Memoria PECL1

Estructuras de Datos

Pedro Barquín Ayuso

Wasim El Hallak Díez

3º GIC

Curso 2015/2016

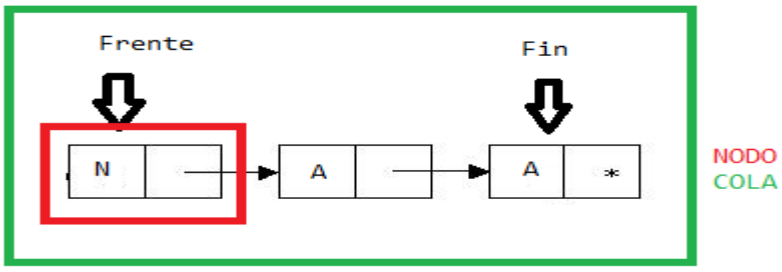
Descripción TAD's implementados

Cola:

```
class Nodo
{
private:
        char valor;
        Nodo *siguiente;
        friend class Cola;
public:
        Nodo(char v, Nodo *sig = NULL)
        {
                valor = v;
                siguiente = sig;
        }
};
typedef Nodo *pnodo;
class Cola
{
public:
        Cola(int a) :fin(NULL), frente(NULL), numCola(a) {}
        ~Cola();
        void encolar(char v);                           char desencolar();
        bool vacia();
        char mostrarPrimero();
        int tamanio();
        void mostrar();

private:
        pnodo frente, fin;
        int numCola;
};
```

La clase cola es un TAD donde se van a almacenar los nodos (TAD) en el que estarán los esquiadores (los cuales son otro TAD) o en nuestro caso la información sobre si son adultos o niños, los nodos que son los que contienen la información del esquiador char (A=Adulto N=niño) y un puntero que apunta a la dirección del próximo nodo cuando se añada y se asigne a la cola, estos nodos están interconectados entre ellos y están añadidos a la cola, para el acceso a los nodos desde la cola contamos con los pnodos que nos sirven para poder acceder a los nodos que hemos encolado en la clase cola conociendo cuál es el primero(frente) y el último (Fin) además cuando se crea una cola se le pasa un número numCola el cual será el número de cola que es para después poder mostrarlo adecuadamente.
En la imagen se representa de forma esquemática lo que serían los TAD nodo y cola y como estarían conectados.



ColaTelesilla:

```
class NodoTelesilla
{
private:
        char valor[6];
        NodoTelesilla *siguiente;
        friend class ColaTelesilla;

public:
        NodoTelesilla(char v[6], NodoTelesilla *sig = NULL)
        {
                valor[0] = v[0];
                valor[1] = v[1];
                valor[2] = v[2];
                valor[3] = v[3];
                valor[4] = v[4];
                siguiente = sig;
        }
};
typedef NodoTelesilla *pnodoTelesilla;

class ColaTelesilla
{
public:
        ColaTelesilla() :fin(NULL), frente(NULL) ,elementos(0){}
        ~ColaTelesilla();
        void encolarTelesilla(char v[6]);
        void desencolarTelesilla(int bajar);
        bool vaciaTelesilla();
        void mostrarPrimeroTelesilla();
        int tamanioTelesilla();
        void mostrarTelesilla();

private:
        pnodoTelesilla frente, fin;
        int elementos;
};
```

En este apartado la idea es la misma que la del apartado anterior contamos con los TAD que son NodoTelesilla y ColaTelesilla, la finalidad de estos TAD es la misma que la anterior, ColaTelesilla será donde vamos encolando los NodoTelesilla, que son los que contienen la información que queremos guardar, la cual en este caso es un char[6] con la información de la categoría de los 5 esquiadores extraídos de las colas individuales y guardados en orden en el char[], además incluye el puntero que apunta a los otros nodos que habrá cuando se creen y se añadan a la cola. Una cosa nueva en este apartado es que sólo podrá haber 10 nodos en la cola ya que es un requisito de la práctica y por ello se añade una variable elementos que es la cual nos permite controlar de forma fácil cuántos hay en cada instante y permitir o no la adición de más elementos a la cola. El resto es igual a la anterior contamos con los pnodotelesilla frente y fin los cuales nos permiten controlar la cola al igual que en la cola normal. Su esquema sería similar al expuesto en el apartado Cola.

Esquiador:

```
#ifndef ESQUIADOR_H
#define ESQUIADOR_H

#include<iostream>

class Esquiador
{
public:
        Esquiador();
        ~Esquiador();
        char categoria();
private:
        char edad = 'Z';//A para adulto N para niño
        int grupo = 0;  //1=A 0=N
};

#endif;
```

El último TAD es un objeto el cual en un principio íbamos a utilizar para ser el encolado en la cola de los esquiadores, pero al ser más complejo a la hora de mostrar la información se optó por la opción de usarlo para la creación de los esquiadores como tal pero luego sólo se almacena en la cola la información de la persona que ha entrado en esa cola y en esa posición, por lo que sólo lo usamos en la creación y asignación aleatoria de su categoría la cual será la que se almacenará en las colas individuales y que luego pasará a la cola telesilla. Su característica especial es que al crearse se le asigna aleatoriamente una categoría A o N.

