Práctico Nº 5
Tema: Estructuras en C

Nota: Todos los programas deben ser realizados usando Eclipse. Crear un espacio de trabajo (workspace) Practico5, y para cada ejercicio crear en dicho espacio de trabajo un proyecto Prac5_ejxx. A partir del Ej.13 realizar los programas con funciones.

Ej.1- ¿Qué significa la sigla IDE? ¿Qué es un IDE? ¿Cuáles son las partes que contiene un IDE? ¿Qué es Eclipse? ¿Qué es un *workspace* en Eclipse?

Ej.2- Abrir la aplicación Eclipse. ¿En qué consiste el entorno de trabajo de Eclipse? ¿Qué nombre recibe una colección de paneles o vistas? Analizar la vista *Explorador de Proyectos*.

Ej.3- ¿A qué se denomina **proyecto** en esta aplicación? Crear en el workspace Practico5 de su home, un Proyecto C "Hello World ANSI C Project", llamado **Prueba**. Analizar los cambios producidos en el Explorador de Proyectos, en el Editor y en la Vista Esquema.

Ej.4- Modificar **Prueba.C**, reemplazando la instrucción **puts** por **printf** y que la función retorne un 0. Grabar los cambios y compilar el proyecto. ¿Qué cambios se produjeron en el *Explorador de Proyectos*? Ejecutar este proyecto.

Ej.5- Visualizar los videos disponibles en el Campus.

Ej.6- Crear un proyecto vacío en Practico5 un proyecto Prac5_ej6

(1) **Recuperar** de **/home/serv/serv99/practico5/** el archivo **p5_ej6.c** y dejarlo en el directorio del proyecto recién creado.
(2) **Renovar(F5)** en Eclipse *(Explorador de Proyectos-Botón derecho en el proyecto Prac5_ej6-Renovar).*
(3) **Editar** el archivo **p5_ej6.c**
(4) **Compilar** *(Proyecto >> Construir Proyecto)*, y leer los errores marcados. Corregir los errores de sintaxis hasta conseguir el ejecutable.

Nota:*(El ejecutable es creado cuando el programa no tiene errores, pero puede tener avisos (warnings)).*

(5) **Ejecutar** (Construir >> Ejecutar). ¿Funciona correctamente este programa?
(6) **Identifique** en la barra de herramientas los íconos para realizar todas las acciones anteriores.
(7) ¿**Cuál** es la opción para depurar un programa? Iniciar la depuración. ¿La aplicación cambia de perspectiva? Analizar todos los paneles, especialmente las vistas de *Variables* y *Puntos de Interrupción*.
(8) ¿Existió alguna diferencia entre ejecutar y depurar? Justifique su respuesta.
(9) ¿Qué es colocar un punto de interrupción o **Breakpoints** en un programa? ¿Cómo se colocan y se eliminan los Breakpoints?.
(10) **Colocar** un **breakpoint** en la línea 42 (printf("\n Ingrese tamaño…..))del main. ¿Cómo visualiza si ha colocado un break?.
(11) **Colocar** otro **breakpoint** en la línea 53 (printf("\n Ingrese tamaño…..)).
(12) Reanudar el depurador *(<F8> o Ejecutar – Reanudar)*.
(13) ¿Dónde se detuvo? ¿Se encuentran todas las variables locales definidas en el main? Si alguna no la visualiza agregarla (En el panel *Variables*, botón derecho – *Crear expresión de observación*). Otra manera de visualizar una variable es seleccionarla en el código y con botón derecho seleccionar *Add Watch Expresion*.
(14) Continuar con la depuración – Oprima *<F6>* avanza línea por línea. O *<F8>* para avanzar al próximo Breakpoint.
(15) Ingresar los datos solicitados en la "consola de ejecución".
(16) Cuando se detenga en la línea 53 (breakpoint) visualice en la ventana examinador si las variables involucradas hasta ese punto se han modificado correctamente.
(17) Si todo está correcto continúe presionando *<F6>* línea por línea hasta la línea 62.

(18) Luego visualice en el panel "Variables" si las variables involucradas hasta ese punto se han modificado correctamente.
(19) ¿Cad2 tiene almacenado lo que acaba de ingresar? Mirar detenidamente el programa y deducir por qué.
(20) Detener el depurador. (Control + F2).
(21) Corregir su programa, guardar las modificaciones y construir el ejecutable nuevamente.
(22) Ejecutar. ¿Funcionó correctamente?
(23) Agregar un Breakpoint en la línea 62 *(printf("\nIngrese la posición).*
(24) Vuelva a comenzar la depuración y analice los valores de las variables cada vez que se detenga en un breakpoint.
(25) ¿Funcionó el ejecutable correctamente?
(26) Eliminar los breakpoints de las líneas 42 y 53. *Oprimir el botón rojo de dicha línea.*
(27) Comenzar nuevamente la depuración.

