INSTITUTO POLITÉCNICO NACIONAL
UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERÍA Y TECNOLOGÍAS AVANZADAS

Práctica No. 7. Manejo de estructuras.

Unidad Temática: IV Manejo de Apuntadores y Estructuras.

Lugar de realización: Laboratorio de Cómputo **Duración:** 9.5 hrs.

Objetivo

Desarrollo de programas utilizando los apuntadores para el paso de parámetros por valor y por referencia a funciones.

Resultados Esperados

- Diferenciar el uso de memoria de estructuras y uniones, dada por el pre-reporte
- Manejo de estructuras simples y complejas
- Comprensión y manipulación de estructuras anidadas
- Identificación y uso de arreglos de estructuras y manejo de ellas en funciones

Pre-reporte.

Para el buen desarrollo de la práctica el alumno entregará un trabajo previo que incluya:

1. Definición y diferencia de estructura y unión.
2. Palabras reservadas para el uso de estructuras.
3. Definición y ejemplo de estructuras anidadas

Nota: El pre-reporte cuenta como calificación de la práctica, en caso de no entregar el pre-reporte se le restará 25% a dicha calificación.

"La mejor estructura no garantiza los resultados ni el rendimiento. Pero la estructura equivocada es una garantía de fracaso"
(Peter Drucker)

Material y Equipo.

- IDE Dev-Cpp o compatible.
- Computadora

Introducción.

Estructuras

Una **estructura** contiene varios datos ya que su función es agrupar datos de diferente o igual tipo o de manejar datos que serían muy difíciles de describir en los tipos de datos primitivos. La forma de definir una estructura es haciendo uso de la palabra clave `struct`. Los componentes individuales de una estructura se llaman **miembros**, y la estructura puede contener cualquier número de miembros con un nombre único.

Declaración:

struct nombre_estructura { tipo_dato miembros_de_estructura };

```
struct MiEstructura{
   int variable1;
   float varaible2;
   char* variable3;
};
```

La declaración especifica simplemente el nombre y el formato de la estructura de datos, pero no reserva almacenamiento en la memoria, para la definición de una variable (crear un área en memoria) en donde los datos se almacenan de acuerdo al formato estructurado se pueden definir de dos formas:

Definición:

A: listándolas inmediatamente después de la llave de cierre en la declaración

struct nombre_estructura { tipo_dato miembros_de_estructura }**variable**;

```
struct MiEstructura{
   int variable1;
   float variable2;
   char* variable3;
}var1, var2;
```

B: listando el tipo de la estructura creado, en cualquier lugar del programa

struct nombre_estructura nombre_de_variable;

```
struct MiEstructura var3;
```

Para dar valores iniciales a una estructura se utilizan llaves al momento de la definición, sin importar por cuál de las dos definiciones se haya optado pero respetando el orden en que fueron declarados los miembros de la estructura.

Inicializar:

struct nombre_estructura nombre_variable_estuctura = {valor1, valor2, valor3};

```
struct MiEstructura var4 = {4, 7.2, "Juan"};
```

Para tener acceso a los miembros de una estructura se usa el operador punto.

Acceso:

nombre_variable_estuctura.nombre_miembro

```
var4.variable2 = 38.51;
```

Desarrollo.

Ejemplo:

Lista de matrioshka. Escriba un programa en C con las siguientes características:

1. Crear la estructura de matrioshka que contenga el color (arreglo de caracteres), la inicial de su nombre (char) y el tamaño de la muñeca.
2. Generar 5 elementos con sus respectivas características, donde cada uno contendrá (apuntará) al siguiente de menor tamaño.
3. Desde la muñeca de mayor tamaño cambiar el tamaño a la más pequeña, haga uso de apuntadores.
4. Desde la muñeca de mayor tamaño cambiar el color a la tercera (usar apuntadores)
5. Imprimir los datos de las 5 muñecas rusas.

