



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: TISTA GARCÍA EDGAR

Asignatura: ESTRUCTURA DE DATOS Y ALGORITMOS I

Grupo: 1

No de Práctica(s): 6] Estructuras de datos lineales: Cola circular y cola doble.

Integrante(s): MURRIETA VILLEGAS ALFONSO

Semestre: 2018 - 2

Fecha de entrega: 2 DE ABRIL DE 2018

Observaciones:

CALIFICACIÓN: _____

ALUMNO: ALFONSO MURRIETA

1

Estructuras de datos lineales: Cola circular y cola doble.

1] Introducción

Como pudimos ver en prácticas pasadas las estructuras de datos son una forma de organizar datos para facilitar y manejar de manera más eficiente el claro ejemplo fue el uso y manejo de información a través de dos principales estructuras que son la pila y la cola

En el caso de la pila o en inglés Stack es una estructura de datos lineal y dinámica, la cual es del tipo LIFO (Last - in, First - out) debido a que el último elemento que se agrega es el primero que se elimina. La cola (o queue) también es una estructura de datos lineal, sin embargo, es del tipo FIFO (first in first out). Es lineal porque cada elemento tiene un único **predecesor** y un único **sucesor**, y es dinámica porque su tamaño no es fijo y se puede definir conforme se requiera. Las operaciones básicas dentro de una lista al igual que en la pila son BUSCAR, INSERTAR Y ELIMINAR.

En el caso de la función INSERTAR como dice su nombre, inserta elementos al final de la estructura y permite ELIMINAR elementos por el inicio de la misma. La operación de INSERTAR también se le llama ENCOLAR y la operación de ELIMINAR también se le llama DESENCOLAR

2] Objetivo de la práctica

- Se revisará las definiciones, características, procedimientos y ejemplos de las estructuras lineales Cola circular y Cola doble, con la finalidad de que comprendas sus estructuras y puedas implementarlas.

3] Desarrollo

Anteriormente se mencionó de manera general los aspectos de la cola, pero en esta práctica en particular debemos hacer hincapié en el hecho de que no solo existe un tipo de cola, un ejemplo es la cola simple la cual implementa la política First-In, First-Out (FIFO), además de que para eliminar elementos se recorre a un apuntador HEAD al siguiente elemento de la estructura, dejando espacios de memoria vacíos al inicio de la misma. Existen dos mejoras de la cola simple que utilizan de manera más eficiente la memoria: la **cola circular** y la **cola doble**.

3.1] Análisis de la teoría

COLA CIRCULAR

La cola circular al igual que la cola simple son estructuras de datos lineales, también ambas operaciones comparten las mismas operaciones como ENCOLAR o DESENCOLAR, sin embargo, una de las mayores diferencias es que en las colas circulares el siguiente elemento del último es el primero.

La cola circular posee dos referencias, una al inicio (HEAD) y otra al final (TAIL) de la cola. En una cola circular vacía ambas referencias (HEAD y TAIL) apuntan a nulo.

Por lo tanto y como es obvio no podemos desencolar a una cola que se encuentra vacía pues para ello se necesita de al menos un elemento.

Para poder llevar acabo lo anterior agregamos elementos a través de la función ENCOLAR, sin embargo, esto como resultado nos cambia los valores de nuestras referencias donde HEAD y TAIL ahora apuntan al único elemento existente. Por otro lado, cuando tenemos a una cola que se encuentra llena (Qué está a su máxima capacidad) para poder encolar otro elemento debemos hacer uso de la función DESENCOLAR a través de la referencia HEAD la cual a su vez se recorre al siguiente elemento.

Sin embargo, una de las características que diferencia la cola circular a la cola simple es sobretodo el hecho de que las colas circulares pueden ir cambiando las posiciones de referencia tanto del TAIL como del HEAD y esto es posible porque al encolar o desencolar elementos cuando están llenas las colas, de manera indirecta estamos cambiando las posiciones de las referencias (Véase en Diagrama 1).

