	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): 5] ESTRUCTURAS DE DATOS LINEALES: PILA Y COLA

Integrante(s): MURRIETA VILLEGAS ALFONSO

Semestre: 2018 - 2

Fecha de entrega: 19 DE MARZO DE 2018

Observaciones:

CALIFICACIÓN: _____

ESTRUCTURAS DE DATOS LINEALES: PILA Y COLA

1] Introducción

Las estructuras de datos son una forma particular de organizar datos para facilitar y manejar de manera más eficiente datos.

Como anteriormente hemos visto, el uso de estructuras de datos consiste en la colección de nodos o registros del mismo tipo que mantienen relaciones entre sí. Cuando hacemos referencia a nodo, hablamos de la unidad mínima de almacenamiento de información en una estructura de datos.

Por otra parte, al momento de referirnos a estructuras de datos lineales indicamos que son aquellas donde los elementos ocupan lugares sucesivos en la estructura y cada uno de ellos tiene un único sucesor y un único predecesor.

En la presente práctica se analizarán y utilizarán dos estructuras de datos comunes, es el caso de la pila y la cola.

PILA

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.

Algunas características particulares de la pila son:

Este tipo de dato soporta operaciones de inserción y eliminación de elementos de un conjunto, su uso radica en aplicaciones en las que se necesita recuperar información en orden inverso a como ha entrado.

Además, contiene como miembros 3 índices enteros, para el mínimo (inicializado en 1), el máximo (límite del arreglo) y el tope de la pila (índice variable).

Por otro lado, las funciones que se emplean en todo el proceso de creación y uso de una pila son;

1) La función ***crearPila*** que no requiere ningún parámetro, pero retorna una estructura tipo pila, además de que en esta el miembro "tope" es inicializado en 0 (sin elementos).

2) La segunda función se denomina ***esVacia*** (o ***isEmpty***), donde recibe como parámetro una pila. Dentro de esta función se compara el valor de tope a través de una condición if para así determinar si está vacía o no la pila.

3) También tenemos a la función ***clear*** la cual se encarga como dice su nombre de borrar a todos los elementos de la pila.

4) Por otro lado, tenemos a la función ***push*** y ***pop*** donde a ambas como parámetro se les da una pila, sin embargo, en el caso de ***push*** sirve para anexar elementos a la pila mientras que la función ***pop*** sirve para conocer el último elemento de la pila, algo importante de la función ***pop*** es que

esta “saca” al elemento.

5) Por último, está la función ***top*** la cual regresa el elemento que se encuentra en la parte de hasta arriba de la pila, sin embargo, y a diferencia de la función ***pop*** esta no saca al último elemento, solo lo visualiza.

COLA

La cola (o queue) es una estructura de datos lineal y es un tipo abstracto de datos de tipo FIFO (first in first out). en la cual el elemento obtenido a través de la operación ***eliminar*** está predefinido y es el que se encuentra al inicio de la estructura.

La cola es una estructura de datos de tamaño fijo y cuyas operaciones se realizan por ambos extremos; permite ***enqueue*** elementos al final de la estructura y permite eliminar elementos por el inicio de la misma. La función de ***enqueue*** también se le llama ***encolar*** y la función de ***eliminar*** también se le llama ***desencolar***.

Además de estar, algunas funciones que se emplean en todo el proceso de creación y uso de una cola son:

1) La función ***crearCola*** que no requiere ningún parámetro, pero retorna una estructura tipo cola.

2) La segunda función se denomina ***esVacia*** (o ***isEmpty***), donde recibe como parámetro una cola.

3) También tenemos a la función ***clear*** la cual se encarga como dice su nombre de borrar a todos los elementos de la pila.

4) Por otro lado, las funciones ***enqueue*** y ***dequeue*** las cuales requieren como parámetro una cola, y a través del uso de este, en el caso de ***enqueue*** se encarga de ingresar un elemento a la cola meintra que en el caso de ***dequeue*** se encarga de extraer el que está al inicio de la cola.

