =[],carry = 0
2   for i = n to 1
2       C[i+1] = (A[i] + B[i] + carry)
3       if(C[i+1]) >= 2 then
4          carry = 1
5       else
6          carry = 0
7       C[i+1] = (C[i+1]) mod 2
8   C[1] = carry
```

9 **return** C