

Doble Grado en Ingeniería Informática y Matemáticas

Relaciones de Estructuras de Datos.

TDA Lineales

J. Fdez-Valdivia

*Departamento de Ciencias de la Computación e I.A. ETS Ingeniería Informática
Universidad de Granada. 18071 Granada. Spain.
Email: jfv@decsai.ugr.es*

Resumen

En este documento se presenta relación de problemas sobre TDA Lineales que se propone para la asignatura de estructura de datos de segundo curso del Doble Grado en *Ingeniería Informática y Matemáticas*.

1. TDA Lineales

Problema 1.1

Diseñar funciones para trabajar con listas de enteros capaces de realizar las operaciones siguientes:

- *Invertir la lista.*
- *Formar una lista que contenga los elementos comunes de otras dos.*
- *Insertar un elemento después del elemento i -ésimo.*
- *Ordenar los elementos en orden creciente.*
- *Calcular la media y la desviación típica de los elementos de la lista.*
- *Mover un nodo j , n posiciones más hacia adelante.*

Problema 1.2 *Diseñar tres funciones que eliminen todas las apariciones de un elemento x en una lista, una pila y una cola respectivamente.*

Problema 1.3 *Diseñar una función swap que intercambie los contenidos de dos listas. Implementarlo mediante una función independiente y como función miembro (o función amiga) de listas. ¿Qué eficiencia tienen?*

Problema 1.4 *Un palíndromo es una cadena de caracteres que se lee igual hacia delante que hacia atrás (pudiéndose modificar los espacios en blanco). Por ejemplo: Dábale arroz a la zorra el abad. Escribir un programa para detectar si una cadena de caracteres es un palíndromo o no. (Nota: un mecanismo efectivo de solución consiste en el uso simultáneo de una pila y una cola).*

Problema 1.5 *La notación infija es la manera habitual de representar las expresiones aritméticas, donde el operador se coloca entre los operandos (p.e., " $a + b$ "). Esta notación tiene el inconveniente de que obliga a usar paréntesis para determinar el orden de realización de las operaciones. Otro tipo de notación es la postfija, en donde los operadores se colocan después de los operandos, y no entre ellos (p.e., " $a b +$ " representa la suma de " a " y " b "). La notación postfija tiene la ventaja de que el orden en que se deben realizar las operaciones está completamente determinado por las posiciones de los operadores y los operandos, y nunca es necesario el uso de paréntesis. Diseñar programas para:*

-
1. Evaluar el resultado de una expresión aritmética escrita en notación postfija. Por simplicidad, suponed que los operadores empleados sólo pueden ser “+”, “-”, “*” y “/”, y los números son siempre enteros.
 2. Transformar una expresión aritmética escrita en notación infija a notación postfija, con las mismas restricciones del apartado anterior, pero permitiendo la utilización de variables en lugar de constantes enteras.

Problema 1.6 Implementar el tipo de dato Pila utilizando como representación el tipo de dato abstracto Lista.

Problema 1.7 Implementar el tipo de dato Cola utilizando como representación el tipo de dato abstracto Lista.

Problema 1.8 Modifique la implementación del tipo cola, de forma que use dos pilas para almacenar los elementos que se insertan y eliminan. Las inserciones se realizan sobre la primera y los borrados sobre la segunda. En caso de que la segunda no tenga elementos, se pasan los elementos de la primera. ¿Cuál es la eficiencia de la operación pop?

Problema 1.9 Implementar el tipo de dato Cola con prioridad utilizando como representación el tipo de dato abstracto Lista.

Problema 1.10 Suponga una cola de valores enteros. Implemente una función que elimine todos los elementos repetidos consecutivos (si hay una secuencia de elementos repetidos sólo se mantiene uno de ellos).

Problema 1.11 Implementar una función, que recibe una lista de enteros L y un número entero n, y modifica la lista eliminando todas las apariciones de ese número entero.

Problema 1.12 Implemente una función que recibe como entrada una lista de enteros y devuelve como salida dos listas que contienen, respectivamente, los pares e impares de la primera.

Problema 1.13 Implementar una función Reemplazar que tenga como argumentos una pila con tipo de elemento int y dos valores (nuevo y viejo) de forma que si el segundo valor aparece en algún lugar de la pila, sea reemplazado por el primero. Realizar dos versiones:

1. Resolviendo el problema haciendo uso de una pila auxiliar donde ir almacenando los valores que se sacan para luego restaurarlos.
2. Sin usar una pila auxiliar, haciendo uso de la recursividad.

