

EjercicioListaContigua1

Estructuras de Datos

Tema 2: listas, pilas y colas

1º Grado en Ingeniería de la Computación

© Profesor Dr. Carlos Grima Izquierdo (www.carlosgrima.com)

URJC (www.urjc.es)

Programar la clase “ListaContigua” que será una lista contigua de números de tipo “int”. Además será redimensionable automáticamente según se van añadiendo o quitando elementos.

La clase ListaContigua tendrá las siguientes operaciones públicas:

- Un constructor para crear una lista vacía y con capacidad 0.
- Un destructor para liberar su memoria. El destructor es un método especial (como el constructor) que, en C++, tiene que llamarse igual que la clase pero con un “~” delante. El destructor debe de ser público, y se ejecuta automáticamente cuando muere el objeto. No puede tener ningún parámetro. Tienes más información de cómo programar un destructor en C++ en <http://c.conclase.net/curso/?cap=030>
- Un método para devolver el elemento de una determinada posición (empezando en 0).
- Un método para modificar el elemento de una determinada posición
- Un método para insertar un elemento al final de la lista. El método por dentro analiza primero si la capacidad de la lista es suficiente para contener al nuevo elemento que se va a insertar. Si no es así, antes de insertar el elemento tendrá que ampliar la capacidad en “INCREMENTO” posiciones más (INCREMENTO es una constante #define que se pondrá en el .h de la clase). Pondremos que INCREMENTO sea 2 (aunque esto no sea muy realista), con el fin de probar mejor el programa.
- Un método para eliminar el último elemento de la lista. Si, después de eliminar, se ve que en la lista sobran 2*INCREMENTO posiciones, la capacidad de la lista se reduce en INCREMENTO.
- Un método para devolvernos el tamaño actual de la lista.
- Un método para devolvernos la capacidad actual de la lista.

Fuera de la clase, en una biblioteca aparte, programa un método para imprimir por pantalla una lista. Imprimirá el tamaño, la capacidad y la propia lista de los elementos, separados por comas entre ellos (después del último elemento no deberá aparecer la coma). Recibirá un puntero a un objeto de tipo ListaContigua, pues si recibiera un objeto en sí (en vez de un puntero), tendríamos que copiar el parámetro real en el formal, con el consiguiente gasto innecesario de tiempo y memoria. Además, para que funcionara, tendríamos que tener un constructor de copia (es un constructor especial de C++ que no vemos en esta asignatura).

Para acceder a un método público de un objeto a partir de su puntero, se debe hacer con el operador flecha en vez del operador punto:

```
MiClase* punteroAObjeto;  
punteroAObjeto->metodo(); // Equivale a (*punteroAObjeto).metodo();
```

El método de imprimir por pantalla es un método de “interfaz” (se dedica a relacionarse con el usuario del programa, en este caso mediante la pantalla). Sin embargo, la lista en sí misma es “modelo” (es el núcleo del programa). Es una buena práctica mantener separado el interfaz del modelo, por eso no es buena práctica meter un método de interfaz (como imprimir) en la clase ListaContigua (que es de modelo). De este modo, si en el futuro queremos cambiar la interfaz, no tendremos que tocar nada del modelo.

Algunas consideraciones:

- Recuerda que, a partir de ahora, en los comentarios de cada método o función siempre deberás poner la complejidad temporal en el peor caso.
- Utiliza `realloc()` para reasignar memoria (ampliarla o disminuirla). Visita su página de información en www.cplusplus.com para obtener todos los detalles.
- En el destructor, libera la memoria con `free()`
- Se deberán añadir los atributos y métodos privados que se consideren oportunos. Se valorará una buena división de las tareas siguiendo el principio de la máxima cohesión y el mínimo acoplamiento.

Para probar la lista, el main deberá hacer lo siguiente:

1. Crear una lista vacía. Imprimirla.
2. Rellenar con los números naturales de 0 a 11, en orden. Imprimir la lista cada vez que se inserta uno.
3. Imprimir el elemento 0 de la lista.
4. Imprimir el elemento 11 de la lista.
5. Cambiar el elemento 4 por 50. Imprimir la lista.
6. Borrar los cuatro últimos elementos de la lista. Imprimir la lista.
7. Insertar 100, 101, 102 y 103 al final de la lista. Imprimir después de cada inserción.

```
C:\WINDOWS\system32\cmd.exe
Nueva ListaContigua creada:
n=0|Max=0|ListaContigua=vacia
Rellenando ListaContigua:
n=1|Max=2|ListaContigua=0
n=2|Max=2|ListaContigua=0,1
n=3|Max=4|ListaContigua=0,1,2
n=4|Max=4|ListaContigua=0,1,2,3
n=5|Max=6|ListaContigua=0,1,2,3,4
n=6|Max=6|ListaContigua=0,1,2,3,4,5
n=7|Max=8|ListaContigua=0,1,2,3,4,5,6
n=8|Max=8|ListaContigua=0,1,2,3,4,5,6,7
n=9|Max=10|ListaContigua=0,1,2,3,4,5,6,7,8
n=10|Max=10|ListaContigua=0,1,2,3,4,5,6,7,8,9
n=11|Max=12|ListaContigua=0,1,2,3,4,5,6,7,8,9,10
n=12|Max=12|ListaContigua=0,1,2,3,4,5,6,7,8,9,10,11
Elemento0=0|Elemento11=11
Cambio elemento 4 por 50. Nueva ListaContigua:
n=12|Max=12|ListaContigua=0,1,2,3,50,5,6,7,8,9,10,11
Borramos el ultimo elemento. Nueva ListaContigua:
n=11|Max=12|ListaContigua=0,1,2,3,50,5,6,7,8,9,10
Borramos el ultimo elemento. Nueva ListaContigua:
n=10|Max=12|ListaContigua=0,1,2,3,50,5,6,7,8,9
Borramos el ultimo elemento. Nueva ListaContigua:
n=9|Max=12|ListaContigua=0,1,2,3,50,5,6,7,8
Borramos el ultimo elemento. Nueva ListaContigua:
n=8|Max=10|ListaContigua=0,1,2,3,50,5,6,7
Insertamos 100 al final. Nueva ListaContigua:
n=9|Max=10|ListaContigua=0,1,2,3,50,5,6,7,100
Insertamos 101 al final. Nueva ListaContigua:
n=10|Max=10|ListaContigua=0,1,2,3,50,5,6,7,100,101
Insertamos 102 al final. Nueva ListaContigua:
n=11|Max=12|ListaContigua=0,1,2,3,50,5,6,7,100,101,102
Insertamos 103 al final. Nueva ListaContigua:
n=12|Max=12|ListaContigua=0,1,2,3,50,5,6,7,100,101,102,103
Presione una tecla para continuar . . . ■
```