Si fuera necesario agregar en el examinador las variables que necesite. Continuar depurando y solucionar todos los problemas. Si desea seguir paso a paso la depuración en una función oprima *<F5>*.

***Nota**: Por cada programa debe crear en el directorio Practico5 un proyecto Prac5_ejxx*

Ej.7- Recuperar de **/home/serv/serv99/practico5/** el archivo **p5_depurar.c.** Compilarlo y ejecutarlo. Si no funciona correctamente corregirlo utilizando el depurador analizando en cada sentencia el valor de las variables.

Ej.8– Dado el siguiente código que se encuentra almacenado en la ubicación /home/serv/serv99/practico5:

```
# include <stdio.h>
int main (){
   float real = 1.0, *p_real;
   int entero = 2983, *p_entero;
   char caracter = 'a', *p_caracter;
   char letras [5]= {'a','e','i','o','u'}, *p_letra;
   int numeros [3] = {98,-76,34}, *p_num;

   printf("El tamaño de un real es %d y del puntero es %d \n",
                                    sizeof(real), sizeof(p_real));
   printf("El tamaño de un entero es %d y del puntero es %d\n",
                                    sizeof(entero), sizeof(p_entero));
   printf("El tamaño de un caracter es %d y del puntero es %d\n",
                                    sizeof(caracter), sizeof(p_caracter));
   printf("El tamaño del arreglo es %d y del puntero es %d\n",
                                    sizeof(letras), sizeof(p_letra));
   p_letra = letras;
   printf("El tamaño de una posición de arreglo es %d y del puntero es %d\n",
                                    sizeof(letras[0]), sizeof(p_letra));
   p_num = numeros;
   printf("El tamaño del arreglo es %d y su dirección es %d\n",
                                    sizeof(numeros), sizeof(p_num));
   p_num++;
   printf("p_num ahora apunta a %d y su dirección es %d\n", *p_num, p_num);

   getchar();
   return (0);
}
```

a) Copiar el archivo denominado **"p5_ej8.c"** a workspace y luego *compilarlo y ejecutarlo*.

b) Analizar gráficamente la salida del mismo. ¿Los tamaños de los arreglos son iguales? ¿Qué diferencia en bytes existe entre p_num y p_letra?

Ej.9– Dado el siguiente código que se encuentra almacenado en la ubicación /home/serv/serv99/practico5:

```
# include <stdio.h>
int main (){
      struct medidas
      {
            int edad;
            float altura;
            float peso;
      };
      struct medidas reg, *p_reg;
      printf("El tamaño de la estructura es %d  y  del puntero es %d \n",
                                          sizeof(reg), sizeof(p_reg));
      getchar();
      return(0);
}
```

a) **Copiar** el archivo **"p5_ej9.c"** a su workspace y luego ejecutarlo.
b) ¿Qué **diferencia** hay entre "struct medidas", "struct medidas reg" y "struct medidas *p_reg"?
c) ¿Cuál es la salida del programa? **Analizar** gráficamente.
d) Agregar a la estructura medidas un campo sexo de tipo char.
e) Ejecutar, se modificó algo en la salida?, es lo que Ud pensaba? Justifique?
f) Agregar al código anterior antes de getchar(); las siguientes sentencias.

```
(*p_reg).edad = 8;     //(3)
(*p_reg).altura = 1.20;
(*p_reg).peso = 24;


reg1 = reg;            //(1)
reg1.peso = 24.3;
reg.peso = reg1.peso;  //(2)


printf(" %d\n", reg1.edad);
printf(" %f\n", reg1.altura);
printf(" %f\n", reg1.peso);
printf(" %f\n", reg.peso);
```

g) Compilar, **agregar** lo que sea necesario y ejecutar paso a paso.
h) ¿Son correctas las sentencias señaladas con (1) y (2)? ¿Qué realiza cada una de ellas?
i) Analizar los distintos valores que se van asignando a las variables.
j) Coloque otra forma de escribir la sentencia señalada con (3) usando p_reg

Ej. 10.- Realizar un programa que defina una estructura con los siguientes campos: ***codigo*** (entero), ***tipo*** (carácter en el rango A-Z) y ***longitud*** (real). Posteriormente, usar dos variables del tipo de la estructura definida anteriormente, ingresar desde teclado la información para dichas variables, luego **intercambiar todos sus datos** y mostrar sus contenidos.

Nota: ¿Es posible **intercambiar únicamente el campo *código*** de ambas variables? Justificar y **codificar** una función para implementarlo.