5) Por último, al igual que la función ***top,*** la función ***first*** se encarga de regresar de dar a conocer el primer elemento de la cola sin eliminarlo como es el caso de ***dequeue.***

2] Objetivo de la práctica

- Se revisará las definiciones, características, procedimientos y ejemplos de las estructuras lineales Pila y Cola, con la finalidad de que se comprenda sus estructuras y así se pueda implementar.

3] Desarrollo

3.1] Análisis de la teoría

En la presente práctica a se emplearán principalmente dos estructuras de datos lineales que son la ***pila o stack*** y la ***cola o queue***.

De manera general podemos notar que en ambas estructuras se pueden realizar inserción y eliminación de datos, sin embargo, para poder llevar a cabo estas acciones tanto en la pila como en la cola se necesitan de elementos como son el caso de índices y listas (Elementos ya sean de tipo primitivo o abstracto).

Dicho lo anterior, también podemos ver que tanto en la pila como en la cola existen distintos casos en los que se presentan estos tipos de estructura de datos;

El primer caso es cuando estos se encuentran vacíos y es a través de las funciones mencionadas previamente que podemos ingresar o insertar elementos.

El segundo caso es cuando estas estructuras se encuentran llenas, para ello por ejemplo en el caso de la pila hacemos uso de la función ***pop*** la cual puede extraer los elementos de la pila.

Por último, encontramos el caso 3 que es aquel donde simplemente no estamos en ningún extremo, solamente se tienen elementos dentro de la pila o cola, en este caso se pueden realizar cualquiera de las funciones relacionadas con el tipo de estructura.

3.2] Análisis de los problemas (Práctica)

EJERCICIO I

En este ejercicio se nos pide analizar y completar un programa en el cual le hacen falta algunas opciones dentro de las funciones que emplea.

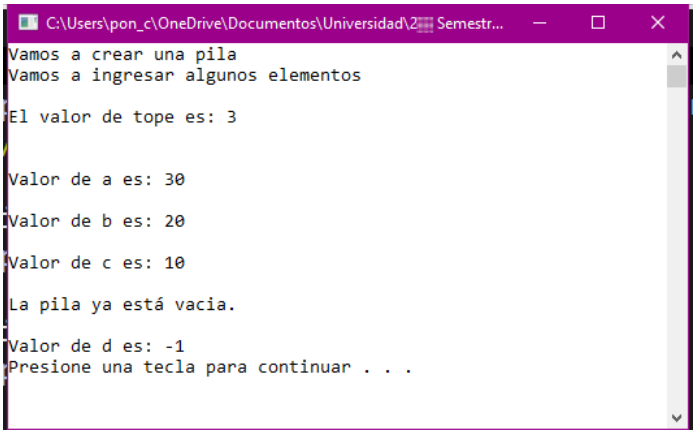
Para ello, lo primero que debemos hacer es determinar para que sirve cada una de las funciones empleadas, a continuación, se puede ver todas las funciones que son ocupadas dentro del programa.

```
19  Pila crearPila();//Crea una pila
20  int isEmpty(Pila);//verifica si la pila tiene elementos o no
21  void meter(Pila*,int);//Agrega valores a la pila
22  int sacar(Pila*);//Saca el valor que está en el tope de la pila
23  int top(Pila);// Da a conocer el valor que está hasta arriba de la pila
```

Lo primero que podemos notar es que para poder hacer uso de la función sacar pila necesitamos de la función ***isEmpty*** ya que debemos corroborar si nuestra pila está vacía o no, también y para poder extraer el elemento que se encuentra hasta arriba de la pila, es necesario de una variable auxiliar la cual alberga del dato extraído.

Por último, dentro de la función int se agregaron las líneas faltantes para que se pudiera imprimir los datos pedidos.

