Guía Precontrol 1

Profesores: Iván Sipiran
Nelson Baloian
Patricio Poblete

Auxiliares: Alonso Almendras, Albani Olivieri
Vicente Olivares, Ricardo Valdivia
Sebastián Acuña, Martín Paredes

P1. Mezclar listas enlazadas

Suponga que se dispone de la siguiente implementación de Nodo de una lista enlazada:

```
class Nodo:
    def __init__(self, valor, sgte):
        self.valor = valor
        self.sgte = sgte
```

Diseñe un algoritmo iterativo y escriba el método

```
mezclarListasOrdenadas(self, lista2)
```

Este se llamará desde una lista ordenada y recibirá como parámetro una segunda lista ordenada (ascendentemente por *valor*), retornará una lista que representará la mezcla ordenada de ambas listas. Defina claramente cuáles son los invariantes del problema, la inicialización, la condición de término, en qué parte del código se rompe el invariante, dónde se recupera, etc.

P2. Ecuaciones de Recurrencia

Resuelva las siguientes ecuaciones de recurrencia

a)

$$\begin{aligned} a_n &= 3a_{n-1} + 4a_{n-2} \\ a_0 &= 2 \\ a_1 &= 3 \end{aligned}$$

b)

$$f(n) = 2f(\sqrt{n}) + \log_2 n$$

Sugerencia: Haga el cambio de variable $n = 2^k$ y aplique el Teorema Maestro

c)

$$\begin{aligned} a_n &= 2a_{n-1} + a_{n-2} \\ a_0 &= 0 \\ a_1 &= 4 \end{aligned}$$

P3. Lista Rotada

Supongamos que tenemos una lista ordenada de un cierto largo n, por ejemplo:

$$[13, 20, 34, 41, 55, 62, 75, 84, 93]$$

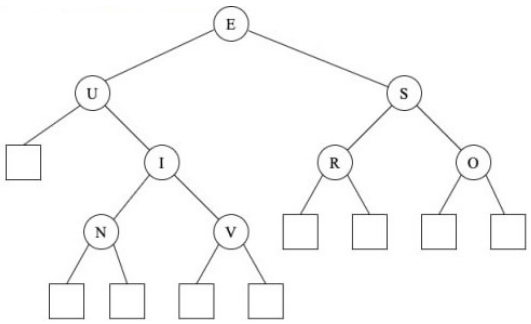
Si la desplazamos circularmente una cierta cantidad de posiciones hacia la derecha, lo que resulta se llama una lista rotada. Por ejemplo,

$$[75, 84, 93, 13, 20, 34, 41, 55, 62]$$

Escriba una función `encuentra_minimo(a)` que reciba como parámetro un arreglo a conteniendo una lista rotada, y que retorne el subíndice en donde se encuentra el mínimo elemento de la lista. Su función debe operar en tiempo $\Theta(\log n)$. Justifique por qué su algoritmo cumple con este requerimiento.

P4. Árbol de letras

Para el siguiente árbol:



a) Calcule su altura.

b) Indique en qué orden se visitarían los nodos según el tipo de recorrido.

1) Preorden:
2) Inorden:
3) Postorden:

P5. Matriz ascendente

Suponga que se tiene una matriz (tabla rectangular) a de $m \times n$, en que cada fila y cada columna están ordenadas ascendentemente. Por ejemplo:

$$\begin{pmatrix} 12 & 28 & 35 & 56 & 72 \\ 25 & 33 & 40 & 61 & 80 \\ 37 & 44 & 52 & 65 & 84 \\ 50 & 60 & 71 & 86 & 90 \end{pmatrix}$$

En esta matriz se desea buscar un número dado x

a) Suponga que se hace una búsqueda secuencial. ¿Cuánto demora el algoritmo en el peor caso?

b) Suponga que se hace búsqueda binaria en cada fila. ¿Cuánto demora en el peor caso?

c) Considere un nuevo algoritmo que comienza comparando x contra el elemento de la esquina inferior izquierda ($a_{m,1}$). Indique cuáles elementos podría descartar si $x < a_{m,1}$ y cuáles podría descartar si $x > a_{m,1}$. Si se itera esta idea, ¿cuánto demoraría el algoritmo en el peor caso?

P6. Particionando listas enlazadas como Quicksort

Se tiene una lista enlazada que se desea particionar al estilo Quicksort. Para esto, se debe tomar el primer elemento como pivote, y luego recorrer el resto de la lista construyendo dos listas enlazadas: una con los elementos menores que el pivote y la otra con los elementos mayores o iguales que el pivote. La función debe formar la listas de salida utilizando los mismos nodos de la listas de entrada, sin crear nuevos nodos. Por lo tanto, al terminar, la lista de entrada debe quedar vacía. Más concretamente, usted debe escribir una función particiona que reciba como parámetro una lista y retorne una tupla que tenga (lista de los menores, pivote, lista de los mayores), donde "pivote" es una lista de largo 1 que contiene solo al pivote.

P7. Filtrando numeros positivos

Suponga que se tiene una lista enlazada a con cabecera, que contiene números enteros. Escriba una función que se pueda invocar como"'a.filtrapositivos()", que al ejecutarse elimine de la lista todos los nodos que contienen números ≤ 0 y que deje solo los que contienen números positivos. Utilice las definiciones de Nodo y de Lista que aparecen en los apuntes.

P8. Ordenando una Pila

Suponga que tiene una pila S_1 que contiene n elementos. Genere un algoritmo que ordene los elementos en S_1 de tal forma que el más pequeño quede en el tope de la pila al terminar de ejecutarlo. Asuma que puede usar una segunda pila S_2 para almacenar elementos temporalmente. ¿Cuál es la complejidad temporal del algoritmo?