laboratorio 2

José David Ardila Acosta

September 2018

1 Resultados Fibonacci

Para el fibonacci hecho en C y en Java se observo que el archivo tipo byte se desbordaba en el 14 ciclo, el archivo tipo short se desbordaba en el 23 ciclo, el archivo tipo entero se desbordaba en el 46 ciclo, el long la maquina no soportaba tal cantidad de iteraciones y no pudo ser comprobado.

Para el fibonacci hecho en python para 47 procesos le tomaba 1105.3 segundos (mas o menos 20 min) en pythonanywhere por ello se dejo en ese valor como ultima prueba y dio como resultado 4807526976

2 Resultados Insertion Sort

3 2.1-1

Usando la figura 2.2 como modelo, ilustrar la operación INSERTION-SORT en el arreglo 31,41,59,26,41,58

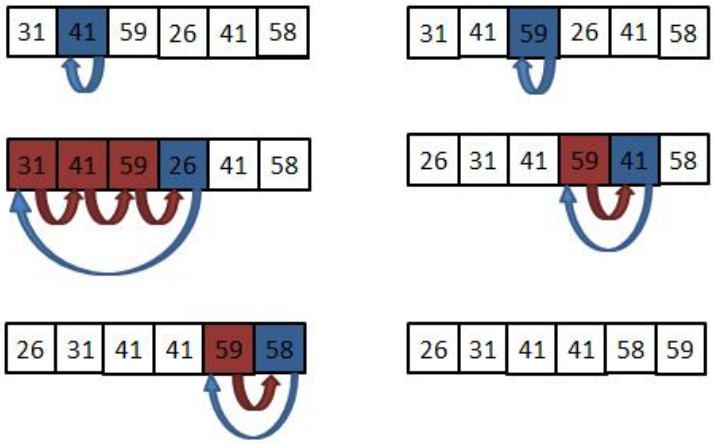

Figure 1: operación INSERTION-SORT

4 2.1-2

Reescribir el procedimiento del INSERTION.SORT para organizar en orden ascendente en lugar de orden descendente.

```
1 for j = 2 to A:length
2   n = A:length - 2 - j
3   key = A[n]
4   i = n + 1
5   while i < A:length and A[i] < key
6       A[i-1] = A[i]
7       i = i + 1
```

```
8   A[i - 1] = key
```

5 2.1-3

Considerando el pproblema de busqueda:

Input: Secuencia de n numeros $A = a_1, a_2, ..., a_n$ y un valor v

Output: Un indice i tal que $v = A[i]$ o null si v no pertece a A.

Escriba un pseudocodigo para una busqueda lineal, que revise por la secuencia buscando v. Usando un ciclo invariante, pruebe que su algoritmo funciona

```
1 read A
2 j = 0
3 i = null
4 while j < A:length
5     if A[j] = v
6         i = j
7         break
8     j = j + 1
9 return i
```

```
def search(A, v): #read A
    j = 0
    i = 'NIL'
    while j < len(A):
        if A[j] == v:
            i = j
            break
        j = j + 1
    return i

v = 4

A = [0,1,2,3,4]
print "el indice es", search(A, v)
```

```
el indice es 4
```

```
A2 = [1,2,3,5,6]
print "el indice es", search(A2, v)
```

```
el indice es NIL
```

```
A3 = [0,1,2,3,5,6,7,4,8,9,10,15]
print "el indice es", search(A3, v)
```

```
el indice es 7
```

Figure 2: Prueba del algoritmo

6 2.1-4

considere el problema de añadir dos enteros binarios de n bits, colocados en dos erreglos de n elementos A y B. La suma de los dos enteros debe ser colocada en forma binaria en un arreglo de n+1 elementos C. exprese formalmente el problema y escriba el pseudocodigo.

```
1  read x,y # read x and y integers
2  int a = x, int b = y
3  A = binary[], B = binary[], C = binary[]
4  i = 0
5  carrier = 0
6  while (a > 1) and (b > 1)
7      A[i] = a%2
8      a = a/2
9      B[i] = b%2
10     b = b/2
11     C[i] = (A[i] + B[i] + carrier)%2
12     carrier = (A[i] + B[i] + carrier)/2
13 C[i] = carrier
```