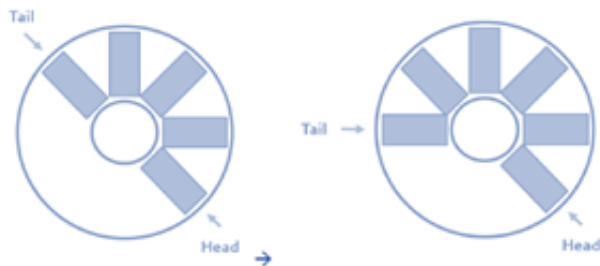


Diagrama 1: Esquema donde se muestra como cambiaría la posición de las referencias HEAD y TAIL

Por lo tanto, para diseñar un algoritmo que defina el comportamiento de la cola circular es necesario considerar 3 casos para las operaciones de ENCOLAR y DESENCOLAR:

- Estructura vacía
- Estructura llena
- Estructura con elemento(s)

COLA DOBLE

La cola doble o también conocida como bicola, es una estructura de datos lineales similar a la cola simple con operaciones como ENCOLAR O DESENCOLAR, sin embargo, como principal diferencia está que este tipo de estructura tiene la capacidad de poder realizar las operaciones anteriores en ambos extremos de la cola es decir tanto en el HEAD como en el TAIL.

Como es notable la cola doble es una mejora de una cola simple debido a que es posible realizar operaciones de inserción por ambos extremos de la estructura, permitiendo con esto utilizar el máximo espacio disponible de la estructura

Al igual que cualquier cola existen 3 casos de situación los cuales son cuando la cola se encuentra vacía, llena o simplemente contiene elemento(s)

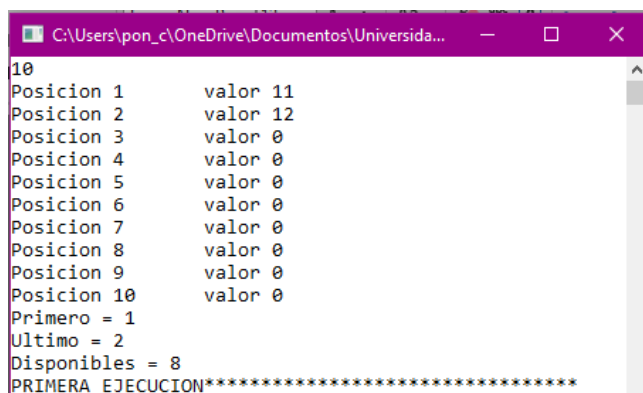
Como se dijo anteriormente, este tipo de estructura tiene la capacidad de insertar o extraer elementos de ambos extremos por ejemplo al momento de tener una cola llena podemos sacar no solamente el último elemento como era el caso de las colas simples, ahora también podemos extraer el primer elemento, es por ello que lo más relevante de este tipo de estructura es el tener siempre en cuenta la posición tanto del HEAD como del TAIL.

3.2] Análisis de los problemas (Práctica)

EJERCICIO I

En este ejercicio se usó un programa previamente escrito el cual además utiliza una biblioteca llamada “colaCircularDoble.h” (NOTA: Cambié el nombre de la biblioteca debido a la naturaleza del tipo de estructura que crea). Para poder analizar de una manera más fácil dividiré el análisis en 3 apartados debido a que de hecho el programa tiene 3 ejecuciones.

En la primera ejecución primero se condicionó el programa a tener un tamaño de 10 espacios dentro de una cola creada a partir de la librería “colaCircularDoble.h”, posteriormente mediante la función “encolarFinal()” se agregaron 2 valores a las posiciones 1 y 2, recordemos que se agregan al inicio debido a la naturaleza que tienen las colas (FIFO), a continuación se muestra la salida de la primera ejecución:



```
10
Posicion 1      valor 11
Posicion 2      valor 12
Posicion 3      valor 0
Posicion 4      valor 0
Posicion 5      valor 0
Posicion 6      valor 0
Posicion 7      valor 0
Posicion 8      valor 0
Posicion 9      valor 0
Posicion 10     valor 0
Primero = 1
Ultimo = 2
Disponibles = 8
PRIMERA EJECUCION*****
```

