	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): 7

Integrante(s): GÓMEZ LUNA ALEJANDRO

No. de Equipo de cómputo empleado: 42

No. de Lista o Brigada: 14

Semestre: 2019-2

Fecha de entrega: 3/Abril/2019

Obervaciones:

CALIFICACIÓN: _____

- Objetivo de la práctica

  Revisarás las definiciones, características, procedimientos y ejemplos de las estructuras lineales Lista simple y Lista circular, con la finalidad de que comprendas sus estructuras y puedas implementarlas

- Desarrollo

  Introducción de la guía: En la guía se empieza a hablar de que es una lista y une breve descripción de su funcionamiento general, sin embargo, en clase se mencionó que una lista tiene que tener solamente elementos del mismo tipo, además de que los elementos de la lista no se almacenan contiguamente, por lo que resultaban mejor su almacenamiento en memoria. Estas últimas cuestiones no se mencionan en la guía.

  En la guía, la descripción del funcionamiento de las operaciones en una lista ligada es un tanto distinto y limitado al visto en clase. En primer lugar, la operación de inserción en una lista ligada solo se realiza al inicio de la lista. En segundo lugar, no existe la referencia al último elemento, el cual es nombrado Tail. En último lugar, las explicaciones son bastantes superficiales, sin mayor cantidad de esquemas para lograr una mejor ejemplificación. En los demás puntos, la explicación dada en clase coincide con lo explicado en la guía.

  Para la descripción de una lista circular, se emplea el término de Tail y Head, sin embargo, en clase se mencionó que la referencia al último elemento, es decir, Tail, ya no existe y solamente se conserva la referencia al elemento Head. De igual forma que para la lista ligada simple, la operación definida solo es inserción al inicio de la lista circular. Asimismo, las explicaciones no son bastante detalladas. En los demás puntos, la explicación dada en clase coincide con lo explicado en la guía.

  En ambos casos, tanto para lista ligada simple, como para la lista circular, se ofrecen ejemplos en donde se ocupan este tipo de estructuras de datos.

  Por último, cabe resaltar que las explicaciones en clase contienen más información, así como una explicación más simple, puesto que se realizan esquemas que permiten ejemplificar el funcionamiento.

  En cuanto a las bibliotecas proporcionadas, para la biblioteca lista.h podemos encontrar:

  - crearLista( ). Crea una lista vacía, en donde la referencia a head se le da un valor de NULL. Esta función retorna una lista.

- print_list( ): Esta función imprime todos los valores de los nodos en la lista que fue pasada como argumento, verificando primero que la lista no esté vacía.
- addFInalLista( ): Esta función toma como argumentos el valor del nodo y la lista en donde agregar el nodo, que será hasta el final de la lista. Si la lista esta vacía solamente actualiza la referencia de head para que sea este nuevo nodo.
  En caso contrario, utiliza una variable temporal asociada con head para poder recorrer toda la lista y poder agregar el nuevo nodo al final de la lista.
- addPrincipioLista( ): Esta función toma como argumentos el valor del nodo y la lista en donde agregar el nodo, que será hasta el inicio de la lista. Dado que es al inicio, solamente hay que asociar el nodo con la referencia de head y actualizar head para que sea este nuevo nodo.
- addIesimoLista( ): Esta función toma como argumentos el valor del nodo y la lista en donde agregar el nodo , que será en cierta posición. Dentro de la función se pide al usuario que ingrese la posición en donde quiere insertar el nodo. Ya con este valor obtenido, se recorre la lista haciendo uso de una variable temporal, hasta la posición correspondiente para posteriormente asociar el nodo con la lista.
  Cabe resaltar que esta función no tiene un prototipo previamente declarado.
- borrarPrimero( ): Esta función toma como argumento la lista en donde se borrará el nodo, que será al inicio de la lista. Se verifica primero que la lista no este vacía. En caso de no estarlo, se procede a eliminar el nodo. Para realizarlo se usa una variable auxiliar que actualiza el valor de head para que sea el siguiente y luego se libera el espacio asignado para el nodo eliminado.
- borrarUltimo( ): Esta función toma como argumento la lista en donde se borrará el nodo, que será al final de la lista. Se verifica primero que la lista no este vacía. En caso de no estarlo, se procede a eliminar el nodo. Para realizarlo se usa una variable auxiliar que recorre toda la lista hasta llegar al penúltimo nodo. Se actualiza la referencia del penúltimo para que sea NULL y luego se libera el espacio asignado para el nodo eliminado.
- primerElemento( ): Esta función toma como argumento la lista de al cual se quiere saber el primer elemento y retorna el valor almacenado en el primer nodo.