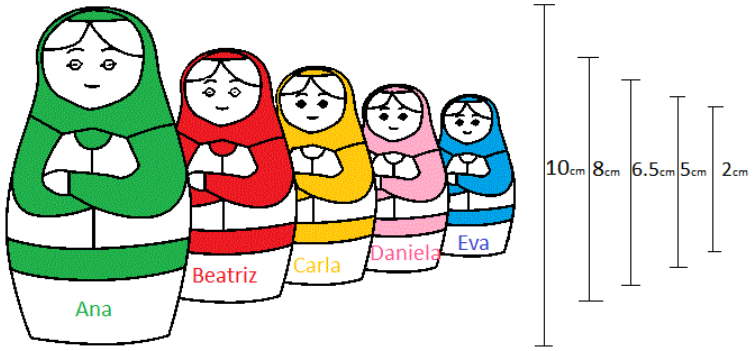

Ilustración 7.1. Ejemplo de matrioshka

Solución (Alternativa 1):

1. Creación de la estructura matrioshka

```
struct matrioshka{
    char nombre;
    float tamanio;
    char color[10];
};
```

2. Apuntadores y elementos
   a. Modificar la estructura para que pueda contener a otra matrioshka, apuntando a ella.

```
struct matrioshka{
    char nombre;
    float tamanio;
    char color[10];
    struct matrioshka* espacioHueco;
};
```

   b. Declarar 5 variables del tipo estructura matrioshka

```
struct matrioshka Ana; //puede declarar una por una
```

```
struct matrioshka Beatriz, Carla, Daniela; // declarar varias
struct matrioshka Eva={'E',2,"Azul"};//declaración y asignación
```

c. Colocar los datos de cada una de ellas

```
Ana.nombre='A';
Ana.tamanio=10;
strcpy ( Ana.color, "Verde" );
Beatriz.nombre='B';
Beatriz.tamanio=8;
strcpy ( Beatriz.color, "Rojo" );
Carla.nombre='C';
Carla.tamanio=6.5;
strcpy ( Carla.color, "Amarillo" );
Daniela.nombre='D';
Daniela.tamanio=5;
strcpy ( Daniela.color, "Rosa" );
```

d. Agregar a Eva dentro de Daniela, a Daniela dentro de Carla y así sucesivamente

```
Ana.espacioHueco=&Beatriz;
   //Ana tiene la referencia de un espacio hueco (espacio de memoria) que será ocupado por Beatriz,
   //Ana tiene el hueco (espacio) donde guardará la dirección (referencia) de que dentro de ella va Beatriz
   //Ana y Beatriz son independientes
Beatriz.espacioHueco=&Carla;
Carla.espacioHueco=&Daniela;
Daniela.espacioHueco=&Eva;
Eva.espacioHueco=NULL; //Dentro de Eva ya no va ninguna muñeca
```

3. Cambiar tamaño a Eva

```
Ana.espacioHueco->espacioHueco->espacioHueco->espacioHueco->tamanio=3;
```

Explicación:

```
Ana.espacioHueco->espacioHueco->espacioHueco->espacioHueco->tamanio=3;
```

Ana

Beatriz //en el espacioHueco de Ana está Beatriz. espacioHueco es miembro de la estructura Ana y accedemos con punto para hacer referencia a Beatriz.

Carla //espacioHueco de Beatriz es miembro de la estructura Beatriz y accederíamos con punto, pero como Beatriz es referenciada desde Ana a través de su espacio de memoria, entonces Beatriz es un apuntador por lo que a los miembros de Beatriz accedemos con flecha. Es el equivalente a tener `Beatriz->espacioHueco` para hacer referencia a Carla, donde Beatriz es un apuntador