Imagen 1: Salida del ejercicio 1, solamente se muestra la primera ejecución

NOTA: También podemos notar que la cantidad disponible de espacios son 8 debido a que son

En la segunda ejecución mediante la función `desencolarInicio` simplemente se extraen los valores previamente ingresados a la cola, al extraer estos elementos de la cola como resultado obtenemos nuevamente una cola vacía donde el índice primero se encuentra en la primera posición de la cola, a continuación, se muestra la salida de la segunda ejecución.

```
Posicion 1      valor 0
Posicion 2      valor 0
Posicion 3      valor 0
Posicion 4      valor 0
Posicion 5      valor 0
Posicion 6      valor 0
Posicion 7      valor 0
Posicion 8      valor 0
Posicion 9      valor 0
Posicion 10     valor 0
Primero = 1
Ultimo = 0
Disponibles = 10
SEGUNDA EJECUCION*****
```

Imagen2: Salida del ejercicio 1, solamente se muestra la segunda ejecución

Por último, en la tercera ejecución del programa vemos que ahora se asignan valores tanto al inicio como al final de la cola mediante las funciones `encolarFinal()` y `encolarInicio()` lo cual nos hace ver que este tipo de estructura es una cola circular doble debido a que tiene la capacidad de encolar elementos tanto al inicio como al final de la cola.

Además, se desencola (Se extrae) el último elemento asignado en el final de la cola que en ese caso fue el número 14, a continuación, se muestra la salida de la tercera ejecución del programa.

```
Posicion 1      valor 13
Posicion 2      valor 0
Posicion 3      valor 0
Posicion 4      valor 0
Posicion 5      valor 0
Posicion 6      valor 0
Posicion 7      valor 0
Posicion 8      valor 0
Posicion 9      valor 9
Posicion 10     valor 10
Primero = 9
Ultimo = 1
Disponibles = 7
TERCERA EJECUCION*****Presione una tecla para continuar . . .
```

Imagen3: Salida del ejercicio 1, solamente se muestra la tercera ejecución

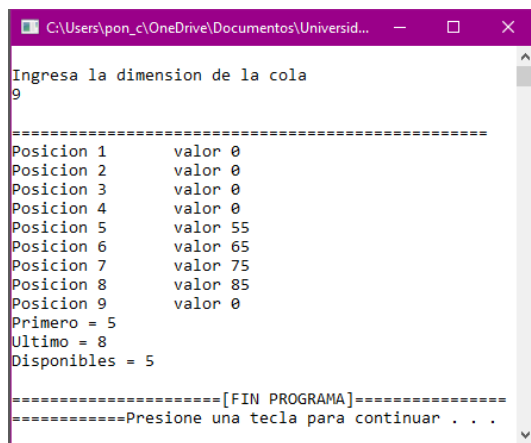
EJERCICIO II

Para este ejercicio se escribió un programa basado en el anterior solo que se tuvieron que hacer algunas excepciones debido al tipo de estructura que se utilizará.

Primero, al utilizar colas circulares y no colas circulares dobles debemos tomar en cuenta que las funciones "encolarInicio" y "desencolarFinal" de la biblioteca "colaCircularDoble.h" no podemos usarlas debido a que la naturaleza de la cola circular solamente permite que se encole o agreguen elementos al inicio de esta y solamente se puede desencolar o extraer datos al inicio.

2A

Una vez codificado el programa, se probó usando un tamaño de 9 elementos dentro de la cola, la salida del programa fue la siguiente:



```
C:\Users\pon_c\OneDrive\Documentos\Universid...
Ingresa la dimension de la cola
9
=====
Posicion 1      valor 0
Posicion 2      valor 0
Posicion 3      valor 0
Posicion 4      valor 0
Posicion 5      valor 55
Posicion 6      valor 65
Posicion 7      valor 75
Posicion 8      valor 85
Posicion 9      valor 0
Primero = 5
Ultimo = 8
Disponibles = 5

=====
=====Presione una tecla para continuar . . .
```

Imagen4: Salida del ejercicio 2, al dar una tamaño de 9 elementos.