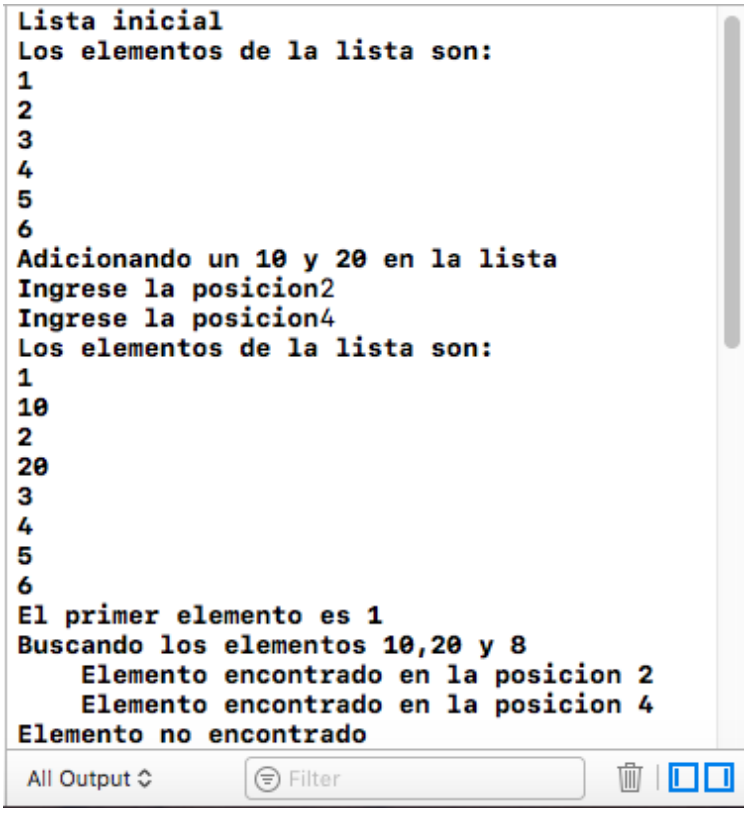
Para la biblioteca listacirc.h se tienen casi las mismas funciones de la biblioteca lista.h, aunque éstas tienen ligeras variaciones, dado que es una lista circular. Dichas variaciones son:

