Estructuras de Datos y de la Información

Ingeniería Técnica en Informática de Gestión. Curso 2007/2008

Ejercicios del Tema 4

1. Transforma a iterativo los siguiente algoritmos recursivos:
   (a) Función *log* del ejercicio 2 de la hoja 2.
   (b) Procedimiento *dosFib* del ejercicio 5 de la hoja 2.
   (c) Función del ejercicio 17 de la hoja 2.

   Aplica en cada caso la técnica de transformación que conlleve un menor consumo de espacio en el algoritmo iterativo.

2. Una frase se llama palíndroma si la sucesión de caracteres obtenida al recorrerla de izquierda a derecha (ignorando los blancos) es la misma que si el recorrido se hace de derecha a izquierda. Esto sucede, por ejemplo, con la socorrida frase "dábale arroz a la zorra el abad". Construye una función iterativa ejecutable en tiempo lineal, que decida si una frase dada en forma de *cola de caracteres* es o no palíndroma. El algoritmo utilizará una *pila de caracteres*, declarada localmente.

3. El agente 0069 ha inventado un nuevo método de codificación de mensajes secretos. El mensaje original X se codifica en dos etapas:
   – Primero, X se transforma en X' reemplazando cada sucesión de caracteres consecutivos que no sean vocales por su imagen especular.
   – A continuación, X' se transforma en la sucesión de caracteres X" obtenida al ir tomando sucesivamente: el primer carácter de X'; luego el último; luego el segundo; luego el penúltimo; etc.

   Ejemplo: para X = "*Bond, James Bond*", resultan:

   X' = "*BoJ ,dnameB sodn*" y X" = "*BnodJo s, dBneam*"

   Construye los algoritmos de codificación y descodificación de mensajes y analiza su complejidad, utilizando pilas y colas. Supón que el mensaje inicial viene dado como una cola de caracteres.

4. Añade las siguientes operaciones a la clase *TListaDinamica*<*TElem*>:
   – ==: comparación sobre igualdad entre listas
   – <=: comparación de orden entre listas
   – *ordenada*: determina si una lista está ordenada
   – *insertaOrd*: inserta un elemento delante del primero que es mayor que él.

5. Resuelve los problemas que siguen aplicando el esquema de recorrido de secuencias. En cada caso, el contenido de la secuencia debe quedar inalterado.
   (a) Contar el número de apariciones de 'a' en una secuencia de caracteres dada.
   (b) Dada una secuencia de enteros, contar cuántas posiciones hay en ella tales que el entero que aparece en esa posición es igual a la suma de todos los precedentes.

6. Resuelve los problemas que siguen aplicando el esquema de búsqueda secuencial.
   (a) Buscar la primera aparición de 'b' en una secuencia de caracteres dada.
   (b) Dada una secuencia de enteros, buscar la primera posición ocupada por un número que sea menor que todos los anteriores.

7. Algunos problemas de procesamiento de secuencias requieren modificar y/o combinar los esquemas de recorrido y búsqueda secuencial. Resuelve los casos siguientes:
   (a) Dada una secuencia de caracteres, copiarla en otra eliminando los blancos múltiples.
   (b) Contar el número de caracteres anteriores y posteriores a la primera aparición de 'a' en una secuencia de caracteres dada.
   (c) Contar el número de parejas de vocales consecutivas que aparecen en una secuencia de caracteres dada.

8. Una expresión aritmética construida con los operadores binarios '+', '–', '*', '/' y operandos (representados cada uno por un solo carácter) se dice que está en forma *postfija* si es o bien un solo operando o dos expresiones en forma postfija una tras otra, seguidas inmediatamente de un operador. Lo que sigue es un ejemplo de una expresión escrita en la notación infija habitual, junto con su forma postfija:

   Forma infija: `(A/(B-C))*(D+E))` Forma postfija: `ABC-/DE+*`

   Diseña un algoritmo iterativo que calcule el valor de una expresión dada en forma postfija por el siguiente método: se inicializa una pila vacía de números y se van recorriendo de izquierda a derecha los caracteres de la expresión. Cada vez que se pasa por un operando, se apila su valor. Cada vez que se pasa por un operador, se desapilan los dos números más altos de la pila, se componen con el operador, y se apila el resultado. Al acabar el proceso, la pila contiene un solo número, que es el valor de la expresión. Representa la expresión dada como secuencia de caracteres, y supón disponible una función *valor* que asocie a cada operando su valor numérico.

9. Dado un número natural N ≥ 2, se llaman *números afortunados* a los que resultan de ejecutar el siguiente proceso: se comienza generando una *cola* que contiene los números desde 1 hasta *N*, en este orden; se elimina de la cola un número de cada 2 (es decir, los números 1, 3, 5, etc.); de la nueva cola, se elimina ahora un número de cada 3; etc. El proceso termina cuando se va a eliminar un número de cada *m* y el tamaño de la cola es menor que *m*. Los números que queden en la cola en este momento son los afortunados. Diseña un procedimiento que reciba *N* como parámetro y produzca una secuencia formada por los números afortunados resultantes.

   (*Indicación*: para eliminar de una cola de *n* números un número de cada *m*, hay que reiterar *n* veces el siguiente proceso: extraer el primer número de la cola, y añadirlo al final de la misma, salvo si le tocaba ser eliminado.)

10. Añade las siguientes operaciones a la clase *TSecuenciaDinamica<TElem>*:
   – *busca*: operación que busca una secuencia *s'* dentro de otra secuencia *s*. La búsqueda se realiza en *s*, a partir de la posición actual del punto de interés, y los elementos a buscar son aquellos que aparecen a la derecha del punto de interés de *s'*. Esta operación modifica el punto de interés de *s*, colocándolo delante de la primera aparición de la subsecuencia buscada, o al final del todo si la subsecuencia no se ha encontrado. La operación no está definida si el punto de interés de *s'* está a la derecha del todo –no hay nada que buscar–.

     Por ejemplo, si trabajásemos con secuencias de caracteres:

     Entrada:

     ```
     s  : Una casa al borde del mar, un gran mar, en la playa
     s' : La mar
     ```

     Salida:

     ```
     s  : Una casa al borde del mar, un gran mar, en la playa
     s' : La mar
     ```

   – *borraSec*: operación que borra todas las apariciones de una secuencia *s'* dentro de otra secuencia *s*. La búsqueda de los elementos a eliminar se realiza de la misma forma que en la operación *busca*, es decir, se busca en *s* a partir de la posición del punto de interés, y la subsecuencia a eliminar es la compuesta por los elementos de *s'* a la derecha del punto de interés. Igualmente, la operación no está definida si el punto de interés de *s'* se encuentra al final.

11. Implementa el TAD secuencia utilizando como tipo representante una par de listas, según las indicaciones vistas en clase. Para mejorar la eficiencia de algunas operaciones, añade a la implementación de las listas una operación de concatenación por la izquierda.