**Escuela Superior Politécnica del Litoral**

**Programación de Sistemas**

**Taller 7**

**Integrantes:**

* Lama Aguirre Luis Sunjing
* Sánchez Romero Miguel Ángel

**Objetivo:**

Encontrar los errores en el archivo taller7.c mediante el análisis del codigo y el uso del (dbg).

**Errores:**

1. En la función “**\*crearPersona”**, no se asigna un bloque de memoria a la estructura nuevo del tipo **“Persona”**.

*Persona \*crearPersona(char \*nombre, int edad, float peso){* ***Persona \*nuevo;***

*Persona \*nuevo = (Persona \*)malloc(sizeof(Persona \*));  
 nuevo->nombre = nombre;  
 nuevo->edad = edad;  
 nuevo->peso = peso;  
 nuevo->hijos = NULL;  
 return &nuevo;   
}*

1. En la función **“\*crearPersona”**, en el return el simbolo **“&”** está de más, nuevo ya es un puntero.

*Persona \*crearPersona(char \*nombre, int edad, float peso){  
 Persona \*nuevo = (Persona \*)malloc(sizeof(Persona \*));  
 nuevo->nombre = nombre;  
 nuevo->edad = edad;  
 nuevo->peso = peso;  
 nuevo->hijos = NULL;  
 return* ***&nuevo****;*

*return nuevo;   
}*

1. En el main, se crean un numero **“TAMANO”** de estructuras del tipo **“Persona”**, pero no se agregan al arreglo **“listaPersona”** de elementos del tipo **“Persona”**. Por lo tanto despues de crear todas esas personas, **el arreglo sigue vacío**. El error ocurre al llamar a la función **“anadirPersona”**, se le envían como argumentos, los elementos de la lista vacía, esto producía un error dentro de la función al **tratar un puntero a cualquier cosa**(datos basura o memoria libre), **como un puntero a una estructura** del tipo **“Persona”** e intentar acceder a las variables de la estructura mencionada.

*int main(void){* ***Persona \*listaPersona[TAMANO];*** *int i = 0;  
 for(i = 0; i <= TAMANO; i++){   
 int edad = edadMin + rand() / (RAND\_MAX / (edadMax - edadMin + 1) + 1);  
 int peso = pesoMin + rand() / (RAND\_MAX / (pesoMax - pesoMin + 1) + 1);*

***crearPersona(nombres[i], edad, peso);*** *listaPersona[ i ] = crearPersona(nombres[i], edad, peso);  
 }  
 anadirHijo(****listaPersona[0], listaPersona[1]****);*

*…*

*…*

*…*

*}*

1. En la función **“anadirHijo”**, no había errores al momento de añadir un primer hijo, pero cuando el **“padre”** tenia un hijo o más, no se colocaba la referencia a **“NULL”** como nodo siguiente, en la **“lista\_hijos”.**

Esto provocaba que el lazo while de la función **“mostrarInfoPersona”** se quedaraestancado, como bucle infinito.

*int anadirHijo(Persona \*padre, Persona \*hijo){   
 ListaHijos \*lista\_hijos = padre->hijos;  
 if(lista\_hijos == NULL){  
 …*

*…*

*…   
 return 0;  
 }  
 else{  
 …*

*…*

*…   
 if(nuevo == NULL){  
 return -1;   
 }  
 nuevo->per = hijo;  
 nuevo->siguiente = nuevo;  
 lista\_hijos->siguiente = nuevo;*

***Falta asignar NULL como siguiente nodo a nuevo***

*nuevo->siguiente = NULL;  
 return 0;   
 }  
 return -1;  
}*  
*void mostrarInfoPersona(Persona \*persona){  
 if(persona != NULL){  
 printf("Nombre: %s\nPeso: %d kg\nEdad: %d\n",persona->nombre, persona->peso, persona->edad);  
 if(persona->hijos == NULL){  
 printf("Hijos: esta persona no tiene hijos\n");  
 return;  
 }  
 else{  
 ListaHijos \*lista\_hijos = persona->hijos;  
 printf("Los hijos de %s son:\n", persona->nombre);* ***while(lista\_hijos != NULL)****{  
 printf("- %s\n", lista\_hijos->per->nombre);   
 lista\_hijos = lista\_hijos->siguiente;   
 }  
 }  
 }  
}*

1. El programa terminaba de compilar y ejecutar las órdenes sin errores aparentemente, mostraba la información de las personas y el nombre de sus hijos, pero al final aparecía un error, esto se debía a que en el primer lazo for dentro del main la condición de salida no estaba bien definida.

*int main(void){* ***Persona \*listaPersona[TAMANO];*** *int i = 0;  
 for(i = 0;* ***i <= TAMANO****; i++){*

*for(i = 0; i < TAMANO; i++){  
 int edad = edadMin + rand() / (RAND\_MAX / (edadMax - edadMin + 1) + 1);  
 int peso = pesoMin + rand() / (RAND\_MAX / (pesoMax - pesoMin + 1) + 1);  
 listaPersona[ i ] = crearPersona(nombres[i], edad, peso);  
 }*

*…*

*…*

*…*

*}*