

```

#include<stdio.h>
#include<stdlib.h>

Struct Datos
{
    Char Identificador [30];
    Char Proyecto [30];
    Char Diego [30];
    Char matricula [30];
    Char Peter [25];
    Char matricula [25];
    Char Grado [5];
    Char Materia [25];
    Char Entrega [20];
    Char Maestro [30];
} Info = "Ep2.Evidencia de producto 2"
        "Uso de Listas"
        "Hurtado Olivares Diego Gustavo"
        "Hodo140402"
        "Oropeza Martínez Peter Savier"
        "OMPI152172"
        "3-B"
        "Estructura de Datos"
        "17 de Noviembre del 2015"
        "M. en C. Roberto Enrique López Díaz"
};

Int main () {

Return 0;
}

```

Void Índice (Reporte) {

Portada.....1

Introducción.....3,4

Marco Teórico.....5,6

Diseño de la Solución.....7-17

Pruebas.....18-21

Conclusiones.....22

Referencias Bibliográficas.....23

}

## Introducción:

En esta práctica se desarrollará un programa para la administración de pacientes de un Centro médico, en el área de urgencias. En la sala de urgencias, los pacientes son atendidos dependiendo del tipo de emergencia que el paciente requiera; Atención inmediata, atención rápida y atención pronta. A su vez como vayan llegando los pacientes, para cada gravedad de emergencia existe una fila. El hospital consta de 2 doctores de planta para atender a los pacientes, pero si se llegara a necesitar más doctores entonces se puede llamar a uno de los doctores o a los 2 doctores. Esto depende de la cantidad de los pacientes que se encuentran en espera, si se llama a un doctor o se llama a los 2. Se realizará un registro de todos los pacientes, para cada paciente se le pedirá su nombre completo, el padecimiento que presenta y el tipo de emergencia que presenta el paciente. También se realizará un registro de los nombres de los doctores de cabecera y se es necesario a los doctores auxiliares. Durante el programa se podrá visualizar la fila de espera de cada tipo de emergencia, a su vez también todos los pacientes que se hayan atendido. Una vez que se vaya atender al paciente se puede elegir con que doctor se desea atender, si se llegara a necesitar un doctor auxiliar entonces se diría con cuál de los doctores se desea atender, una vez hecho

esto se atenderá dependiendo del tipo de emergencia y del tiempo en el que llegó el paciente para que el primero en llegar sea el primero en ser atendido. Para cada doctor se realizará un registro de todos los pacientes que haya atendido.

## Marco Teórico:

Una lista es una colección de estructuras auto referenciadas, llamadas nodos, conectadas por medio de ligas de apuntadores. Se accede a una lista ligada a través de un apuntador al primer nodo de la lista. Se accede a los nodos subsiguientes a través del miembro liga almacenado en cada nodo. Por convenciones apuntador liga del último nodo de una lista se establece en NULL para marcar el final de la lista. Un nodo Puede contener datos de cualquier tipo, incluso otros objetos STRUCT. Las pilas y las colas también son estructuras de datos y son versiones restringidas. Las listas de datos puede almacenarse en arreglos, pero las listas ligadas proporcionan muchas ventajas. Una lista ligada es adecuada, cuando el número de elementos a representarse en la estructura de datos es impredecible.

Las pilas son importantes para los compiladores y los sistemas operativos; las inserciones y las eliminaciones se hacen solo en un extremo de la pila, esto es, en la cima. Las colas representan líneas de esperas; las inserciones se hacen en la parte final (también conocido como talones) de una cola, y las eliminaciones se hacen de la parte inicial (También conocido como cabeza) de una cola.

Una pila es una versión restringida de una lista ligada. Los Nuevos nodos Pueden añadirse y eliminarse de una pila solo por la cima. Por esta razón, a una pila se le reconoce como una estructura de datos ultima en entrar, primera en salir (LIFO). A la pila se le hace referencia por medio de un apuntador hacia el elemento en la cima de la pila. Las funciones básicas que se utilizan para el uso de una pila es empujar (Push) y sacar (POP).

Las colas son otra estructura de datos de acceso restringido, En ella solo se encuentran dos operaciones básicas: Insertar un elemento al principio de la cola y eliminar el elemento final de la cola. Los nodos de una cola solo se eliminan solo de la cabeza de la cola, y se insertan solo en los talones de ella. Por esta Razón, a una cola se le conoce como una estructura de datos primera en entrar, primera en salir (FIFO). Las operaciones de insertar y eliminar se conocen como agregar en la cola y retirar de la cola.