Definición de operaciones de los TAD's

Cola:

-Creación y destrucción: Se crea la cola pasando los parámetros inicializando los valores de la cola y poniendo frente y fin a null lo cual nos dice que está vacía, en el caso del destructor elimina la cola.

-encolar: Recibe un char el cual es la información del esquiador a encolar, crea un nuevo nodo con la información pasada y en función de si la cola estaba vacía o no actualiza la información de los pnodos, caso vacío asigna que frente y fin es el nuevo nodo creado, en caso de contar con más elementos asigna al fin->siguiente el nuevo nodo y después dice que el nuevo nodo es el fin,

-desencolar: Va eliminando y extrayendo la información antes de eso a los nodos de la cola, para ello crea un nodo auxiliar al cual iguala con frente y una vez realizado este paso asigna que ahora frente es el próximo nodo y extrae la información del nodo en cuestión y guardándola para retornarla al finalizar la función, después se elimina el nodo aux y perdemos la referencia al nodo por lo que ese nodo que se ha "extraído" ya es inalcanzable.

-vacía: Comprueba que la cola esta vacía mirando si el frente está a null y en función de la situación retorna un valor u otro.

-mostrarPrimero: Con el método del nodo aux igualamos el pnodo a frente y de él ya podemos extraer la información del char que buscamos con nodo->valor.

-tamanio: Con el método del nodo aux vamos recorriendo desde frente hasta el fin pasando por los nodos que forman la cola y cada vez que pasamos uno añadimos 1 al contador que es el valor que retornamos al final de la ejecución de este método.

-mostrar: Con el método del nodo aux recorremos de igual manera que en el método tamaño la cola lo que pasa que en este caso mostramos la información mientras que en el otro sólo se incrementaba la variable al pasar.

ColaTelesilla:

-Creación y destrucción: Creación y destrucción: Se crea la cola pasando los parámetros inicializando los valores de la cola y poniendo frente y fin a null lo cual nos dice que esta vacía, en el caso del destructor elimina la cola.

-encolarTelesilla: Recibe un char[6] el cual es la información de los esquiadores a encolar, crea un nuevo nodotelesilla con la información pasada y en función de si la cola estaba vacía o no actualiza la información de los pnodostelesilla, caso vacío asigna que frente y fin es el nuevo nodotelesilla creado, en caso de contar con más elementos asigna al fin->siguiente el nuevo nodotelesilla y después dice que el nuevo nodotelesilla es el fin, También hay un límite para no poder encolar más de diez elementos ya que es un requisito de la práctica, incrementa el valor de los elementos en 1.

-~desencolarTelesilla: Va eliminando y mostrando la información antes de eso a los nodostelesilla de la cola, para ello crea un nodotelesilla auxiliar el cual iguala con frente y una vez realizado este paso asigna que ahora frente es el próximo nodotelesilla y muestra la información del nodotelesilla, después se elimina el nodo aux y perdemos la referencia al nodo por lo que ese nodo que se ha "extraído" ya es inalcanzable, decremento del valor de los elementos en 1 al final.

-vaciaTelesilla: Comprueba que la cola está vacía mirando si el frente está a null y en función de la situación retorna un valor u otro.

-mostrarPrimeroTelesilla: En este caso se ha optado por el método alternativo al anterior que es directamente imprimir el valor del pnodotelesilla que está al frente que es el que tiene la información del primero.

-tamanioTelesilla: Retorna el valor de la variable elementos de la colatelesilla.

-mostrarTelesilla: Con el método del nodo aux recorremos de igual manera que en el método tamaño la cola lo que pasa que en este caso mostramos la información mientras que en el otro sólo se incrementaba la variable al pasar.

Esquiador:

-categoría: Con este método retornamos el valor de la categoría A=adulto o N=Niño del esquiador que lo está invocándo.

Explicación funcionamiento del programa y métodos más importantes

El funcionamiento principal del programa es que se han de crear 5 colas en las cuales se encolaran 10 esquiadores (esquiadores en nuestro caso encolamos el tipo de esquiador que es no el propio esquiador tal y como se ha explicado arriba), los esquiadores son o adultos o niños y han de ser asignados de manera aleatoria, una vez realizado este paso hemos de transferir de las 5 colas a una de las 10 góndolas (colatelesilla) para ello hemos de extraer a los esquiadores de las colas y agruparlo para poder añadirlos a la colatelesilla, después de estar en la colatelesilla se irán desencolando de ésta y volveremos a añadir desde las colas repitiendo este proceso hasta terminar con todos los esquiadores en las dos colas (cola, colatelesilla).

Importante: Se ha hecho global la declaración de las colas individuales y la colatelesilla ya que al realizar operaciones con ellas en alguna de las funciones mostradas a continuación sucedía que se perdía la referencia y ocurrían errores y para simplificar el modo de hacer las funciones se ha optado por esta opción.