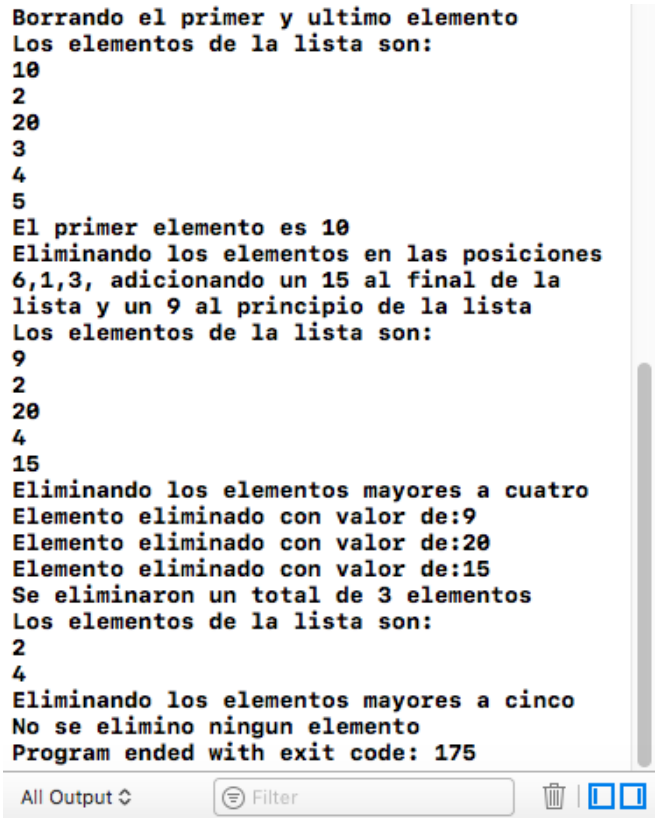
- crearLista( ): No presenta variaciones.
- print_list( ): Esta función varía en la forma en como imprime los valores de todos los nodos, ya que ahora recorre toda la lista haciendo uso del tamaño de la lista.
- addFinalLista( ): Esta función varia en la forma en como añade el nodo, ya que ahora el siguiente de éste debe apuntar a la referencia de head en la lista, ya que es una lista circular y el último nodo está relacionado con el primer nodo. Asimismo, para poder recorrer toda la lista e insertar el nodo al final, se hace uso del tamaño de la lista. En caso de que no haya nodos, el siguiente del nuevo nodo es el mismo nodo.
- addPrincipioLista( ): Esta función varia en la forma en como añade el nodo, ya que ahora el siguiente del último nodo va a apuntar hacia este nueva nodo. Por otro lado, la referencia de head pasa a ser este nuevo nodo. Por último, para poder ligar el último nodo con este nuevo nodo, se recorre toda la lista hasta llegar al último nodo.
- borrarPrimero( ): Esta función no presenta cambios, no obstante, su funcionamiento es un tanto erróneo, ya que al eliminar el primer nodo, se pierde la ligadura entre el último y primer nodo, puesto que no se relaciona el último nodo con el nuevo nodo que se encuentra en head.
- borrarUltimo( ): Esta función presenta cambios, los cuales no son del todo correctos. El principal error de esta función es que nunca se cumple la segunda condición y el ciclo while nunca se detiene, ya que todos los nodos tienen un sucesor.

Ejercicios propuestos

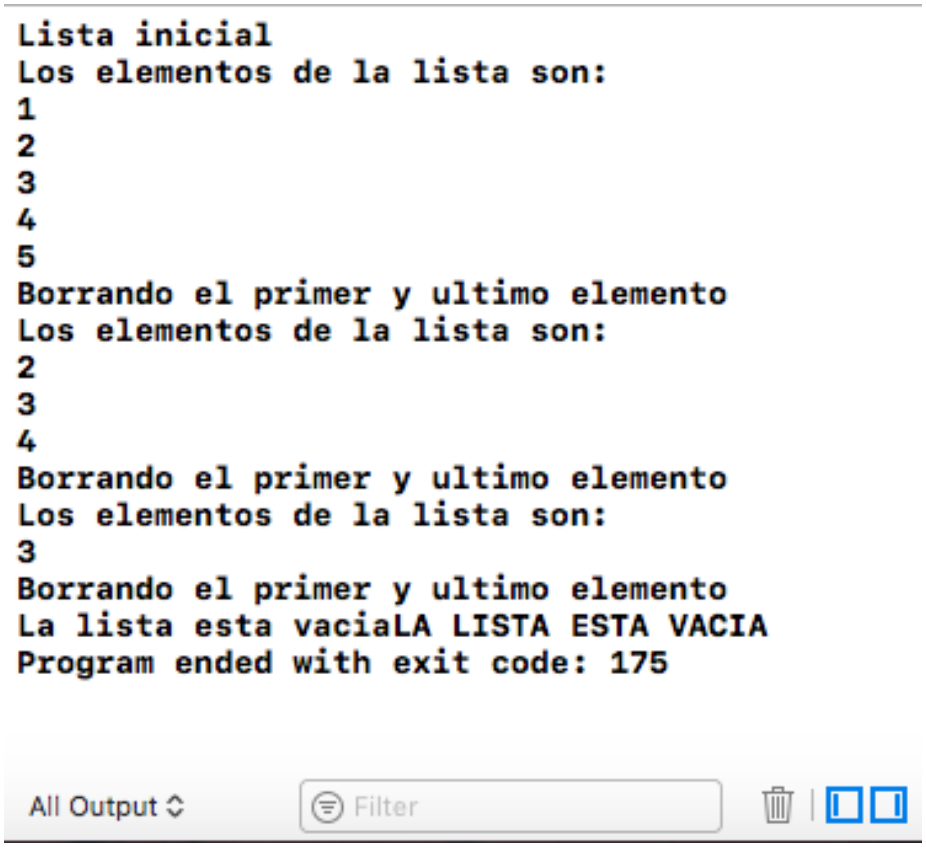
1) Actividad 1: Para esta actividad, se tuvieron que realizar varias cuestiones con base en la biblioteca lista.h, las cuales son:
    a. Se elaboró un sencillo programa para determinar el funcionamiento de las funciones que se encontraban en lista.h. Al ejecutar este pequeño programa de prueba, se pudo comprobar que todas las funciones que se encontraban en la biblioteca tuvieron un funcionamiento adecuado.