## Diseño de la Solución ejercicio:

### Entrada:

- Capturar nombres de médicos
- Capturar información del Paciente
- Asignación de paciente para atención

### Proceso:

Primero se capturan los nombres de los 2 doctores de cabecera. Una vez hecho esto se irá capturando la información de cada paciente, para cada paciente se le pedirá su nombre completo, el padecimiento que presenta y el tipo de emergencia que presenta el paciente. Los Pacientes son atendidos dependiendo del tipo de emergencia que el paciente requiera; Atención inmediata, atención rápida y atención pronta, Una vez hecho esto a cada paciente se le asignara a un fila de espera dependiendo del tipo de emergencia que requiera. El hospital consta de 2 doctores de planta para atender a los pacientes, pero si se llegara a necesitar más doctores entonces se puede llamar a uno de los doctores o a los 2 doctores. Esto depende de la cantidad de los pacientes

que se encuentran en espera, Si la cantidad de los pacientes en espera es de 15 o el número de pacientes de atención inmediata es 5 personas entonces se agregara un doctor auxiliar, sino si la cantidad de los pacientes es de 20 personas o el número de pacientes en espera inmediata es mayor a 5 personas y el número de personas que esperan en la fila de espera rápida son 5 entonces se agregaran los 2 médicos. Se realizara un registro de todos los pacientes, para cada paciente se le pedirá su nombre completo, el padecimiento que presenta y el tipo de emergencia que presenta el paciente. También se realizara un registro de los nombres de los doctores de cabecera y se es necesario a los doctores auxiliares. Una vez que se valla atender al paciente se puede elegir con que doctor se desea atender, si se llegara a necesitar uno doctor auxiliar entonces se diría con cuál de los doctores se desea atender, una vez hecho esto se atenderá dependiendo del tipo de emergencia y del tiempo en el que llego el paciente para que el primero en llegar sea el primero en ser atendido. Para cada doctor se



realizara un registro de todos los pacientes que haya atendido.

### Salida:

- Se mostrara los Pacientes en espera
- Se mostrara el Listado de pacientes atendidos

## Algoritmo ejercicio

1. Inicio
2. Se muestra en pantalla el menú
3. Si la opción escogida es igual a 1 “Capturar nombres de médicos” entonces se pedirá que introduzca el nombre del doctor al usuario y el nombre del doctor se guardará en un arreglo y este a su vez en una estructura donde se guarda todos los registros del doctor. El número máximo de doctores de cabecera son 2, si se desea agregar otro doctor o más doctores primero se tiene que verificar que Si la cantidad de los pacientes en espera == 15 | el número de pacientes de atención inmediata == 5 personas entonces se podrá agregar un tercer doctor, sino si la cantidad de los pacientes es == 20 personas | el número de pacientes en espera inmediata es > 5 personas & el número de personas que esperan en la fila de espera rápida son == 5 entonces se podrá agregar un cuarto doctor. Si el usuario desea agregar otro doctor entonces se mostrara un mensaje indicando que no se puede agregar otro doctor porque el número máximo de doctores es de 4.
4. Si la opción es igual a 2 “Capturar Paciente” entonces se pedirá que introduzca el nombre completo del paciente y el nombre del paciente se guardada en un arreglo, se

pedirá que introduzca el padecimiento del paciente y el padecimiento del paciente se guardada en un arreglo y por último se pedirá que introduzca el tipo de emergencia que requiere, Si su emergencia es inmediata entonces el paciente se pasara a la fila de espera de emergencia inmediata(1), sino si el tipo de emergencia del paciente es rápida entonces el paciente se pasara a la fila de espera de emergencia rápida(2), sino si su emergencia es pronta entonces el paciente se pasara a la fila de espera de emergencia pronta (3) y en tipo de emergencia se guardara en una variable. Todos los datos del paciente se guardaran en una estructura que tendrá toda la información del paciente.

5. Si la opción es igual a opción 3 “Mostrar Pacientes en espera” entonces se mostraran todos los pacientes que están en la sala de espera, Se mostraran los que están en la fila de emergencia inmediata, rápida y pronta.

