Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC 2133 – Estructuras de Datos y Algoritmos
Interrogación 2

Hora inicio: 14:00 viernes 4 de junio
Hora máxima de entrega: 23:59 viernes 4 de junio en SIDING

0. Formalidad. Responde esta pregunta en papel y lápiz, incluyendo tu firma al final. Entrega esta pregunta junto con cualquier otra pregunta.
   a. ¿Cuál es tu nombre completo?
   b. ¿Te comprometes a no preguntar ni responder dudas de la prueba a nadie que no sea parte del cuerpo docente del curso, ya sea de manera directa o indirecta?

Responda 3 de las siguientes preguntas:

1. Una cierta comuna muy afectada por la pandemia experimentó un inesperado aumento del número de miembros de su Club de Lectores, por lo que la municipalidad decidió contratarte para que desarrolles una aplicación para manejar en memoria principal la información de qué libros ha leído una persona y qué personas han leído un determinado libro. La idea es que la aplicación va a ser usada durante un año, más o menos. La municipalidad había pensado en usar una tabla, en que las filas representan a los libros y las columnas a las personas. Pero tú te diste cuenta de que desde el punto de vista del uso de memoria la tabla es ineficiente: si, en cifras redondas, al Club pertenecen 5,000 personas y el Club tiene 600 libros, la tabla tendría que tener $5{,}000 \times 600 = 3{,}000{,}000$ casillas; pero si una persona lee en promedio 6 libros al año, entonces en un mismo año sólo se estaría usando el 1% de las casillas. Describe una estructura de datos para resolver este problema más eficientemente que usando una tabla convencional.

   a. Primero, describe una estructura de datos basada en listas ligadas que sólo necesite ocupar una cantidad de memoria del orden de los miles de casillas ($5{,}000 \times 6 = 30{,}000$), y no de millones.

   b. Segundo, y si hacemos el supuesto simplificatorio de que las personas miembros del Club no cambian durante el año (es decir, no se agregan ni se eliminan), explica qué elemento adicional necesita tu estructura, y de qué tamaño, para que el usuario de la aplicación pueda especificar las personas por sus rut's. Explica también cómo funciona este elemento adicional y cómo se conecta al resto de la estructura.

   c. Tercero, y si además hacemos el supuesto simplificatorio de que los libros del Club tampoco cambian durante el año, explica qué otro elemento adicional necesita tu estructura, y de qué tamaño, para que el usuario pueda especificar los libros por sus códigos ISBN-13 (doce dígitos más un dígito verificador). Explica también cómo funciona este segundo elemento adicional y cómo se conecta al resto de la estructura.

      Dibuja la estructura de datos resultante de modo que ayude a entender tus respuestas a a), b) y c).

2. Considera el problema de asignar dígitos del 0 al 9 a las letras de tres strings de manera tal que se cumpla que la suma de los primeros dos strings de el tercero. Todas las ocurrencias de una letra deben asignarse al mismo dígito, y ningún dígito puede asignarse a más de una letra. A continuación se presentan dos problemas de ejemplo, con una visualización del problema solucionado y la solución correspondiente.

Problema	Vista Problema Solucionado	Solución
$\begin{array}{r} AABB \\ +\ CCDA \\ \hline BBED \end{array}$	$\begin{array}{r} 1133 \\ +\ 2241 \\ \hline 3374 \end{array}$	$A = 1$ $B = 3$ $C = 2$ $D = 4$ $E = 7$
$\begin{array}{r} BEEB \\ +\ XXXE \\ \hline EXZZZ \end{array}$	$\begin{array}{r} 9119 \\ +\ 8881 \\ \hline 18000 \end{array}$	$B = 9$ $E = 1$ $X = 8$ $Z = 0$

   a. Identifica las variables del problema, en general, y sus respectivos dominios.

   b. Explica con tus palabras una mejora a backtracking que se pueda aplicar al resolver este problema con backtracking. La mejora puede corresponder a Poda, Heurística o Propagación. Es importante que tu explicación muestre cómo aplica a este problema en particular.

   c. Responde b) nuevamente con otra mejora distinta. Nota: solo una de tus mejoras puede ser Propagación. No olvides explicar cómo aplica a este problema particular.

3. El *heap*, visto en clases, es una estructura de datos recursiva. El *heap*, en abstracto, es un árbol binario, aunque este árbol puede representarse completamente en un arreglo, tal como se vio en clases. Se llama *min-heap* cuando la propiedad de *heap* se aplica de manera tal de que un nodo padre debe ser siempre menor a sus dos hijos.

   a. Para los siguientes árboles binarios llenos (como se definieron en clases), representados como arreglos, dibuja su representación como árbol binario y marca en cada nodo si cumple con la propiedad de *min-heap*.
      i. [7]
      ii. [1, 4, 3, 2]
      iii. [9, 8, 5, 2, 1, 6]

   Considera la operación *sift down*, vista en clases, pero modificada para el caso *min-heap*.

   b. Dado un árbol binario, explica cómo usar *sift down* para restarurar la propiedad de *min-heap* del árbol, realizando la menor cantidad posible de llamados originales de *sift down* (un llamado original es un llamado que no corresponde a una recursión).

   c. Muestra con dibujos cómo usar tu explicación en b) para restaurar la propiedad de *min-heap* del árbol representado por el arreglo [27, 10, 30, 8, 18, 20, 35, 6, 7, 15].

4. Una página de Instagram hace un concurso de fotografías donde cada persona puede enviar todas las fotos que quiera, pero para seleccionar a los finalistas se consideran solo las k fotos con más "me gusta" de cada persona. Los finalistas se seleccionan como las f personas con mayor suma de me gustas en las k fotos seleccionadas.

   Si la página recibe m fotografías de n personas distintas, describe un algoritmo, y las estructuras de datos asociadas, para encontrar a los f finalistas en tiempo promedio

$$m \cdot log(k) \; + \; n \cdot \log(f)$$