Como podemos ver lo que pasa es que al agregar datos dentro de la cola se van encolando de manera ascendente, sin embargo, no llega a estar completamente llena (Para de agregar datos en la integración del séptimo elemento). Posteriormente, se desencolan 3 elementos los cuales son los primeros 3 de la cola, por último, se agrega otro elemento (En la posición 8) y además se desencola otro elemento que es el de la posición 4.

Como resultado final tenemos que la cola tiene solamente elementos en las posiciones 5, 6, 7 y 8 , mientras que las posiciones 1 a 4 y 9 están vacías (Obteniendo solamente 5 elementos vacíos). Por otro lado, tenemos que el primero se encuentra en la posición 5 mientras que el último se encuentra en 8.

2B

Una vez codificado el programa, se probó usando un tamaño de 5 elementos dentro de la cola, la salida del programa fue la siguiente:

NOTA: Recordemos que para poder agregar elementos a una cola está debe tener espacios disponibles o usar la función desencolar para poder obtener espacios vacíos.

```
C:\Users\pon_c\OneDrive\Documentos\Universi...
Ingresa la dimension de la cola
5
ya esta a su maxima capacidad
ya esta a su maxima capacidad

=====
Posicion 1      valor 85
Posicion 2      valor 0
Posicion 3      valor 0
Posicion 4      valor 0
Posicion 5      valor 55
Primero = 5
Ultimo = 1
Disponibles = 3

=====
=====Presione una tecla para continuar . . .
```

Imagen5: Salida del ejercicio 2, al dar un tamaño de 5 elementos

Lo que sucede en este caso los primeros 5 elementos son encolados a la cola, sin embargo, y como es visible al llegar a la máxima capacidad de la cola los siguientes dos elementos no se pueden encolar, pero el último elemento que en este caso es el 85 si se pudo encolar debido a que anterior a la línea encargada de eso se desencolan dos elementos lo cual permitió que se pudiera agregar el 85.

Como resultado final tenemos que la cola tiene solamente elementos en las posiciones 1 y 5, mientras que las posiciones 2 a 4 están vacías (Obteniendo solamente 3 elementos vacíos). Por otro lado, tenemos que el primero se encuentra en la posición 5 mientras que el último se encuentra en 1 (Recordemos que sucede esto debido a que es una cola del tipo circular).

EJERCICIO III

En el presente ejercicio se pidió que se programara una cola doble y que a su vez fuera circular para poder obtener 3 estados (Ejecuciones de la cola) indicados. Para ello se utilizaron funciones como encolarFinal(), desencolarInicio(), encolarInicio() y desencolarFinal() incluidas en la biblioteca "colaCircularDoble.h".

Para poder analizar de una manera más fácil se dividió el análisis en 3 apartados debido a que de hecho el programa tiene 3 ejecuciones o estados.

NOTA: Para este ejercicio se consideró un tamaño de 8 elementos dentro de la cola.

- 1) En el primer estado de la cola se puede observar que el primer elemento y los últimos 3 elementos están vacíos, lo cual nos hace notar que en este estado la estructura se comporta como una cola circular ya que para extraer el primer elemento solo se utilizó la función desencolarInicio(). A continuación, se puede ver la implementación de las funciones para llevar a cabo este apartado (Parte del código)