6. Si la opción es igual a 4 entonces escogió la opción “Asignar paciente para atención” entonces se pregunta al usuario Con cual doctor desea asignar el paciente, Si elige la primera opción entonces escogió el primer doctor una vez que se eligió el doctor se asigna al paciente a ese doctor, Dependiendo el tipo de emergencia se pondrá en una fila de espera al paciente, Sino si escogió la segunda opción entonces escogió

el segundo doctor una vez que se eligió el doctor se asigna al paciente a ese doctor, Dependiendo el tipo de emergencia se pondrá en una fila de espera al paciente, sino y si, solo si hay un tercer doctor auxiliar entonces se podrá escoger el tercer doctor, una vez que se eligió el doctor se asigna al paciente a ese doctor, Dependiendo el tipo de emergencia se pondrá en una fila de espera al paciente y si escogió la opción y si, solo si esta requerido el doctor auxiliar entonces se asigna al paciente a ese doctor, Dependiendo el tipo de emergencia se pondrá en una fila de espera al paciente.

7. Si la opción es igual a 5 entonces escogió la opción “Listado de pacientes atendidos” entonces se mostrara todos los pacientes que fueron atendido por cada doctor, a su vez se mostrara el nombre, el padecimiento y tipo de emergencia: Emergencia inmediata, Emergencia rápida y emergencia pronta.

8. Si la opción es igual a 6 entonces escogió la opción “Salir”, se mostrara un mensaje indicando que desea salir y se terminara el ciclo para que ya no se ejecute el programa.

9. Fin

## Funciones y estructuras implementadas ejercicio.

Struct	
struct paciente	
<b>Descripción</b>	Contiene Toda la información del paciente
<b>Entradas</b>	<pre>char *nombre; char *padecimiento; int gravedad; struct paciente *siguiente; struct paciente *anterior;</pre>
<b>Salidas</b>	Ninguna

Struct	
struct paciente	
<b>Descripción</b>	Contiene Toda la información del Doctor
<b>Entradas</b>	<pre>int n_pacientes; char *nombre_doctor; struct paciente *ini; struct paciente *fin; struct doctor *siguiente; struct doctor *anterior;</pre>
<b>Salidas</b>	ninguna

Interfaz.h	
Menú();	
<b>Descripción</b>	Contiene Todas las opciones del programa y a su vez alimenta a las funciones.
<b>Entradas</b>	<pre> struct doctor **inicio struct doctor **fin struct paciente **ini struct paciente **fini </pre>
<b>Salidas</b>	No retorna nada; Muestra el menú;

Interfaz.h	
push_paciente();	
<b>Descripción</b>	Funcion para insertar nuevos pacientes,a su vez la información del paciente.
<b>Entradas</b>	<pre> struct paciente ***inicio struct paciente ***final int valor struct paciente *q struct paciente *p char *nombre char *problema </pre>
<b>Salidas</b>	No retorna nada; Inserta nuevos pacientes y su información.

Interfaz.h	
push_doctor();	
<b>Descripción</b>	Función para insertar nuevo doctor, a su vez la información del doctor.
<b>Entradas</b>	<pre>struct doctor ***inicio struct doctor ***final int valor char *nombre_doc</pre>
<b>Salidas</b>	No retorna nada; Inserta nuevo doctor y su información.

Interfaz.h	
asignar_paciente_a_doctor();	
<b>Descripción</b>	Función para asignar un paciente al doctor asignado
<b>Entradas</b>	<pre>struct doctor ***inicio struct paciente ***ini int *in int *rap</pre>
<b>Salidas</b>	No retorna nada; Inserta un paciente a un doctor determinado

