

Ejercicios Cormen

Luis Felipe Dávila Goyenche

September 2018

1 Exercises

1.1 Problem 2.1-1

Using Figure 2.2 as a model, illustrate the operation of INSERTION-SORT on the array $[31, 41, 59, 26, 41, 58]$.

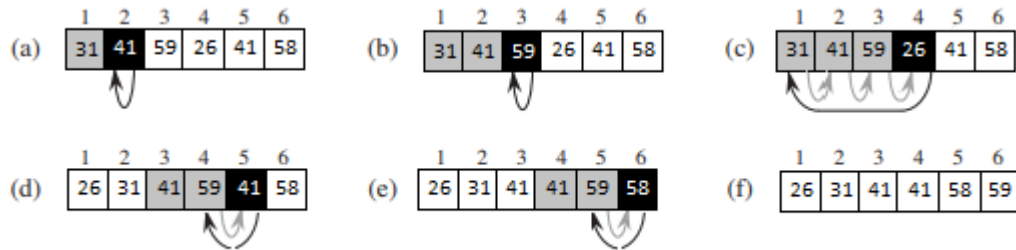


Figure 1: The operation of INSERTION-SORT on the array $[31, 41, 59, 26, 41, 58]$.

1.2 Problem 2.1-2

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

Algorithm 1 INSERTION-SORT

```
1: procedure INSERTIONSORT(array)
2:   for  $j \leftarrow 2$  to  $\text{lenthg}(\text{array})$  do
3:      $\text{key} \leftarrow \text{array}[j]$ 
4:     while  $j > 0$  and  $\text{key} > \text{array}[j - 1]$  do
5:        $\text{array}[j] \leftarrow \text{array}[j - 1]$ 
6:        $j \leftarrow j - 1$ 
7:      $\text{array}[j] \leftarrow \text{key}$ 
```

1.3 Problem 2.1-3

Consider the searching problem:

Input: A sequence of n numbers $[a_1, a_2, \dots, a_n]$ and a value n .

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

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

Algorithm 2 linear-Search

```

1: procedure LINEARSEARCH( $A, n$ )
2:    $index \leftarrow NULL$ 
3:    $\triangleright$  Invariant( $I_j$ ) :=  $\{\forall A_j \subseteq A$  de tamaño  $j \leq length(A)$  con
       $A_j = [A[0], A[1], \dots, A[j-1]]$  se cumple que  $n \notin A_j\}$ 
4:   for  $j \leftarrow 0$  to  $length(A)$  do
5:      $\triangleright \{(j < length(A)) \cap (A[j] \notin A_j) \cap I_{j-1} \longrightarrow I_j\}$ 
6:     if  $A[j] == n$  then
7:        $\triangleright \{n \notin A_j \rightarrow I_j\} A[j] == n \{A[j] \notin A_j\}$ 
8:        $index \leftarrow j$ 
9:        $\triangleright \{(A[index] = A[j]) == n\} index = j \{A[index]\}$ 
10:    return  $index$ 
11:     $\triangleright index \neq NULL$  is solution y
       $I_j == \text{true}$  para el final del loop
12:  return  $index$ 

```

1.3.1 Prueba por inducción

Initialization: $j = 0$

$$(A_0 = \emptyset) \subseteq A \rightarrow A[0] \notin A_0 \quad (1)$$

Si $A[0] == n$ o $A[0] \neq n$ entonces $n \notin A_j$ y el invariante de loop es verdadero justo antes del for.

Maintenance: Suponer para $j = k < length(A)$ y $A[j] \neq n$

$$(A_k = [A[0], A[1], \dots, A[k-1]]) \subseteq A \rightarrow A[k] \notin A_k \quad (2)$$

Se demuestra para $j = k + 1$

$$A_{k+1} = [A[0], A[1], \dots, A[k-1], A[k]] = A_k \cup [A[k]] \quad (3)$$

$$A_k \cup [A[k]] \subseteq A \quad (4)$$

Dado que $n \notin A_k$ y $A[k] \neq n$ entonces $n \notin A_{k+1}$ por tanto la ivariante de loop se mantiene durante el cuerpo del loop siempre que $A[j]$ sea distinto de n

Termination: El loop finaliza cuando $A[j] == n$ o cuando $j = length(A)$.

En el el primer caso es evidente que $A[j] \notin A_j$ y por tanto si $A[j] == n$ entonces $n \notin A_j$ y la invariante se mantiene hasta el instante en que se realiza el retorno de la función, es decir, cuando finaliza el loop. En el caso que $j = \text{length}(A)$ entonces significa que $A_j = A$ y por tanto $A[n]$ no es un índice válido para acceder al arreglo, esto significa que no se encontró un objeto en el arreglo que fuese igual al valor del entrada o $n \notin (A_j = A)$ y nuevamente el invariante se cumple incluso en la finalización del loop.

References

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms. McGraw-Hill, 2001.

James Apnes. Correctness proof. 2003