Ej. 11.- Copiar desde serv99/practico5 el archivo **"p5_ej12.c"** y luego realizar lo siguiente:

- **Abrir** el programa **"p5_ej12.c"** y analizar su contenido. Ejecutarlo.

- **Colocar** en el código fuente los siguientes comentarios donde corresponda:

Definición Global del tipo empleado.

Empleado es un tipo estructura con dos campos uno Código y el otro Nombre.

Uno de los parámetros actuales es "todos", un arreglo tipo estructura empleado.

Declaro "todos" como un arreglo tipo estructura empleado.

Uno de los parámetros formales es puntero del tipo struct empleado.

Acceso al campo "codigo" mediante puntero.

Acceso al campo "nombre" mediante puntero.

Imprime los datos ingresados.

Incremento el puntero.

- **Guardar** los cambios.

- **Contestar** las siguientes preguntas:

a) ¿De qué tipo son los campos de "empleado"?
b) ¿Cuántas variables tipo "empleado" se han declarado en el programa?
c) ¿Puedo declarar más variables y/o arreglos de tipo "struct empleado"?
d) ¿Cuántos datos de empleados puedo almacenar?
e) ¿Por qué el primer parámetro formal de la función "ingreso" es un puntero?
f) En la función ingreso, ¿qué formato de información permite ingresar en el campo nombre?
g) Si scanf("%d",&((*a).codigo)); es lo mismo que escribir scanf("%d",&(a->codigo)), ¿cuál es el equivalente para el campo nombre? Probar.

Ej. 12- Realizar en el programa anterior las siguientes modificaciones:

a) Agregar a la estructura empleado dos nuevos campos: Sueldo (float) y el otro Estado Civil (char) (**C**asado, **S**oltero, **D**ivorciado, **V**iudo).

b) Incorporar sentencias en la función "ingreso" y el "main" para que se ingrese información en los dos campos agregados.
Realizar todos los controles necesarios.

c) Definir una función que reciba un estado civil (1 char) imprima por salida estándar el correspondiente estado ('C' = **C**asado, 'S' = **S**oltero, 'D' = **D**ivorciado, 'V' =**V**iudo).

d) Dado un código ingresado por el usuario, definir una función que busque el empleado correspondiente a dicho código, luego modifique el sueldo de dicho empleado.

e) Dado un código ingresado por el usuario, definir una función que imprima el nombre, el estado civil (palabra) y el sueldo de dicho empleado.

f) Repetir todas las veces que el usuario desee los puntos d y e.

Ej. 13- Modificar el Ej. 12 del Práctico 2, ahora cada patente deberá ser almacenada en una estructura formada por dos campos; un campo para la parte alfabética y el otro para la parte numérica.

Ej. 14- Modificar la estructura del ejercicio 9 agregando un campo denominado **nombre** para que junto con los datos de edad, altura y peso, permita llevar el control físico de un alumno. Realizar un **programa** que permita **ingresar en un arreglo** los datos de 20

alumnos que practican natación (utilizar la estructura anterior). Finalmente, el programa deberá mostrar con sus carteles correspondientes el **nombre** y **edad** de **el/los alumno/s de mayor altura** por otro lado **el/los de menor peso.**

Ej. 15- Realizar un programa que permita almacenar una agenda, ingresar hasta 50 contactos. Los datos que se almacenarán por contacto son: nombre y apellido (no más de 30 caracteres), teléfono fijo, teléfono móvil, día, mes y año de nacimiento. Luego se deberá separar los contactos almacenados en dos arreglos: el primero contendrá nombre, apellido y número de teléfono móvil de los contactos con característica de San Luis en ambos números telefónicos. Y el segundo arreglo toda la información del contacto más joven y del más grande. Finalmente imprimir la información de los tres arreglos con sus respectivos carteles.

Ejercicios Propuestos

Ej. 1 – Escriba un programa que ingrese dos números reales A y B. Luego, en una **función** deberá calcular A^B y $A^{1/B}$ sólo si A es positivo y B es natural. En caso contrario, la función debe retornar el error por el cual no se pudo realizar dicho cálculo (usando return). Los dos números ingresados deben ser almacenados en una variable tipo estructura local a la función *main,* e imprimir con carteles adecuados los valores antes y después de la invocación a la función en el main.

Ej 2.– Realizar un programa que **almacene en 3 arreglos** (llamados A, B y C) de 10 posiciones cada uno, datos cuyo código ascii varíe en el rango de **70 a 90.** Luego de ingresados los caracteres, el programa deberá **armar un arreglo de estructura** llamado D donde cada posición del mismo poseerá los caracteres de los arreglos A, B y C en esa **misma posición** y en ese **mismo orden.** Finalmente se deberá **mostrar** por pantalla el contenido de los arreglos A, B, C y el arreglo resultante con los carteles indicativos correspondientes.