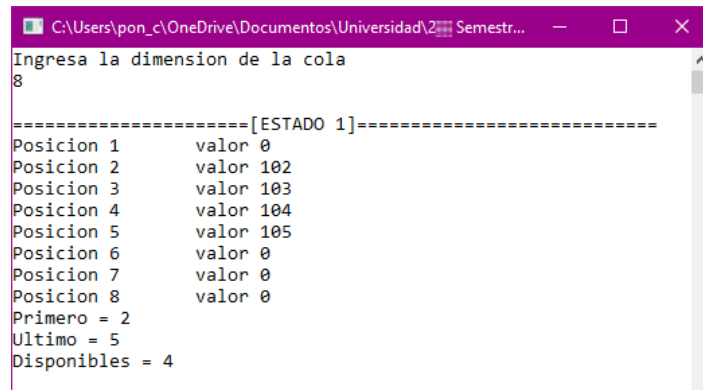
```
encolarFinal(&queue1,98);
encolarFinal(&queue1,102);
encolarFinal(&queue1,103);
encolarFinal(&queue1,104);
encolarFinal(&queue1,105);
desencolarInicio(&queue1);

mostrarValores(queue1);
mostrarIndices(queue1);
```

Imagen6: Parte del código encargada del estado 1

NOTA: No es necesario llenar la cola en el estado 1 ya que podemos ahorrarnos los pasos de encolar y desencolar los elementos de las posiciones 6,7 y 8.

A continuación, se muestra la salida del primer estado del programa.



```
C:\Users\pon_c\OneDrive\Documentos\Universidad\2º Semestr...
Ingresa la dimension de la cola
8

-----[ESTADO 1]-----
Posicion 1      valor 0
Posicion 2      valor 102
Posicion 3      valor 103
Posicion 4      valor 104
Posicion 5      valor 105
Posicion 6      valor 0
Posicion 7      valor 0
Posicion 8      valor 0
Primero = 2
Ultimo = 5
Disponibles = 4
```

Imagen7: Salida del programa del ejercicio 3 (Estado 1)

Como podemos ver el primer valor al igual que los valores 6,7 y 8 están vacíos dando como resultado el primero en la posición 2 y el ultimo en la posición 5 con 4 espacios disponibles dentro de la cola.

- 2) En el segundo estado de la cola se puede observar que el único elemento vacío se encuentra en la posición 7 de la cola, además de que se ha omitido el valor del primero y se han agregado elementos tanto en el final como en el inicio de la cola.

Para poder llevar a cabo lo pedido fue necesario tanto de la adición de elementos en el final como al inicio de la cola por lo tanto y a diferencia del estado anterior, en este caso la cola será de tipo doble y circular debido a que este tipo de estructura permite agregar elementos en ambos extremos de la cola.

Lo primero que se debe hacer para obtener este estado es extraer los elementos tanto del inicio como del final de la cola, después simplemente agregamos los elementos que se piden, en el caso de 99,98 y 97 los agregamos usando la función `encolarInicio()` mientras que 120 y 121 lo hacemos con la función `encolarFinal()`, esto lo podemos ver en la siguiente imagen.

```
desencolarInicio(&queue1);
desencolarFinal(&queue1);

//Asignación de los valores pedidos
encolarInicio(&queue1,99);
encolarInicio(&queue1,98);
encolarInicio(&queue1,97);
encolarFinal(&queue1,120);
encolarFinal(&queue1,121);
```

Imagen8: Parte del código encargada del estado 2

Podemos ver la posición 7 vacía dando como resultado el primero en la posición 8 y el último en la posición 6 con 1 espacio de la cola libre.

```

===== [ESTADO 2] =====
Posicion 1      valor 98
Posicion 2      valor 99
Posicion 3      valor 103
Posicion 4      valor 104
Posicion 5      valor 120
Posicion 6      valor 121
Posicion 7      valor 0
Posicion 8      valor 97
Primero = 8
Ultimo = 6
Disponibles = 1

```

Imagen9: Salida del programa del ejercicio 3 (Estado 2)

- 3) Por último, el tercer estado de la cola se puede observar que los elementos vacíos se encuentran en las posiciones 8, 1 y 2 de la cola, además de que se ha extraído el valor de la primera, segunda y tercera posición de la cola además de que se ha incluido otro elemento en el final de la cola. Para poder llevar a cabo lo pedido fue necesario el extraer elemento del inicio de la cola y encolar un elemento al final a través de las funciones que se pueden ver en la siguiente imagen.

```

desencolarInicio(&queue1);
desencolarInicio(&queue1);
desencolarInicio(&queue1);

//Asignación del valor pedido
encolarFinal(&queue1,122);