Problema 1.14 Implemente una función Eliminar que tenga como argumentos una cola con tipo de elemento string y un valor de este tipo, de manera que elimine la primera aparición de ese elemento en la cola. Realice dos versiones:

1. Usando una cola auxiliar.
2. Considerando la inserción de un elemento centinela que indica el final.

Problema 1.15 Implementar una función Mezcla2 que tenga como parámetros dos listas de valores reales ordenados de menor a mayor y que devuelva una nueva lista como unión de ambas con sus elementos ordenados de la misma forma. Las listas originales no se deben de modificar.

Problema 1.16 Considerar un vector de listas de números reales ordenados. Implementar una función Mezcla que reciba como parámetros ese vector (un puntero al tipo Lista y un entero indicando el número de listas), y devuelva como resultado una lista que contenga todos los elementos ordenados. Para ello, utilizar la función de mezcla del ejercicio anterior.

Problema 1.17 Considere un nuevo tipo de dato Cola Doble. Su funcionamiento es parecido al tipo de dato Cola, pero en este caso se permite realizar inserciones, borrados, y consultas en ambos extremos.

-
1. Diseñe este nuevo tipo de dato con las funciones básicas de forma similar a las del tipo de dato Cola.
 2. Considere distintas implementaciones. ¿Cuáles cree que son más adecuadas? Escoja una e implémentelo.

Problema 1.18 Considere una lista de números reales. Implemente una función que reciba una lista y dos posiciones p, q en ella y obtenga como resultado la suma de los elementos que hay entre ellas (incluyendo el elemento de p y excluyendo el de q , es decir, los del intervalo semiabierto $[a_p, a_q)$).

Problema 1.19 Considere una lista que almacena valores reales. Implementar una función que devuelva el número de elementos de la lista, la media y la desviación típica.

Problema 1.20 Implementar una función que recibe dos listas, l_1 y l_2 , y devuelve la posición en l_1 que corresponde al comienzo de la lista l_2 , o end en caso de que no sea sublista.

Problema 1.21 Implemente una función que, dado un entero i , suprima el elemento i -ésimo de una lista.

Problema 1.22 Considere listas de enteros. Implemente una función para ordenar una lista usando el algoritmo de selección.

Problema 1.23 Considere el tipo de dato Lista implementado con celdas doblemente enlazadas circulares.

1. Implemente una nueva función miembro swap que toma como entrada dos posiciones en la lista y como efecto, intercambia los dos elementos correspondientes. Como restricción, deberá realizarla de forma que no se copien elementos de la lista, sino que sólo se reasignen punteros.
2. Implemente una función miembro operator + que devuelva una nueva lista que contiene la suma de las dos de entrada. Las dos originales quedan sin modificar.
3. Implemente una función miembro unir que tome como parámetro una lista y que como resultado, la lista *this se expande con los elementos de ese parámetro. La lista que se une queda vacía. Como restricción, la función deberá implementarse en tiempo $O(1)$.
4. Implemente una función miembro swap que tome como parámetro otra lista. Como resultado se intercambia el contenido de *this y la lista que se pasa como parámetro.
5. Implemente la función anterior como función externa a la clase.

Problema 1.24 Considere una matriz de de tamaño n por m que representa un laberinto. Para ello, almacena elementos de tipo char de forma que en la matriz aparecen

1. Un caracter 'E'. Indica la entrada.
2. Un caracter 'S'. Indica la salida.
3. Caracteres 'P' (Pared). Indican que no se puede pasar.
4. Caracteres ' ' (Vacío). Se puede pasar.

Si comenzamos desde la casilla 'S' y sólo podemos realizar movimientos horizontales y verticales, desarrollar un algoritmo para devolver el camino desde la entrada a la salida, o un camino vacío en caso de que no exista.

Problema 1.25 Diseñar (especificar e implementar) un tipo de dato abstracto Entero Largo, que permita realizar operaciones aritméticas con precisión arbitraria¹. Para ello, considere que un número entero se puede representar como una secuencia de elementos. Implemente como ejemplos las operaciones de menor, mayor, igual, distinto, menor o igual, mayor o igual, suma y resta.

¹Limitada por la cantidad de memoria de la máquina.