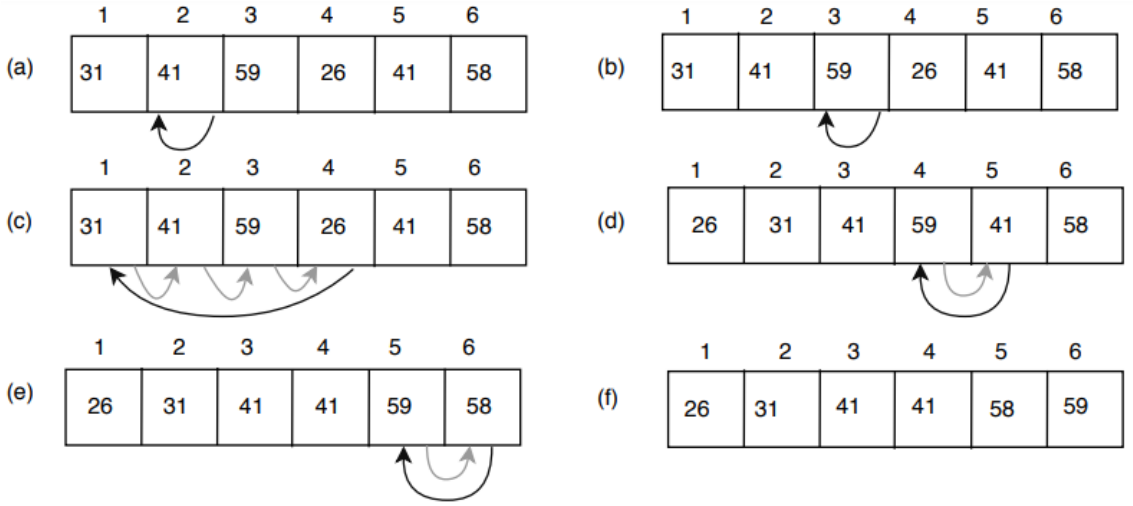
Algoritmos
Universidad Nacional de Colombia
Ingeniería de Sistemas y Computación

David Fernando Guerrero Alvarez

September 2018

1 Taller 1

1) Using Figure 2.2 as a model, illustrate the operation of INSERTION-SORT on the array A = $\langle 31, 41, 59, 26, 41, 58 \rangle$.



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

```
//"Cormen"
for j = 2 to A.length
    key = A][j]
    i = j -1
    while i > 0 and A[i] < key
        A[i + 1] = A[i]
        i = i - 1
    A[i + 1] = key

//"Código Python"
def insertionSortInverse(array):
   for i in range(1,len(array)):
       aux = array[i]
       j = i - 1
       while j > -1 and aux > array[j]:
```

```
            array[j + 1] = array[j]
            j = j - 1
        array[j + 1] = aux

In: [5, 4, 6, 3, 7, 2, 8, 1, 9]
Out: [9, 8, 7, 6, 5, 4, 3, 2, 1]
```

3) Consider the ***searching problem***:

Input: A sequence of n numbers A = $\langle a1, a2, ..., an \rangle$and a value v.

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

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

```
"Programa: Búsqueda secuencial"
*Variables
arreglo A[]
entero v

*Pseudocódigo:
Funcion Busqueda(A[], v):
  Para i = 0 hasta i < len(A) con incremento 1 hacer:
    Si A[i] == v entonces
      escribir "Valor Encontrado"
      Retornar i
    Fin Si
  Fin Para
  Retornar NIL
Fin
```

Invariante de Loop: Al inicio de cada iteración, el Subarreglo A[0...i-1] estará compuesto de elementos diferentes de v. **Inicializacion:** Al iniciar el subarreglo está vacío, por lo que la condición se cumple de manera trivial .

Mantenimiento: Para cada iteración en proceso, sabemos que el subarreglo de elementos anteriores al índice de la iteración no contiene a v, por lo que compramos el elemento A[i] con v y comprobamos si son o no iguales, en caso de no serlo, continúa el invariante y el proceso, de otra forma, retorna la posición i.

Terminacion:
El proceso finaliza al llegar al temaño máximo del arreglo, y como todos los elementos fueron comparados, podemos decir que el invariante se mantuvo hasta el final, pues todos los elementos de A son difetenes de v; de este modo el resultado que obtenemos es NIL.

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.

```
Declaramos las variables A,B,C:
A[0] ... A[n-1] (tam = n)
B[0] ... B[n-1] (tam = n)
C[0] ... C[n] (tam = n+1)
Y se toman A[0] y B[0] como los bits más significativos.

//Pseudocódigo:
```

```
Carry = 0
Para i = n - 1 a 0
    C[i+1] = (A[i] + B[i] + Carry) mod 2
    Carry = (A[i] + B[i] + Carry) / 2
C[0] = Carry
```