Metodología de la Programación y Algoritmia

Convocatoria de Septiembre 2013
SOLUCIÓN

Apellidos ______________________________

Nombre ______________________________ **DNI** ______________

1.- Proteger un terreno con ubicación de dispositivos.

SOLUCIÓN:

Este ejercicio se puede resolver utilizando el algoritmo de las n-reinas, visto en clase.
Hay que incluir la tipificación del problema, describir el funcionamiento, técnica utilizada, implementar algoritmo y realizar la traza.

2.- Dada la función

```
función calcular(x:entero, y:entero):entero
      z:entero
      si x = 0
            z ← 0
      si no
            z ← x * y + calcular(x - 1, y)
      fsi
      devolver z
ffunción
```

2.a) ¿Está bien definida? Justifica tu respuesta.
2.b) ¿Qué tipo de recursividad es? Identifica todos los componentes con los del esquema general correspondiente.
2.c) Realiza la traza para x=4 e y=6, indicando en cada llamada los valores de los parámetros y el resultado parcial. ¿Cuántas llamadas se realizan a la función? ¿Cuál es el resultado final?
2.d) Obtén la versión iterativa aplicando el esquema general de transformación adecuado.
2.e) Obtén la ecuación del tiempo de ejecución T y calcula la complejidad asintótica.

SOLUCIÓN:

2.a) ¿Está bien definida? Justifica tu respuesta.

No. Como el parámetro x es un entero puede tomar valores positivos y negativos. Cuando x<0 la función nunca llega al caso base.

2.b) Tipo de recursividad e identificación de los elementos con los del esquema general.

Lineal no final.

Esquema general:

```
función F (x:tipo1):tipo2
  si B(x)
    devolver S(x)
  si no
    devolver C(F(T(x)))
  fsi
ffunción
```

Identificación de los elementos:

F:	calcular
x:	Los parámetros son: x , y
B:	x = 0
S:	0
T:	Se le resta 1 al primer parámetro y no se modifica el segundo parámetro: x - 1, y
C:	Sumar x * y al resultado de calcular(x-1,y)

donde
- F: nombre de la función recursiva
- x: parámetros de entrada
- B: condición que determina el caso base
- S: solución para el caso base
- T: transformación de los parámetros de entrada
- C: combinación de los resultados

2.c) Traza para x = 4 e y = 6.

1º) calcular(4,6)	= 4 * 6 + 36 = 60	Nº llamadas a la función: 5 Resultado final: 60
2º) calcular(3,6)	= 3 * 6 + 18 = 36	
3º) calcular(2,6)	= 2 * 6 + 6 = 18	
4º) calcular(1,6)	= 1 * 6 + 0 = 6	
5º) calcular(0,6)	= 0	

2.d) Versión iterativa aplicando el esquema.

Aplicando el esquema de inversión funcional, obtenemos la función calcular'

```
función calcular'(x:entero, y:entero):entero
   x0:entero
   s:entero

   x0 ← x
   mientras !(x0 = 0) hacer
      x0 ← x0 - 1
   fmientras
   s ← 0

   mientras !(x0 = x) hacer
      x0 ← x0 + 1
      s ← x0 * y + s
   fmientras
   devolver s
ffunción
```

2.e) T y complejidad asintótica.

Asumiremos que x no puede tomar valores negativos (ya que entonces no acabaría la función, según se ha comentado en el apartado 2.a).

T(x) = 3	si x = 0
T(x) = 7 + T(x-1)	si x > 0

Para calcular la complejidad asintótica se puede hacer aplicando los esquemas o con la técnica del desplegado.

La función T de este ejercicio se ajusta al del tipo de disminución del problema por sustracción, con lo cual podemos aplicar sus esquemas.

$$T(n) = \begin{cases} c_1 & \text{si } 0 \le n \le n_1 \\ a \cdot T(n-b) + c \cdot n^k & \text{si } n > n_1 \end{cases} \qquad T(n) \in \begin{cases} O(n^{k+1}) & \text{si } a = 1 \\ O(a^{n/b}) & \text{si } a > 1 \end{cases}$$

Identificamos los elementos: $c_1 = 3$, $n_1 = 0$, a = 1, b = 1, c = 7, k = 0.
Como a = 1, la complejidad asintótica es O(x).

Metodología de la Programación y Algoritmia

Convocatoria de Septiembre 2013

SOLUCIÓN

3.- Dada la función `algoritmo(V:&entero[n]):entero[n]`

3.a) Calcula la expresión del tiempo de ejecución T para los casos mejor y peor .

3.b) Obtén la complejidad asintótica y justifica por qué.

3.a) Obtener T.

```
                                                          Paso 2:
     función algoritmo (V:&entero[n]):entero[n]
           M:entero[n]
           x:entero
           i,j:natural

(1)        i ← 1                                          t(1) = 1
(2)        mientras i ≤ n hacer                           t(2) = 1+n
(3)              x ← V1                                   t(3) = 2
(4)              j ← 2                                    t(4) = 1
(5)              mientras j ≤ i hacer                     t(5) = i
(6)                    x ← x + Vj                         t(6) = 3
(7)                    j ← j + 1                          t(7) = 2
(8)              fmientras
(9)              Mi ← x / i                               t(9) = 3
(10)             i ← i + 1                                t(10)= 2
(11)       fmientras
(12)       devolver M                                     t(12)= n
     ffunción
```

Paso 1: Este algoritmo solamente tiene un caso.

$$T(n) = t_{(1)} + t_{(2)} + n\left(t_{(3)} + t_{(4)}\right) + \sum_{i=1}^{n}\left(t_{(5)} + (i-1)(t_{(6)} + t_{(7)})\right) + n\left(t_{(9)} + t_{(10)}\right) + t_{(12)}$$

Paso 3: Sustituyendo los valores en la expresión anterior, obtenemos

$$T(n) = 1 + 1 + n + n(2+1) + \sum_{i=1}^{n}(i + (i-1)(3+2)) + n(3+2) + n$$

….

$$T(n) = 3n^2 + 8n + 2$$

3.b) Complejidad asintótica.

Este ejercicio se puede resolver por niveles o a partir de la expresión T del apartado anterior.

La complejidad de este algoritmo es $O(n^2)$, ya que la expresión T pertenece a este orden (por la propiedad de la suma del orden superior).