b. Se pidió elaborar una función de búsqueda, la cual devolviera la posición del nodo en el cual se encontraba dicho valor. En caso de no encontrarlo, se notificaba en pantalla. Esta función recorre toda la lista hasta llegar al último nodo, verificando el valor almacenado en cada nodo. Si encuentra el valor buscado, deja de buscar en los demás nodos.
c. Se pidió elaborar una función para eliminar un nodo en cualquier posición de la lista. Para poder realizarlo, se pide la posición del nodo que se quiere eliminar. Posteriormente, se recorre toda la lista hasta llegar al nodo anterior al que se quiere eliminar, con la finalidad de primero actualizar la lista y después eliminar el nodo sin perder la referencia de los elementos subsecuentes. Cabe remarcar que, si se quiere eliminar un nodo en la primera posición, se llama a la función borrarPrimero.
d. Se pidió elaborar una función para eliminar todos los nodos que tuvieron elementos mayores a un valor patrón. En esta función se recorre toda la lista, nodo por nodo, y cuando se encuentra un nodo que tenga un valor mayor, se usa la función declarada en el punto anterior.



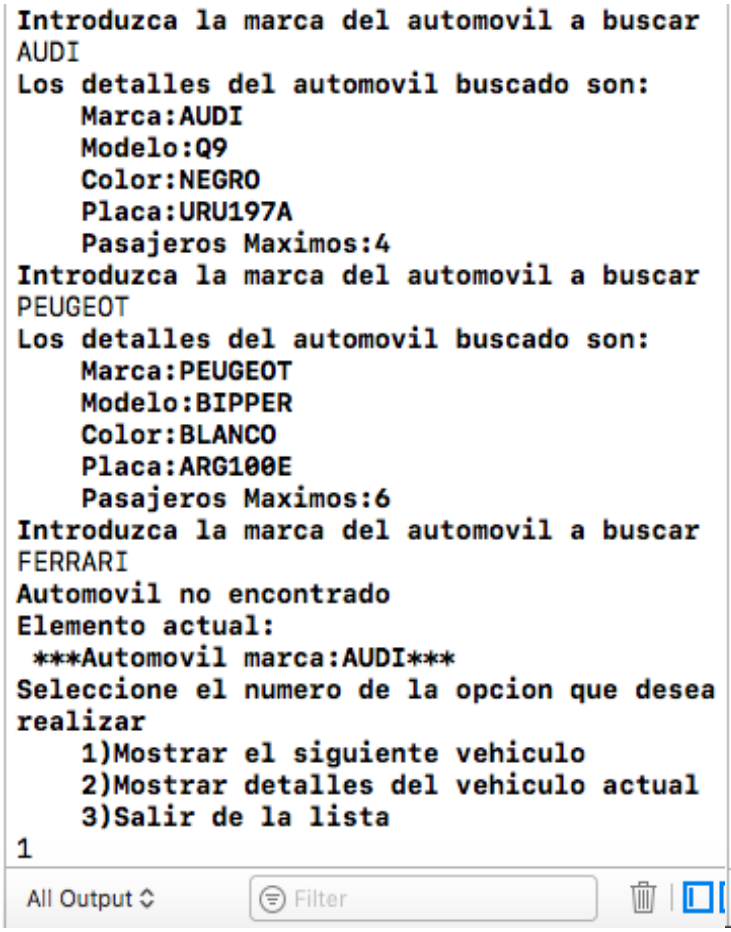


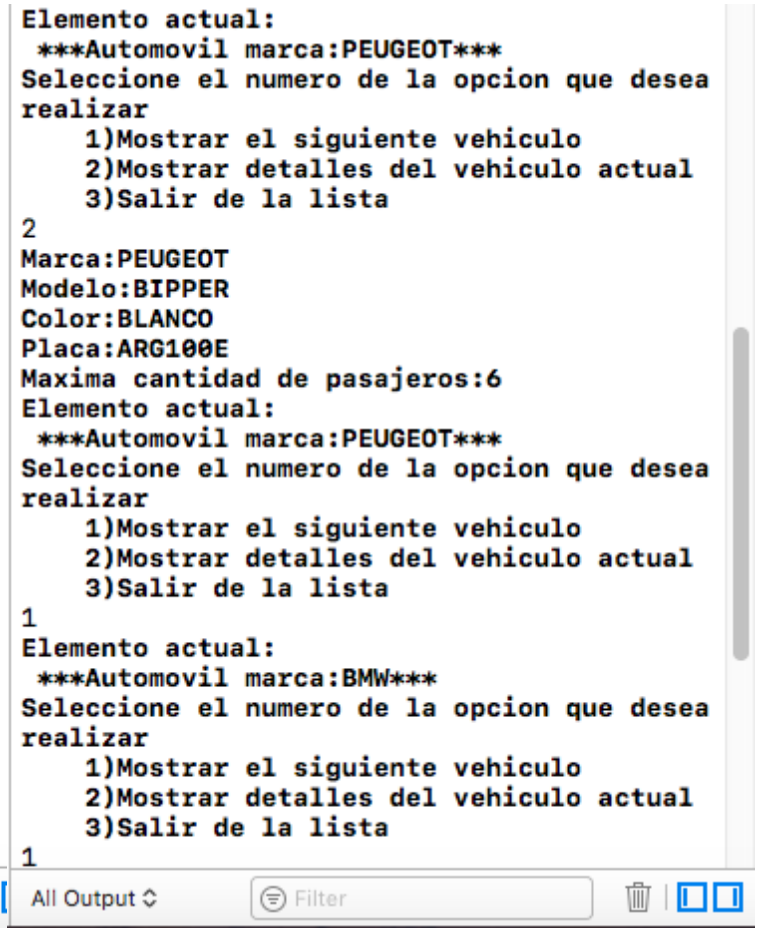
2) Actividad 2: Para esta actividad, se tuvieron que realizar varias cuestiones con base en la biblioteca listacirc.h, las cuales son:
   a. Se elaboró un sencillo programa para determinar el funcionamiento de las funciones que se encontraban en listacirc.h. Al ejecutar este pequeño programa de prueba, se pudo comprobar que existían errores en las funciones de borrarPrimero y borrarUltimo, ya que éstas trataban a la lista circular como una lista simple, lo que ocasionaba que no se relacionara el último nodo con el primer nodo y que la manera en como se recorriera la lista diera origen a un ciclo que nunca termina. Estos errores se corrigieron.