A continuación, se muestra la salida final del programa con las correcciones e inclusiones de líneas de códigos mencionadas previamente:



EJERCICIO II

En este ejercicio en base al programa pila.c se pedía crear una pila, agregar instrucciones para solicitar valores que se debían guardar en la pila creada anteriormente, y además se debía crear 2 pilas adicionales con el fin de determinar el valor mayor ingresado por el usuario. Otro aspecto relevante que se consideró al momento de programar fue que no se podía utilizar variables para almacenar los datos pedidos por el usuario.

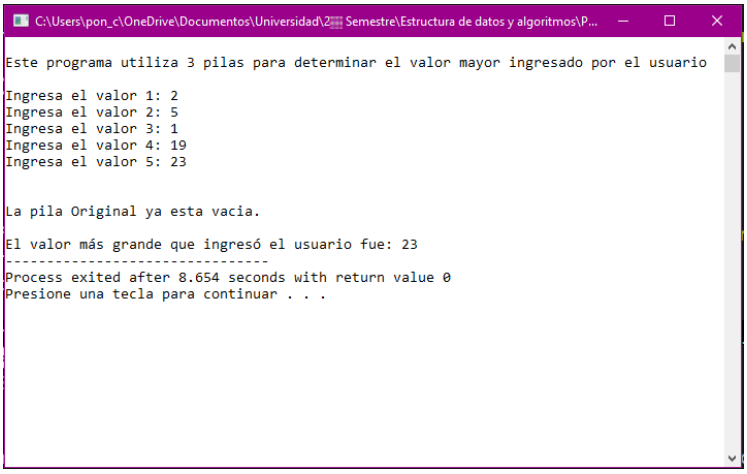
Lo primero que podemos notar es que la primera pila sirve para poder almacenar los datos pedidos al usuario, sin embargo, el punto principal de este ejercicio está en cómo emplear las otras 2 pilas para poder determinar el valor más grande ingresado por el usuario.

Para ello, lo que decidí utilizar fue una función la cual a través del paso por referencia y el uso de apuntadores pudiera hacer más fácil el uso de las pilas adicionales, las cuales sin duda alguna iban a servir principalmente para poder almacenar y comparar los datos ingresados por el usuario.

```
68  void comparacion(Pila *pilaOriginal, Pila *pilaMayor, Pila *pilaMenor){
69
70      //Parte comparativa entre los topes de cada una de las pilas
71      do{
72          if(pilaOriginal->lista[pilaOriginal->tope-1]<pilaMayor->lista[pilaMayor->tope-1]){
73              push(pilaMenor,extraer(pilaOriginal));
74          }
75
76          else
77          if(pilaOriginal->lista[pilaOriginal->tope-1]>=pilaMayor->lista[pilaMayor->tope-1]){
78              push(pilaMayor,extraer(pilaOriginal));
79          }
80
81      }while(top(*pilaOriginal)!=-1);
82
83      /*NOTA: Recordemos que en el código del ejercicio 1 usamos el -1
84      para determinar si estaba o no vacia la pila.
85      Este while para cuando la pila está vacía
86      */
87  }
```

Como se puede apreciar en el segmento de código anterior, a través de la función comparación y mediante el uso de las 3 pilas fue así como se pudo realizar comparaciones entre los distintos valores ingresados por el usuario.

A continuación, se muestra la salida final del programa:



EJERCICIO III

El programa *cola.c* hace uso de las mismas funciones que fueron mencionadas en la introducción de la presente práctica como es el caso de ***crearcola*** e ***isEmpty***, sin embargo, las diferencias entre la teoría y el programa se encuentran en las funciones ***encolar*** y ***desencolar***, donde a través del paso por referencia (Hace uso de apuntadores a la cola) de tal forma que los operadores empleados en estas funciones son el operador flecha para acceder y modificar a los miembros que se encuentran en la cola.

Dicho de otra forma, realmente lo que cambia en este código con respecto a lo visto en la clase o en el manual de prácticas es sobretodo el uso de apuntadores los cuales a su vez afectan de manera indirecta los operadores empleados en el código.

NOTA: Podemos darnos cuenta de que la función top realmente no tiene nada dentro además de que no es usada durante la ejecución del programa.