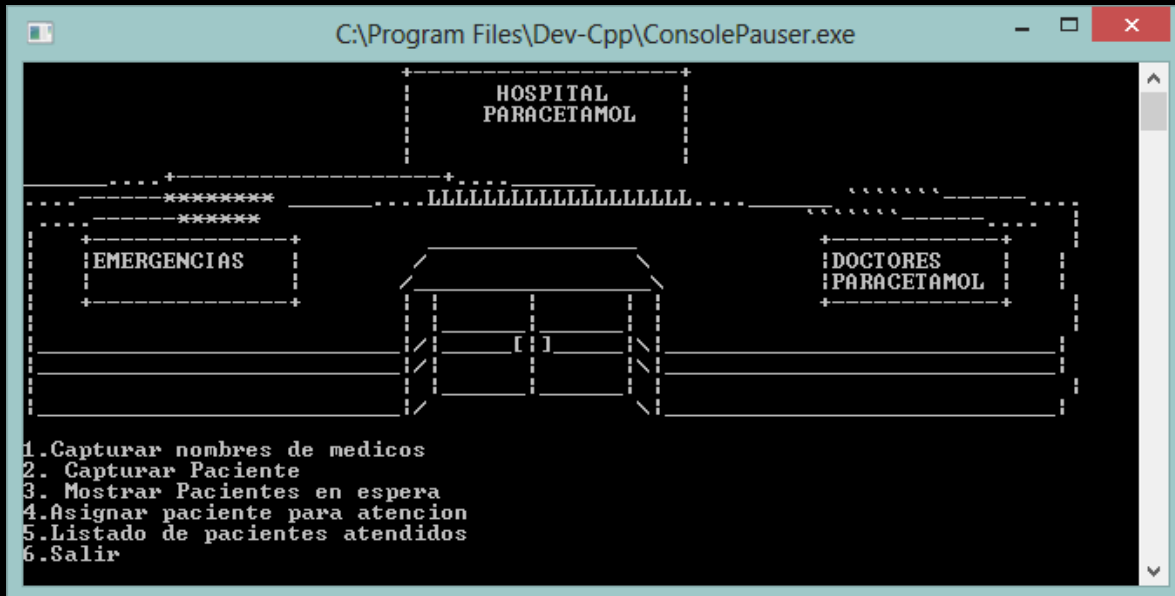
Interfaz.h		
imprimir();		
Descripción	Función para imprimir el contenido de toda la lista	
Entradas	struct doctor **inicio	
Salidas	No retorna nada; Muestra en pantalla el contenido de la lista.	

Interfaz.h		
vaciar_todo();		
Descripción	Función para liberar toda la memoria pedida(Elimina todo el contenido de la lista)	
Entradas	struct doctor ***inicio struct doctor ***fin struct paciente ***ini	
Salidas	No retorna nada; liberar toda la memoria pedida(Elimina todo el contenido de la lista)	



Interfaz.h		
imprimir_lista_espera();		
<b>Descripción</b>	Función para imprimir todos los paciente que se encuentran en espera	
<b>Entradas</b>	struct paciente **inicio	
<b>Salidas</b>	No retorna nada; Muestra en pantalla todos los pacientes en espera	

## Pruebas:



```
C:\Program Files\Dev-Cpp\ConsolePauser.exe

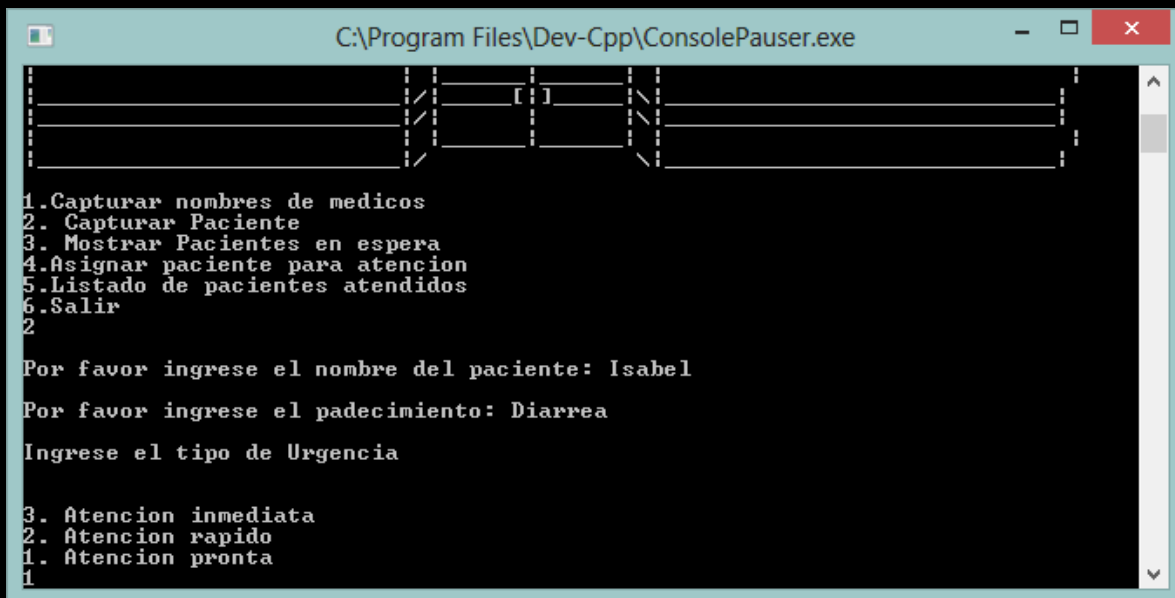
+-----+
| HOSPITAL |
| PARACETAMOL |
+-----+

*****
*****
+-----+
| EMERGENCIAS |
+-----+

+-----+
| DOCTORES |
| PARACETAMOL |
+-----+

1. Capturar nombres de medicos
2. Capturar Paciente
3. Mostrar Pacientes en espera
4. Asignar paciente para atencion
5. Listado de pacientes atendidos
6. Salir
```

