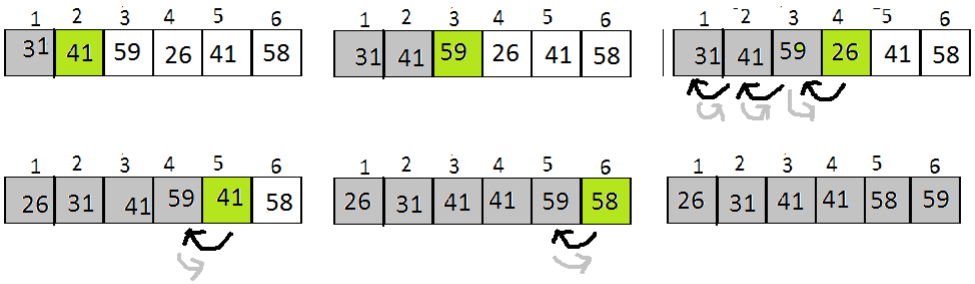
Lab1

jjmontoyag

September 2018

1 Insertion sort



2 Non-increasing

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted
           sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] < key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

3 Linear sort

```
LinearSort(A, v)
.       found=NIL
.       for i=1 to A.length
.           if A[i]==v
.               found=i
.               break
.       return found
```

3.1 Initialization

El arreglo *A[1...j]* consiste de todos los elementos a buscar, con found igual a falso, e i=1 comenzando en *A[1]* , por lo tanto se mantiene el bucle antes de comenzar.

3.2 Maintenance

El bucle **for** se mueve hacia la derecha buscando todo el arreglo para hallar un $v = A[i]$ o en caso contrario no hacer nada.

3.3 Termination

La condición para que el **for** termine es cuando i mayor a j. En este punto se habrá encontrado un $v = A[i]$ e inmediatamente salido del **for** y retornado el valor en found. Si el valor no es encontrado se retorna el valor original del found que es NIL.

4 Adding n-bit binary arrays

```
ADD(a, b, L, i)
.       if a==b and a==1
.           L[i-1]=1
.           if L[i]==0
.               return 0
.           else
.               return 1
.       if a!=b
.           if L[i]==0
.               return 1
.           else
.               L[i-1]=1
.               return 0
.       if a==b and a==0
.           if L[i]==0
.               return 0
.           else
.               return 1

   SUM(A, B)
.      C[A.length]=L[A.length]=zeros
.      for i=A.length-1 to 0
.          C[i]=ADD(A[i], B[i], L, i)
.      C[1]=L[1]
.      return C
```