EJERCICIO IV

En el presente ejercicio lo que se solicita es que mediante una cola se guarden estructuras del tipo documento las cuales deben datos como el nombre del documento, tamaño en Kilobytes y el número de páginas del documento, además de que debe ser dado un supuesto tiempo de impresión del documento con los datos dados.

Lo primero que podemos ver es que debemos declarar dos tipos de datos abstractos donde el primero es del tipo cola el cual dentro de sus elementos se encuentra otro tipo de dato abstracto que es del tipo documento.

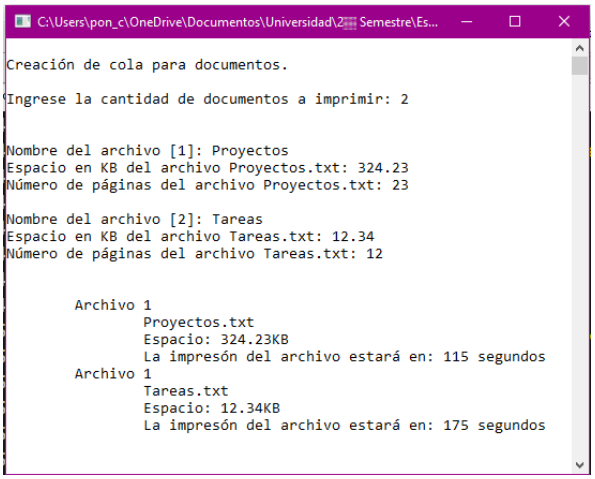
Posteriormente, a través de un análisis detenido de todos los elementos que se piden y se necesitan se crearon las funciones necesarias para resolver este ejercicio, las cuales se pueden ver de manera general en la siguiente imagen.

```
27  //funciones usadas durante todo el programa
28
29  Cola crearCola();//Se encarga de crear una cola
30  Documento crearDocumento(int i);//Se encarga de crear un documento
31  int isEmpty(Cola);//Sirve para saber si la cola está vacía
32  void encolar(Cola*,Documento x);//Ingresa el valor del documento a la cola
33  Documento descolar(Cola*);//Extrae el documento del incio de la cola
```

Como es habitual en los ejercicios de esta práctica debemos hacer uso de funciones como ***isEmpty*** o ***crearcola*** las cuales son de suma importancia para poder llevar acabo los ejercicios.

Por último, debido a que entre los requisitos del programa se requiere pedir al usuario la cantidad de documentos a imprimir para ellos se utilizó una función encargada del llenado de los datos, además de un ciclo for dentro de la función ***main*** para así poder ir llamando a la función ***crearDocumento*** tantas veces como el usuario necesite.

A continuación, se muestra la salida del programa:



EJERCICIO V

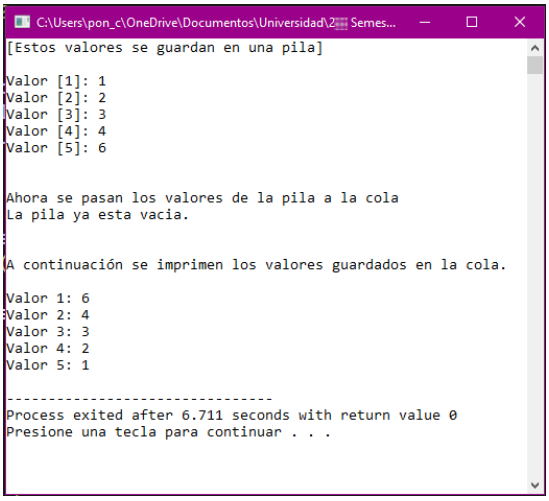
En este ejercicio lo que se nos pide de manera general es que el usuario ingrese 5 elementos los cuales sean guardados dentro de una pila y posteriormente mediante el uso de una cola y sin el uso de variables auxiliares se puedan invertir los valores ingresados por el usuario.

