Pilas y Colas

Autores: Enrique Antonio Vallejo Sanz – DNI: 03203024-K
Grado Ingeniería de Computadores – Laboratorio Viernes 12.00-14.00

- Detalles de la práctica

El objetivo de esta práctica es demostrar los conocimientos adquiridos en relación con las Estructuras de Datos Dinámicas: Pilas y Colas.

Hay que desarrollar una aplicación utilizando dichas estructuras para monitorizar un sistema de góndolas de una pista de esquí.

- **TADs utilizados**

Para realizar esta práctica he implementado dos tipos de TAD, una Pila y dos Colas. Para ello me hemos apoyado en dos clases intermedias, a las que he llamado "Nodo" y "Gondola". Tanto el nodo como la gondola contiene tanto el valor de lo que estamos almacenando en el TAD como la señalización al siguiente elemento de la estructura.
Entrando un poco más en detalles, la Pila está formada por las siguientes operaciones:

- Constructor: Construye una pila vacía.

- apilar(e: Persona)
  Introduce un elemento del tipo Persona (explicado más adelante) en la Pila detrás del último elemento que se hubiera introducido. El nuevo elemento pasa a ser la cima de la pila.

- desapilar(): Extrae el elemento situado en la cima de la Pila, esto es, el último elemento introducido en la Pila. La cima de la Pila pasa a ser el elemento anterior al sacado. La pila NO debe estar vacía para poder aplicar esta operación.

- Destructor: En caso de usar memoria dinámica, la libera.

Además de estas operaciones, una Pila contiene el campo cima, anteriormente referenciado, que hace referencia al primer elemento (de tipo Nodo) que hay dentro de dicha Pila, que es tanto el punto de acceso para introducir como para extraer datos de la estructura, siguiendo una arquitectura LIFO (Last In First Out).

A su vez, la Cola contiene los campos "final" y "frente"(primera cola) y "primero" y "ultimo"(segunda cola), que hacen referencia a los lugares por donde se introducen datos (final) y por donde salen (frente), siguiendo así una estructura FIFO (First In First Out).
Las Colas contienen las siguientes operaciones:

- Constructor: Construye una Cola vacía.

- encolar (): Introduce un elemento en la Cola. El nuevo elemento pasa a ser el último elemento de la cola (final).

- Desencolar(): Extrae el primer elemento situado de la Cola (frente). El frente de la Cola pasa a ser el elemento anterior al sacado. La Cola no debe estar vacía para poder aplicar esta operación.

- getValorFrente(): Persona. Devuelve el primer elemento de la Cola (frente).

- Destructor: En caso de usar memoria dinámica, la libera.

Debemos añadir que ambos TAD usan como elementos para construirse otros tipos de datos denominados como "Nodo" y "Gondola". Cada nodo/gondola está construido por:

- Valor: Campo que contiene el elemento que estamos introduciendo. En nuestra práctica serán Personas.

- Siguiente: Puntero que señalará al siguiente nodo de la estructura o, en su caso, a NULL si es el último de la misma.

- Constructor (v: Persona, *p = NULL). Construirá nuestro nodo introduciéndole el valor que queramos y estableciendo el puntero a NULL. Este puntero cambiará con las operaciones de los TAD.

- Soluciones a problemas surgidos durante la implementación

Los principales problemas que me surgieron fueron debido a la falta de costumbre de trabajar en Programación Orientada a Objetos con C++, ya que, anteriormente, nuestro ámbito de trabajo nunca nos ha exigido ni enseñado este lenguaje ni sus características.

Uno de los problemas surgidos durante la realización de esta práctica fue la generación de números aleatorios. Tras varias ejecuciones, siempre obtenía los mismos resultados.

Después de un tiempo con el debugger, me di cuenta que lo que fallaba era la semilla de la generación de los propios números. Tenía la semilla (función 'srand()') en la función aleatorio, cuál se ejecutaba tantas veces como llamadas tenía al constructor "nodo". Al ejecutarse siempre en el mismo segundo, no variaba el tiempo y por tanto la semilla era la misma. La solución fue poner 'srand()' en el main.

Otro de los problemas ha sido poder encolar un array de Personas. Al pasar el array al constructor de gondola, solo se guardaba el primer carácter del array, obviando el valor de los restantes.

La solución fue la implementación de bucles 'for' para el menaje correcto de los arrays. Uno en el constructor de gondola y otro en el método 'mostrar'.

- **Explicación del comportamiento del programa**

El programa realiza una simulación del funcionamiento de un telesilla. Para ello creamos 5 colas a modo de tornos para que las personas puedan acceder a las gondolas.

Tras esto, generamos aleatoriamente las Personas que van a llenar los TAD creados (5 colas). A continuación, las primeras personas de cada cola, se montan en una gondola, vigilando que 4 niños (simbolizados como 'N') no se monten al mismo tiempo en la gondola.

Una vez que la condición se cumpla, las personas se montarán en la gondola, e irán hasta el final del telesilla.

Una vez que las personas lleguen al final del recorrido, se descolarán de la cola.