```

Imagen10: Parte del código encargada del estado 3

A continuación, se muestra la salida final del estado 3.

```

===== [ESTADO 3] =====
Posicion 1      valor 0
Posicion 2      valor 0
Posicion 3      valor 103
Posicion 4      valor 104
Posicion 5      valor 120
Posicion 6      valor 121
Posicion 7      valor 122
Posicion 8      valor 0
Primero = 3
Ultimo = 7
Disponibles = 3
===== [FIN DE PROGRAMA] =====

```

Imagen10: Salida del programa del ejercicio 3 (Estado 3)

Podemos ver que las posiciones 8, 1 y 2 están vacía dando como resultado el primero en la posición 3 y el último en la posición 7 con 3 espacio libres en la cola.

3.3] Relación entre los ejercicios y el tema

Las colas son una estructura lineal que tiene un sinfín de usos dentro del mundo de la programación. Para poder seleccionar un elemento dentro de un conjunto de datos es muy común que, en diversas aplicaciones, sobre todo en videojuegos

La selección de un conjunto de elementos finitos donde a partir del último elemento se puede regresar al

primero utiliza, de manera implícita, una **cola circular**, es el claro ejemplo de la selección de un personaje, selección de un arma, cambios de uniformes, etc. En general, cuando recorremos un conjunto de elementos o datos finitos donde al momento de llegar al último elemento pasamos enseguida al primero (El siguiente) se tienen una cola circular.

En el caso de las **colas dobles** al igual que los otros tipos de colas tienen muchísimos usos, sin embargo, uno de los principales es la implementación de estas al llevar procesos.

Dentro de un SO (Sistema Operativo) hay procesos que tienen una mayor prioridad ya sea porque necesitan estar ejecutándose constantemente o porque necesitan una mayor atención para llevarse a cabo, pero para poder dar prioridad a un proceso este debe ser atendido lo más pronto posible y esto es posible con las colas dobles donde al momento de llegar dos procesos siempre se le da mayor prioridad al proceso más relevante y esto se hace encolándolo al principio de cola y no al final como es el caso de los otros tipos de colas.

4] Conclusiones

A lo largo de esta práctica realizamos diversos programas que fueron un primer encuentro con estructuras lineales como son la cola circular y la cola doble.

Por ejemplo, en el caso del primer ejercicio a través de los conceptos y principios de las colas, analizamos un programa ya hecho, donde pudimos notar de manera más visible la forma en que funciona una cola circular doble y como es que esta nos da la disponibilidad de agregar y extraer elementos en ambos extremos, por otro lado, en el segundo ejercicio a través de 2 pruebas notamos las funciones principales de las colas como son el verificar cuando está o no llena una cola, fue en el segundo caso donde notamos que una vez que estaba llena la cola simplemente se tuvieron que omitir los elementos que se querían agregar y no fue sino hasta el momento en que se desencoló un elemento donde nuevamente se pudo agregar elementos a la cola.

Por último, el tercer ejercicio fue sin duda un programa que tenía como motivo el globalizar todos los conocimientos aprendidos acerca de las colas ya sean circulares o circulares dobles y es que notablemente se pude apreciar como en el primer estado solamente se considera a una cola circular mientras que en el segundo y tercer estado cambiamos de naturaleza (Cola Circular Doble) para poder llevar la extracción en ambos extremos de la cola.

Sin duda una práctica que realmente me ha resultado fácil de hacer, pero sobretodo muy importante para tener en claro el uso e implementación de las colas como una estructura.

5] Referencias

- 1) Curso de Programación C7 C++. Javier Ceballos, tercera edición. USA, Ra – Ma. Educación 2007
- 2) Introduction to Algorithms. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, McGraw-Hill.
- 3) El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.