Daniela

Eva

El tamaño de Eva

4. Cambiar color a la tercera

```
strcpy(Ana.espacioHueco->espacioHueco->color, "Morado" );
```

5. Imprimir

```
printf("Dir. de Ana %p \n", &Ana);
```

```
printf("Nombre %c \n",Ana.nombre);
printf("Tam %.1f \n",Ana.tamanio);
printf("Color %s \n",Ana.color);
printf("Dir. de espacioHueco %p \n\n", Ana.espacioHueco);

printf("Dir.de Beatriz %p\n",&Beatriz);//debe coincidir con la del espacioHueco de Ana
printf("Nombre %c \n",Beatriz.nombre);
printf("Tam %.1f \n",Beatriz.tamanio);
printf("Color %s \n",Beatriz.color);
printf("Dir.      de      espacioHueco      %p      \n\n",
Beatriz.espacioHueco);

printf("Dir. de Carla %p\n", &Carla);//debe coincidir con el espacioHueco de Beatriz
printf("Nombre %c \n",Carla.nombre);
printf("Tam %.1f \n",Carla.tamanio);
printf("Color %s \n",Carla.color);
printf("Dir.      de      espacioHueco      %p      \n\n",
Carla.espacioHueco);

printf("Dir. de Daniela %p\n", &Daniela);//debe coincidir con el espacioHueco de Carla
printf("Nombre %c \n",Daniela.nombre);
printf("Tam %.1f \n",Daniela.tamanio);
printf("Color %s \n",Daniela.color);
printf("Dir.      de      espacioHueco      %p      \n\n",
Daniela.espacioHueco);

printf("Dir. de Eva %p \n", &Eva); //debe coincidir con la del espacioHueco de Daniela
printf("Nombre %c \n",Eva.nombre);
printf("Tam %.1f \n",Eva.tamanio);
printf("Color %s \n",Eva.color);
printf("Dir. de espacioHueco %p \n\n", Eva.espacioHueco);
```

Solución (Alternativa 2): *Ver Código 7matrioshkaOp2.c*

Programas

Realice los siguientes ejercicios:

1. Se requiere almacenar datos de una colección de discos compactos (CD) de música. Estos datos serán: título, artista, número de canciones, precio y fecha de compra. Para ello necesitará crear dos estructuras:
    a. La estructura Fecha que tenga como enteros: anio, mes y dia.
    b. La estructura Disco que tenga: arreglo de caracteres para el título y artista, un entero para el número de canciones, un flotante para el precio y la fecha de creación (del tipo Fecha creado previamente).
2. Para uso de las estructuras anteriores escriba las siguientes funciones:

a. Función que permita solicitar al usuario los datos, cuyo prototipo sería:
```
Disco solicitarDatos();
```
Argumentos de Función:
Ninguno
Procedimiento:
1) **Declarar una estructura `Fecha`**
2) **Pedir los datos de `Fecha`**
3) **Declarar una estructura de `Disco`**
4) **Agregar la estructura `Fecha` previamente declarada a la estructura Disco**
5) **Pedir los datos de estructura `Disco`**
6) **Regresar el `Disco` como paso por valor**

Retorno de Función:
`Disco`

b. Una función que muestre en pantalla los datos de un CD
```
void imprimirDisco (Disco);
```
Argumentos de Función:
`Disco`
Procedimiento:
1) **Imprimir los datos de la estructura anidada `Fecha`**
2) **Imprimir los datos de la estructura `Disco`**

Retorno de Función:
Ninguno

c. Una función que calcule un descuento del 25% del precio de un CD
```
Disco descuento (Disco);
```
Argumentos de Función:
`Disco`
Procedimiento:
1) **Acceder al miembro de la estructura `Disco` llamado `precio`**
2) **Modificar el `precio` y darle un nuevo valor**
3) **Regresar el `Disco` como paso por valor**

Retorno de Función:
`Disco`

d. Escriba un función que reciba un arreglo de CD's y realice la suma de sus precios de venta.
```
float sumaVenta (Disco []);
```
Argumentos de Función:
Arreglo de Discos
Procedimiento:
1) **Inicializar `suma` en cero**
2) **Para cada disco del arreglo**
   a. **Obtener precio de ese `Disco` en el arreglo**
   b. **Sumar el precio a la variable `suma`**
3) **Regresar `suma`**

Retorno de Función:
float

3. Declare en el main un arreglo de 5 estructuras CD, las cuales el usuario podrá usar a través de un menú que contengas las opciones de:
   a. Ingresar datos
   b. Mostrar datos
   c. Aplicar descuento
   d. Obtener ganancia (suma)

Problema de Aplicación

"Robot in-house". Generar un programa que permita simular el monitoreo de iluminación y accesos a una casa habitación usando estructuras y funciones.
Se requiere monitorear:

- **Focos**. Se requiere saber si están colocados en un espacio con iluminación natural, si están o no prendidos y un rango de intensidad que va de 1 a 5, cuando se instalan por primera vez la intensidad es la máxima.
- **Puertas**. Se requiere saber si está abierta o cerrada la chapa y si ésta conecta hacia el exterior de la propiedad.
- **Ventanas**. Se requiere saber si está abierta o cerrada, si ésta conecta hacia el exterior y en qué nivel (piso) de la casa se encuentra.