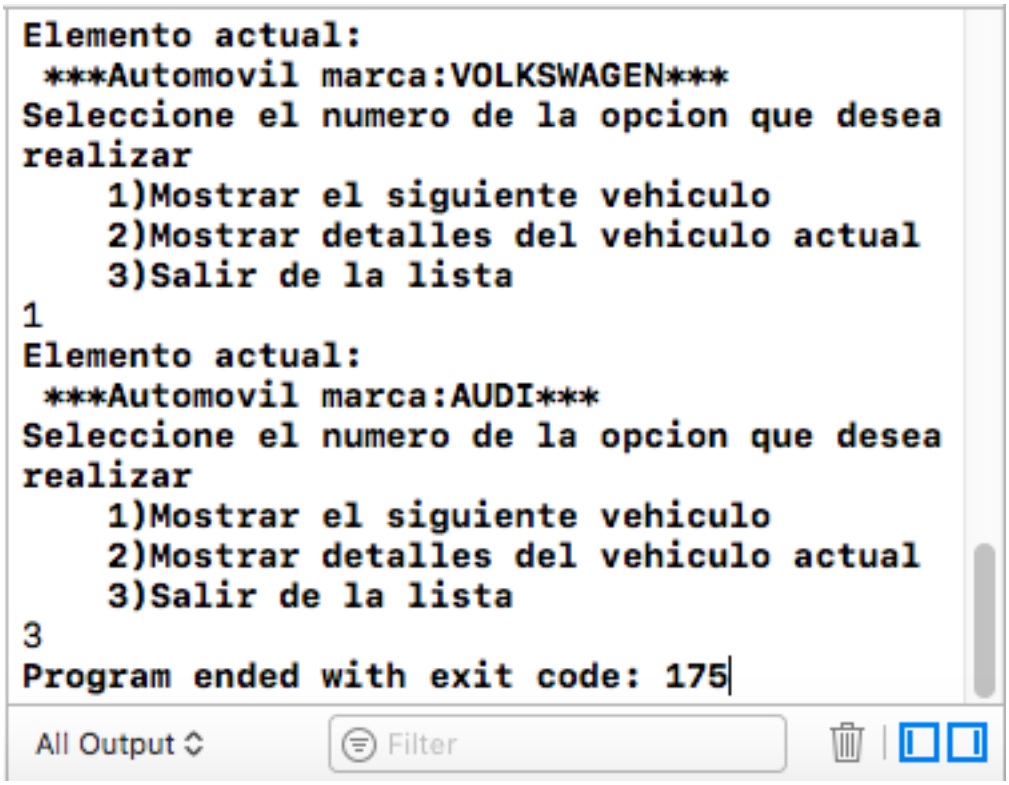


   b. Se pidió modificar la biblioteca de listacirc.h, con la finalidad de poder modelar automóviles, los cuales son un tipo de dato abstracto cuyos miembros que fueron implementados son: marca, modelo, color, placa y número máximo de pasajeros. Para realizarlo, se creó una nueva biblioteca con base en listacirc.h y se denominó como listacircAuto.h. Las modificaciones que se realizaron fue que, los nodos ahora tuvieran como valor a la estructura automóvil, con el propósito de modelar una lista circular de automóviles.

c. Se pidió realizar una función de búsqueda, la cual encuentre un automóvil por su marca. Esta función es bastante similar a la función de búsqueda de lista.h, la única variante es que ahora se introduce el nombre de la marca para buscar que nodo coincide con el nombre de la marca. En caso de que no se encuentre el automóvil, se notifica al usuario.
d. Se pidió realizar una función que fuera recorriendo cada nodo de la lista circular, con la particularidad de que se detuviera en cada nodo de la lista para que el usuario decida si quiere saber los detalles del automóvil que contiene el nodo, si quiere pasar al siguiente automóvil, o si bien, quiere salir de la lista circular. Para esta función, solamente se creó un pequeño menú, el cual se mostraba en cada nodo, para que el usuario escogiera la acción que quiere realizar.







- Conclusiones de la práctica

  En esta práctica se pudo cumplir el objetivo, pues se supieron implementar las listas ligadas simples y dobles en los programas solicitados. Asimismo, se comprendió de mejor manera las características de este tipo de estructuras lineales. Aunque se hayan visto en clase, hacerlo de manera práctica proporciona una mejor visión de su funcionamiento y características.

  Las listas ligadas simples, como las listas circulares, permiten un manejo de la memoria más eficiente. De igual forma, permiten mayor cantidad de funciones a realizar sobre ellas. Una de las funciones más importantes es la de búsqueda, porque podemos conocer si un elemento se encuentra en la lista.

  Cuando se implementan las listas simples y circulares en C, existe la dificultad de saber como utilizar los recursos que ofrece el lenguaje C para realizarlo, ya que en clase solo se ve de manera general.

  La práctica no fue tan compleja para poder realizarse, puesto que se encontró una gran relación con lo visto en clase. También la práctica permite que visualices como se realizan distintas funciones sobre la lista, llegando al punto de que tú crees ciertas funciones para lograr una mejor apreciación del manejo de listas.