Lo primero que debemos destacar es que para poder hacer uso de una pila y cola será necesario utilizar las funciones básicas como son ***crearpila, isEmpty*** entre otras más.

Por otra parte, y como punto esencial de este ejercicio es como emplearemos o crearemos un algoritmo para la inversión de los datos ingresados por el usuario.

Lo primero que podemos notar es el concepto de cola y pila donde sabemos que en la pila debido a su naturaleza podemos extraer elemento tras elemento de esta, para ventaja sabemos que la manera en que se extraen elementos de la pila es desde el último ingresado hasta el primero, por otro lado, sabemos que la manera en que se van guardando los datos dentro de una cola es de orden ascendente, por lo tanto, lo único que debemos hacer en este código es simplemente ir extrayendo elemento por elemento de la pila para posteriormente ir guardándolos en una cola, además de que de esta manera de forma indirecta iremos invirtiendo los valores debido a la naturaleza de guardado de ambas estructuras.

Dada la reflexión previa, y una vez codificadas las funciones y elementos necesarios para el programa, obtuvimos como salida del programa la siguiente.



3.3] Relación entre los ejercicios y el tema

Los conjuntos (colecciones de datos) son tan fundamentales para las ciencias relacionadas con la computación. El uso de estructuras lineales como son las pilas y las colas son un claro ejemplo de cómo mediante la aplicación de tipos de datos abstractos podemos realizar un sinfín de cosas.

En el caso de la pila una de las principales aplicaciones o usos se encuentra en la memoria RAM
Por ejemplo, cuando una aplicación inicia, el método o función principal es invocado y se reserva memoria en la pila o stack.

En el segmento de memoria de la pila es donde se alojan las variables requeridas por las funciones del programa. Así mismo, cada vez que se llama una función dentro del programa una sección de la pila, llamada ***marco*** o ***frame***.

Por otro lado, la aplicación más conocida de la estructura cola es la que se utiliza en la impresión de documentos.

Debido a que las impresoras tienen una cantidad de memoria limitada, la cola de impresión permite enviar documentos de gran tamaño, o varios documentos, a una impresora sin tener que esperar que se complete la impresión para seguir con la siguiente tarea. Es de esta forma que las impresiones se van realizando según vayan llegando los datos.

4] Conclusiones

En la presente práctica se pudo apreciar ya de manera más concreta el primer acercamiento a las estructuras de datos lineales, donde a través de la teoría y uso de dos de las principales estructuras que son la pila y cola notamos la relevancia que tienen este tipo de estructuras en el mundo de la programación.

Es el caso de los ejercicios 1 y 3 donde pudimos ver de manera general como se estructuran las pilas y colas además de las funciones que son elementales para el uso de este tipo de estructura lineal como es el caso de la función ***isEmpty*** la cual constantemente se utiliza para verificar el estado actual ya sea de una pila o de una cola, además de otras funciones que comparten ambas estructuras.

Por otro lado, en el caso del ejercicio 2 y 4 a través de la reflexión y análisis de los requisitos y elementos que se pedían tuvimos que determinar de qué manera podíamos llevar acabo los algoritmos necesarios para poder dar una solución a los ejercicios, sobretodo en el caso del ejercicio 2 donde realmente tuve que pensar bastante en cómo podía llevar acabo la comparación de los elementos ingresados por el usuario a través de pilas.

Por último, pese el ejercicio 5 me pareció realmente fácil creo se debe a toda la capacidad reflexiva que fui adquiriendo a través de la práctica y es que realmente el ejercicio 5 lo único que se debía hacer era pensar un poco en cómo se ordenan los datos en las pilas y colas y de esta forma simplemente ir copiando los elementos de una a otra.

Sin duda esta ha resultado la práctica más larga de todas , sin embargo, a mi parecer ha sido una de las más interesantes ya que esta es la forma más común donde podemos implicar todos los conocimientos adquiridos en prácticas previas.

5] Referencias

1) El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.

2) Curso de Programación C7 C++. Javier Ceballos, tercera edición. USA, Ra – Ma. Educación 2007