Métodos MAIN:

```
void mostrarColas() {
        cout << "Estado actual de las colas: " << endl;
        cola1.mostrar();
        cola2.mostrar();
        cola3.mostrar();
        cola4.mostrar();
        cola5.mostrar();
}
```

Este método lo único que hace es llamar a la función mostrar cola de cada una de las 5 colas y las presenta completas por pantalla.

```
void crearMontar(int total) {
        int caso, veces = 0;
        srand(time(NULL));

        cout << "Se crean y adjudica a cada esquiador una fila" << endl;

        for (size_t i = 0; i < total; i++)
        {
                Esquiador esquiador;
                switch (i % 5)
                {
                case 0: cola1.encolar(esquiador.categoria()); break;
                case 1: cola2.encolar(esquiador.categoria()); break;
                case 2: cola3.encolar(esquiador.categoria()); break;
                case 3: cola4.encolar(esquiador.categoria()); break;
                case 4: cola5.encolar(esquiador.categoria()); break;
                default:esquiador.~Esquiador(); break;
                }
        }
}
```

Este método recibe el número de esquiadores que ha de crear y en función de eso crea cierto número de ellos (al crearlos se auto asigna una categoría) y una vez creados los va almacenando en las colas con el método encolar el cual ya se ha descrito.

```
void montar(int num) {
          int veces = 0;
          cout << "Enpieza el traspaso de los esquiadores a sus gondolas: " << endl;

                    for (size_t i = 0; i < num;)                         {
                              if (cola1.vacia() || cola2.vacia() || cola3.vacia() || cola4.vacia() || cola5.vacia())
                              {
                              cout << endl;
                              cout << "Error al intentar sacar de alguna de las filas por encontrarse vacias" << endl;
                              cout << endl;
                              break;
                              }
                              else if (cola1.mostrarPrimero() == 'N' && cola2.mostrarPrimero() == 'N' &&
cola3.mostrarPrimero() == 'N' && cola4.mostrarPrimero() == 'N' && cola5.mostrarPrimero() == 'N' && veces<3)
                                        {
                                        cout << endl;
                                        cout << "No pueden montar 5 ninios por lo que los ninios 1 y 3 pasan al final de la
cola" << endl;
                                        cola1.encolar(cola1.desencolar());
                                        cola3.encolar(cola3.desencolar());
                                        mostrarColas();
                                        cout << endl;
                                        cout << "Continuamos con la ejecucion" << endl;
                                        cout << endl;
                                        veces++;
                              }
                              else                                   {
                                        if (Telesilla.tamanioTelesilla() == 10 )
          {
                                                  cout << endl;
                                                  cout << "Error al intentar añadir al telesilla ya que esta completo" <<
endl;
                                                  cout << endl;
                                                  break;
                                        }
                                        else                                                  {
                                                  char auzz[6] = "     ";
                                                  auzz[0] = cola1.desencolar();
                                                  auzz[1] = cola2.desencolar();
                                                  auzz[2] = cola3.desencolar();
                                                  auzz[3] = cola4.desencolar();
                                                  auzz[4] = cola5.desencolar();
                                                  printf("Extraemos a los esquiadores: %s\n", auzz);
                                                  Telesilla.encolarTelesilla(auzz);
                                                  i++;
                                        }
                              }
                    }
}
```

Este método es el encargado de extraer a los “esquiadores” de las colas y de encolarlos en la colatelesilla, para ello recibe un valor que es el número de veces que ha de hacer este proceso. Para ello antes de nada hemos de comprobar que las colas no están vacías de ser así saldremos de la ejecución avisando del error, En caso de estar correcto este paso comprobaremos que los 5 primero “esquiadores” de cada cola no son niños y en caso de ser así reubicamos al niño 1 y 3 al final de la cola para evitar que suban 5 niños hasta que se cumpla la condición, también se ha añadido un contador especial para permitir excepcionalmente que monten 5 niños en el caso de que sólo queden 5 y todos sean niños. Una vez pasados estos filtros comprobamos que el telesilla tiene huecos libre y de no ser así avisamos con un mensaje, por último habiendo pasado todos los filtros anteriores procedemos a la extracción de los “esquiadores” y a su almacenado en un char[6] el cual es el que pasaremos a la hora de encolar el grupo de “esquiadores” en la colatelesilla para así guardar la referencia en el orden en que han sido extraídos.

Problemas encontrados y solución adoptada

-Decisión de qué estructura elegir para la sucesión de diez góndolas seguidas (lista enlazada, array, cola…) al final se ha optado por una cola similar a la cola normal pero en este caso el valor a guardar es un char[6] con los 5 datos de los 5 esquiadores.

-Cadenas de caracteres, problemas y estructura elegida para almacenar los cinco ocupantes en cada góndola, ya que a la hora de transferir la información aparecieron problemas en función de si usábamos un char normal y luego se agrupaban o si por el contrario agrupábamos y después pasábamos.

-A la hora de realizar las operaciones con las colas en las funciones del main al pasarlas y operar con ellas se perdía la referencia a las propias colas por lo que para simplificar se ha optado por hacerlas globales arreglando así el error que ocurría.