El usuario tendrá las siguientes opciones:

1. **Inicializar casa.** El usuario ingresa la cantidad de focos, puertas y ventanas que tiene su casa, así como las características de ellos. (solo se realizará al inicio, una sola vez)
2. **Salir de casa**
    a. Salir por un periodo breve. Revisar que las puertas a exterior estén cerradas, que las ventanas en planta baja que dan a exterior estén cerradas y poner los focos que estén encendidos en intensidad mínima.
    b. Salir por periodos largos de tiempo. Revisar que todo esté cerrado y focos apagados.
3. **Regresar a casa**. Regresar la configuración de puertas, ventanas y focos a la configuración en la que se encontraba antes de salir.
4. **Actualizar clima**. El usuario ingresa la hora del día, si llueve o no y si está nublado o no. Según lo que el usuario haya ingresado se debe configurar la casa de la siguiente manera:
    i. En caso de que esté soleado los focos colocados en habitaciones con iluminación natural se apagarán
    ii. Si está lloviendo todas las ventanas se cierran y se asume que está nublado.
    iii. Si es de día (de 8 am a 8pm) los focos con iluminación natural se apagan.
5. **Mostrar estado de la casa**. Imprimir el estado de focos, puertas y ventanas.

Propuesta de Solución.

- Crear cuatro estructuras para: foco, ventana, puerta y clima.
- Hacer un arreglo de estructuras para: foco, ventana y puerta. El tamaño del arreglo dependerá de la cantidad ingresada por el usuario de los elementos que tiene de focos, puertas y ventanas.
- Hacer funciones para: inicializar casa, cerrar todas las puertas, cerrar puertas a exterior, cerrar todas las ventanas, cerrar ventanas a exterior, cerrar ventanas a exterior en la planta baja, apagar todos los focos, apagar focos que están en cuartos con iluminación natural, guardar configuración anterior.
- Llamar las funciones según sea el caso

Proyectos (opcionales).

Obtener promedio de alumnos. Hacer un arreglo de Estudiantes los cuales se distinguen por su nombre y número de boleta, cada uno cursa 6 materias por semestre, obtener el promedio semestral y el del todos los alumnos registrados.

Control de asistencia de un empleado. Hacer una estructura Empleado, la cual almacene datos como nombre, fecha de nacimiento, sueldo semanal y horario de trabajo (el cual puede ser diferente para cada día de la semana). Para la fecha de nacimiento y el horario de trabajo se puede apoyar de otras dos estructuras. El programa permitirá registrar horas de entras y salida de un trabajador y calcular su pago semanal restando los días no asistidos u horas no laboradas.

Ponderación de la Práctica.

Sección	Elemento a Evaluar
Pre-reporte	Elementos básicos de estructuras y uniones
Ejercicio 1	Declaración de estructuras simples y anidadas
Ejercicio 2 a	Manejo de miembros de estructuras simples y complejas. Manejo de retornos de función
Ejercicio 2 b	Manejo de miembros de estructuras simples y complejas. Manejo de argumentos de función
Ejercicio 2 c	Uso de estructuras en funciones
Ejercicio 2 d	Uso de arreglo de estructuras y apuntadores a funciones
Ejercicio 3	Paso de parámetros
Problema de Aplicación	Interpretación del problema, integración y programación de: estructuras, sentencias de control y funciones.

Bibliografía.

Deitel P.J.y Deitel H. M., Como Programar C++, Ed. Prentice Hall, 6 ª Impresión, México, 2009, ISBN: 970-26-1273-X, Págs: 1-1050.

Deitel P.J. y Deitel H. M., Como programar en C#, Ed. Prentice Hall, 2ª Impresión, México, 2007,ISBN: 9702610567, Págs: 1-1080.

Guardati, Silvia, Estructura de Datos Orientado a Objetos con C ++, Ed.Prentice, México 2007, ISBN: 9702607922, Págs: 1–183.

NOTA: Presentar el reporte en un documento (PDF o DOC), el código fuente y la impresión de la pantalla de ejecución.

"Puedes apoyarte de tus compañeros y profesor para aclarar tus dudas"