//Se muestra en pantalla las opciones del sistema



```
C:\Program Files\Dev-Cpp\ConsolePauser.exe

+-----+
| DOCTORES |
| PARACETAMOL |
+-----+

1. Capturar nombres de medicos
2. Capturar Paciente
3. Mostrar Pacientes en espera
4. Asignar paciente para atencion
5. Listado de pacientes atendidos
6. Salir
2

Por favor ingrese el nombre del paciente: Isabel
Por favor ingrese el padecimiento: Diarrea
Ingrese el tipo de Urgencia

3. Atencion inmediata
2. Atencion rapido
1. Atencion pronta
1
```

//Se introduce un paciente

Su nombre, Su padecimiento y el tipo de urgencia.





```
C:\Program Files\Dev-Cpp\ConsolePauser.exe
+-----+
| HOSPITAL |
| PARACETAMOL |
+-----+
+-----+
| ***** |
| ***** |
| ***** |
+-----+
+-----+
| EMERGENCIAS |
+-----+
+-----+
| DOCTORES |
| PARACETAMOL |
+-----+
+-----+
| [ ] |
+-----+
+-----+
1.Capturar nombres de medicos
2. Capturar Paciente
3. Mostrar Pacientes en espera
4.Asignar paciente para atencion
5.Listado de pacientes atendidos
6.Salir
5

La Fila de Filas esta asi:
+-----+
| . |
| . |
| . |
| . |
+-----+

Doctor Diego <n pacientes: 1> -> <N: Danny , P: diarrea , G: 3> ->
||
+-----+
| . |
| . |
| . |
| . |
+-----+

Doctor Peter <n pacientes: 0> ->
||
```

//Se muestra el Doctor y los pacientes que haya atendido.

## Conclusiones:

Llegamos a la conclusión que las estructuras de datos son de muchísima ayuda en nuestro desarrollo de nuestros programas, ya que es una forma particular de organizar datos del programa de forma eficiente, nos permiten manejar grandes cantidades de datos de manera eficaz, nos permiten hacer un óptimo manejo de nuestra memoria de nuestro sistema, entre otras muchísimas cosas más. Pudimos apreciar cómo podemos hacer uso de las pilas, colas y listas. Así ves pudimos usar cada función de cada estructura para hacer manejo de la misma, Ya que cada una implementación de las estructuras tienen su manejo como las listas que pueden definirse estructuras más complejas a partir de las listas, como por ejemplo los arreglos de listas, etc. Pudimos apreciar que las listas son eficaces igualmente para diseñar colas de prioridad, pilas y colas sin prioridad, y en general cualquier estructura cuyo acceso a sus elementos se realice de manera secuencial, también pudimos apreciar el funcionamiento de las pilas que son importantes para los compiladores y los sistemas operativos, a su vez la pila que es una versión restringida de una lista ligada y la lista que es una colección de estructuras autor referenciadas, llamadas nodos, conectadas por medio de ligas apuntador.

## Referencias:

(Francisco A. Martínez Gil, 2003)  
//Introducción a la programación  
estructurada en C  
Martínez, Francisco A. (2003). "Introducción  
a la programación estructurada en C ".  
Universidad de valencia.

Deitel, Harvey M. y Deitel, Paul J.,  
(2004), "Cómo programar en C/C++ y Java",  
Pearson Educación.

Ma Luisa Garzon Villar, 2004),  
"Informática. Temario A. Volumen Ii.  
Profesores de Educación Secundaria del E-  
libro", MAD-Eduforma.