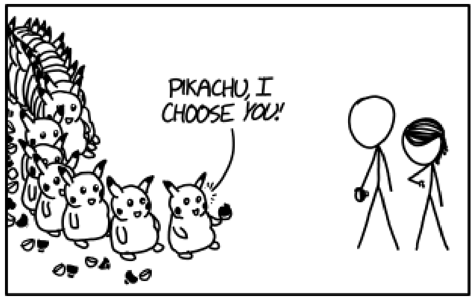
Primer Certamen
Algoritmos y Complejidad

1 de octubre de 2016



1. Obtenga el orden de convergencia del método de Newton si x^* es un cero doble, vale decir, $f(x^*) = f'(x^*) = 0$, $f''(x^*) \neq 0$.

   (20 puntos)

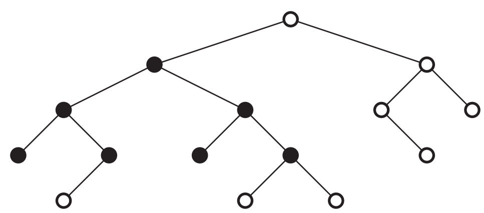


Figura 1: Un árbol binario con un máximo subárbol completo

2. Esboce un algoritmo recursivo que dado un árbol binario entregue la raíz y la altura de un máximo subárbol binario completo. En el ejemplo de la figura 1 uno de ellos está marcado en negro, su altura es 2.
   Pista: Considere un recorrido en postorden.

   (20 puntos)

3. La manera "obvia" de calcular el número de Fibonacci F_n es usar directamente la recurrencia:

   $$F_{n+2} = F_{n+1} + F_n \qquad F_0 = 0, F_1 = 1$$

   Explique cómo calcular cuántas llamadas a la función F se hacen para calcular F_n con esta estrategia.

   (20 puntos)

4. Esboce un algoritmo para enumerar las permutaciones de $[1, n]$ que son desarreglos, vale decir, tales que $\pi(i) \neq i$ para todo $i \in [1, n]$.

   (20 puntos)

5. Dado un arreglo a de n elementos, describa un algoritmo basado en programación dinámica para calcular la subsecuencia creciente más larga (LIS), la secuencia de índices $x_1, x_2, \ldots, x_k$ más larga tal que $a[x_1] \leq a[x_2] \leq \ldots \leq a[x_k]$. Por ejemplo, para $\langle 90, 10, 22, 9, 33, 21, 50, 41, 60, 80 \rangle$ la subsecuencia creciente más larga es $\langle 10, 22, 33, 50, 60, 80 \rangle$ (posiciones $2, 3, 5, 7, 9, 10$).

   a) Dé la recurrencia subyacente.
   b) Demuestre que se cumplen las suposiciones de programación dinámica.
   c) Esboce un algoritmo iterativo, indicando en particular el orden en que se llena la estructura.
   d) Indique cómo extraer la solución de su estructura.
   e) Indique la complejidad de su algoritmo.

   Pista: Considere la secuencia más larga que termina con $a[i]$.

   (35 puntos)

HvB/LaTeX 2_ε