Análisis de Algoritmos y Estructuras de Datos

Práctica 6: Uso del TAD Cola

Versión 2.1

Pasos a seguir

1. Implemente el TAD *Cola* utilizando las representaciones pseudoestática (vector circular) y dinámica.
2. Escriba módulos que contengan las implementaciones de los subprogramas demandados en cada problema.
3. Para cada uno de los problemas, escriba un programa de prueba donde se realicen las llamadas a los subprogramas, comprobando el resultado de salida.

Ejercicios

1. Se dice que una pila es isomórfica a una cola cuando los elementos situados en posiciones pares de la pila coinciden con los situados en posiciones impares de la cola.

Usando los TAD *Pila* y *Cola*, implemente una función que determine y devuelva si una pila y una cola dadas son isomórficas o no.

2. Escriba una función que dada una cola y dos elementos a y b, invierta la secuencia delimitada por ambos dentro de ella.

Dado que la cola puede tener elementos repetidos, se invertirá la secuencia delimitada por la primera ocurrencia de a y b.

Ejemplos: a=1, b=2

Cola original:	47899**18679624**893210
Cola final:	47899**26976814**893210
Cola original:	342342**134219**102
Cola final:	342342**243119**102
Cola original:	34**13413424**223
Cola final:	34**24314314**223

3. Se define una *bicola* o *cola doble* como una cola en la cual se pueden hacer inserciones y borrados en ambos extremos.

Realice la especificación e implementación (mediante una estructura enlazada) del TAD *Bicola*.

4. Implemente el TAD *Pila* utilizando como representación interna una bicola. Repita el ejercicio para el TAD *Cola*.

5. El estacionamiento de las avionetas en un aeródromo es en línea, con capacidad para 12 avionetas. Las avionetas llegan por el extremo izquierdo y salen por el derecho. Cuando llega un piloto a recoger su avioneta, si ésta no está justamente en el extremo de salida (derecho), todas las avionetas a su derecha han de ser retiradas, sacar la suya y las retiradas colocadas de nuevo en el mismo orden relativo en que estaban. La salida de una avioneta supone que las demás se mueven hacia adelante, de tal forma que los espacios libres del estacionamiento estén en la parte izquierda (entrada).

Implemente un subprograma que emule el estacionamiento. Este subprograma, además del aeródromo, recibirá un código de acción sobre una avioneta, y la matrícula de la misma. La acción puede ser entrada (E) o salida (S) de avioneta. En la llegada puede ocurrir que el estacionamiento esté lleno; si es así, la avioneta espera hasta que se quede una plaza libre o hasta que se le dé la orden de retirada (salida).

6. Se desea simular el juego de los reyes. Se trata de un solitario que se desarrolla de la siguiente forma:

Se reparten las cartas de la baraja española en diez montones de cuatro cartas cada uno. Cada montón se corresponde con una de las diez figuras de la baraja. Se escoge uno de los montones al azar y se extrae la carta de dicho montón que esté en la cima. Esta carta se coloca al final del montón correspondiente a la figura que represente, extrayéndose ahora la primera carta de este montón. Este ciclo se repite hasta llegar a un montón en el que las cuatro cartas de la figura correspondiente han sido colocadas.

El juego se considera acabado con éxito si los cuatro reyes están en su montón.

Se pide:

a) Diseñar las estructuras de datos adecuadas para representar los tipos `tCarta` y `tMonton`.

b) Suponiendo que las cartas ya están barajadas y que el resultado se encuentra almacenado en un vector de tipo `tCarta`, desarrollar una función que dado el montón de comienzo, simule este juego y devuelva, a su finalización, el resultado del solitario (éxito o fracaso) y el contenido del montón correspondiente a los reyes.

Nota: La baraja española se compone de un total de 40 cartas repartidas en diez figuras (`AS, DOS, TRES, CUATRO, CINCO, SEIS, SIETE, SOTA, CABALLO, REY`) y cuatro palos (`OROS, COPAS, ESPADAS, BASTOS`).