Algoritmos y Estructuras de Datos II - 1º cuatrimestre 2019 Práctico 1 - Parte 3

ejercicios para el lunes 25/3

1. El siguiente algoritmo calcula el mínimo elemento de un arreglo a: $\operatorname{array}[1..n]$ of nat mediante la técnica de programación $\operatorname{divide} y \operatorname{vencer\'as}$. Analizá la eficiencia de $\operatorname{minimo}(1,n)$.

```
\begin{aligned} &\mathbf{fun} \ minimo(a:\mathbf{array}[1..n] \ \mathbf{of} \ \mathbf{nat}, i, k:\mathbf{nat}) \ \ \mathbf{ret} \ m:\mathbf{nat} \\ &\mathbf{if} \ i=k \ \mathbf{then} \ m:=a[i] \\ &\mathbf{else} \\ &j:=(i+k) \ \mathbf{div} \ 2 \\ &m:=\min(\min(o(a,i,j),\min(a,j+1,k)) \\ &\mathbf{fi} \\ &\mathbf{end} \ \mathbf{fun} \end{aligned}
```

- 2. Dado un arreglo a: array[1..n] of nat se define una cima de a como un valor k en el intervalo $1, \ldots, n$ tal que a[1..k] está ordenado crecientemente y a[k..n] está ordenado decrecientemente.
 - (a) Escribí un algoritmo que determine si un arreglo dado tiene cima.
 - (b) Escribí un algoritmo que encuentre la cima de un arreglo dado (asumiendo que efectivamente tiene una cima) utilizando una búsqueda secuencial, desde el comienzo del arreglo hacia el final.
 - (c) Escribí un algoritmo que resuelva el mismo problema del inciso anterior utilizando la idea de búsqueda binaria.
 - (d) Calculá y compará el orden de complejidad de ambos algoritmos.
- 3. Calculá el orden de complejidad de los siguientes algoritmos:

```
\begin{array}{lll} \textbf{(a)} & \textbf{proc} \ fl(\textbf{in} \ n : \textbf{nat}) & \textbf{(b)} & \textbf{proc} \ f2(\textbf{in} \ n : \textbf{nat}) \\ & \textbf{if} \ n \leq 1 \ \textbf{then} \ \textbf{skip} & \textbf{for} \ i := 1 \ \textbf{to} \ n \ \textbf{do} \\ & \textbf{else} & \textbf{for} \ i := 1 \ \textbf{to} \ n \ \textbf{do} \ t := 1 \ \textbf{od} \\ & \textbf{for} \ i := 1 \ \textbf{to} \ n \ \textbf{do} \ t := 1 \ \textbf{od} \\ & \textbf{for} \ i := 1 \ \textbf{to} \ n \ \textbf{do} \ t := 1 \ \textbf{od} \\ & \textbf{for} \ i := 1 \ \textbf{to} \ n \ \textbf{do} \ t := 1 \ \textbf{od} \\ & \textbf{for} \ i := 1 \ \textbf{to} \ n \ \textbf{do} \ \textbf{do} \ n \ \textbf{do} \ n \ \textbf{do} \ n \ \textbf{do} \ n \ \textbf{do} \
```

ejercicios para el miércoles 27/3

- 4. Ordená utilizando \Box e \approx los órdenes de las siguientes funciones. No calcules límites, utilizá las propiedades algebraicas.
 - (a) $n \log 2^n$ $2^n \log n$ $n! \log n$ 2^n
 - (b) $n^4 + 2\log n$ $\log(n^{n^4})$ $2^{4\log n}$ 4^n $n^3\log n$
 - (c) $\log n!$ $n \log n$ $\log(n^n)$
- 5. Si $\lim_{n\to\infty} h(n) = \infty$, ¿cuáles de las siguientes afirmaciones resultan verdaderas?
 - (a) si $f(n) \sqsubset g(n)$, entonces $f(h(n)) \sqsubset g(h(n))$,
 - (b) si $f(h(n)) \sqsubset g(h(n))$, entonces $f(n) \sqsubset g(n)$,
 - (c) si $f(n) \sqsubset g(n)$, entonces $h(f(n)) \sqsubset h(g(n))$.

ejercicios adicionales

6. Dada la siguiente recurrencia

$$t(n) = \begin{cases} 4 & \text{si } n = 1\\ 2t(n/2) + 2n\log_2(n) & \text{si } n > 1 \end{cases}$$

Demostrá que t(n) es del orden de $nlog^2(n)$. Ayuda: primero demostralo para $n=2^m$.

7. Sean K y L constantes, y f el siguiente procedimiento:

```
\begin{aligned} \mathbf{proc} \ f(\mathbf{in} \ n : \mathbf{nat}) \\ \mathbf{if} \ n &\leq 1 \ \mathbf{then} \ \mathbf{skip} \\ \mathbf{else} \\ \mathbf{for} \ i &:= 1 \ \mathbf{to} \ K \ \mathbf{do} \ f(n \ \mathbf{div} \ L) \ \mathbf{od} \\ \mathbf{for} \ i &:= 1 \ \mathbf{to} \ n^4 \ \mathbf{do} \ operación\_de\_\mathcal{O}(1) \ \mathbf{od} \end{aligned}
```

Determiná posibles valores de K y L de manera que el procedimiento tenga orden:

(a) $n^4 \log n$

(b) n^4

(c) n^5

8. Escribí algoritmos cuyas complejidades sean (asumiendo que el lenguaje no tiene multiplicaciones ni logaritmos, o sea que no podés escribir for i:= 1 to $n^2 + 2 \log n$ do ... od):

- (a) $n^2 + 2 \log n$
- (b) $n^2 \log n$

- (c) 3^n
- 9. Una secuencia de valores $x_1, ..., x_n$ se dice que tiene orden cíclico si existe un i con $1 \le i \le n$ tal que $x_i < x_{i+1} < ... < x_n < x_1 < ... < x_{i-1}$. Por ejemplo, la secuencia 5, 6, 7, 8, 9, 1, 2, 3, 4 tiene orden cíclico (tomando i = 6).
 - (a) Escribí un algoritmo que determine si un arreglo almacena una secuencia de valores que tiene orden cíclico o no.
 - (b) Escribí un algoritmo que dado un arreglo a: $\operatorname{array}[1..n]$ of nat que almacena una secuencia de valores que tiene orden cíclico, realice una búsqueda secuencial en el mismo para encontrar la posición del menor elemento de la secuencia (es decir, la posición i).
 - (c) Escribí un algoritmo que resuelva el problema del inciso anterior utilizando la idea de búsqueda binaria.
 - (d) Calculá y compará el orden de complejidad de ambos algoritmos.
- 10. Calculá el orden de complejidad del siguiente algoritmo:

```
\begin{array}{l} \mathbf{proc} \ f\!3(n:\mathbf{nat}) \\ \mathbf{for} \ j:=1 \ \mathbf{to} \ 6 \ \mathbf{do} \\ \mathbf{if} \ n \leq 1 \ \mathbf{then} \ \mathbf{skip} \\ \mathbf{else} \\ \mathbf{for} \ i:=1 \ \mathbf{to} \ 3 \ \mathbf{do} \ f\!3(n \ \mathbf{div} \ 4) \ \mathbf{od} \\ \mathbf{for} \ i:=1 \ \mathbf{to} \ n^4 \ \mathbf{do} \ t:=1 \ \mathbf{od} \\ \mathbf{od} \end{array}
```