

# Práctica 3

## Tipos de Datos Compuestos – Estructuras, Uniones, Enumerativos y Campos de Bits

### Estructuras

#### 1. Resuelva:

- Defina una estructura `rectangulo` que contenga los siguientes campos: `base (float)` y `altura (float)`.
- Escriba una función que reciba una estructura `rectangulo` y la inicialice a partir de valores ingresados por teclado.
- Escriba una función que dada una estructura `rectangulo`, calcule el área.
- Escriba un programa que defina un arreglo de 10 rectángulos (`struct rectangulo`) y lo inicialice utilizando la función definida en el inciso anterior. Luego, informe el número de rectángulo que tiene menor área.

#### 2. Dados los siguientes bloques de código:

```
struct persona {  
    char nombre[50];  
    long int DNI;  
} unaPersona;  
  
typedef struct persona persona_t;
```

```
typedef struct persona {  
    char nombre[50];  
    long int DNI;  
} persona_t;
```

¿En qué se diferencian ambos bloques? ¿Qué define cada uno?

#### 3. Resuelva:

- Defina una estructura `direccion` que contenga los siguientes campos: `calle` (arreglo de 50 caracteres), `ciudad` (arreglo de 30 caracteres), `codigo_postal` (int) y `pais` (arreglo de 40 caracteres).
- Defina una estructura `alu` que contenga los siguientes campos: `apellido` (arreglo de 50 caracteres), `nombre` (arreglo de 50 caracteres), `legajo` (arreglo de 8 caracteres), `promedio (float)` y `domicilio (struct direccion)`.
  - Renombre el tipo `struct alu` a `alumno` mediante la palabra clave `typedef`.
  - Escriba una función que reciba un alumno y lo inicialice a partir de valores ingresados por teclado.
  - Escriba un programa que defina un arreglo de 30 elementos `alumno` y lo inicialice utilizando la función definida en el inciso anterior. Luego, informe el nombre y apellido del alumno que tiene mejor promedio.
- Defina la estructura `pun3D`, la cual representa una posición en el espacio. La misma debe contener los campos `x (float)`, `y (float)` y `z (float)`. Luego:
  - Renombre la estructura `pun3D` a `punto3D` utilizando la palabra clave `typedef`.
  - Imprima en pantalla el tamaño del tipo `struct pun3D`. ¿Cuánto ocupa? ¿Por qué?
  - Imprima en pantalla el tamaño del tipo `punto3D`. ¿Cuánto ocupa? ¿Es igual al de `struct pun3D`? ¿Por qué?
  - Defina un arreglo de 4 elementos de tipo `punto3D` e imprima en pantalla el espacio ocupado por el mismo. ¿Cuánto ocupa? ¿Por qué?

4. Implemente una estructura y las funciones para implementar un mazo de 50 cartas españolas. Implemente las siguientes funciones y realice un programa para probarlas:
  - a. Barajar el mazo de cartas.
  - b. Sacar una carta: dado un mazo, sacar la carta del mazo y devolverla.
  - c. Imprimir una carta (número/figura con su palo).

*Nota: utilice constantes (define o const) para definir los palos de las cartas, modelice las cartas y el mazo.*

## Uniones

5. Muchas veces al trabajar en modo gráfico con ventanas se necesitan las dimensiones (posición x, posición y, ancho y alto) de diferentes formas para pasarlas como parámetros a distintas funciones. Defina una unión que comparta estas 3 formas de acceso para la definición de un rectángulo de una ventana: todas separadas (x, y, ancho, alto), como dos puntos ([x1, y1], [x2, y2] o [x, y], [ancho, alto]) o todas juntas como un rectángulo ([x, y, ancho, alto])
6. Implemente un tipo de datos Fecha para almacenar día, mes y año teniendo en cuenta las siguientes observaciones:
  - a. Utilice un formato que sea “cómodo” para trabajar con fechas
  - b. Como las comparaciones entre fechas son algo engorrosas, utilice una unión para realizar un “hack” que utilice un campo adicional que se superponga con la fecha y permita compararlas directamente (estudiar orden y tamaño de cada campo de la fecha).
  - c. Implemente un programa que compare distintas fechas para demostrar que esta estrategia funciona (puede aprovechar la declaración de las variables para asignar las fechas).
  - d. ¿Cree que esta implementación funcionaría para todos los compiladores de C sin importar la arquitectura del procesador? (Pista: Little endian vs. Big endian)
7. Desarrolle un programa que permita leer por teclado e imprimir por pantalla la información correspondiente a un estudiante: Apellido, Nombres, Legajo e Identificación. Tenga en cuenta que la identificación es el DNI (Numérico) para estudiantes argentinos y pasaporte (Alfanumérico) para estudiantes extranjeros.

## Enumerativos

8. Analice las siguientes declaraciones de enumerativos e indique los valores que adopta cada miembro y si las declaraciones son correctas (compilan) o no:

```
typedef enum { IZQUIERDA, CENTRO_H, DERECHA } AlineacionHorizontal;

typedef enum { ARRIBA=1, CENTRO_V, ABAJO } AlineacionVertical;

typedef enum { DOS=2, CERO=0, UNO, MENOS_UNO=-1, OTRO } Numeros;

typedef enum { LET_A = "A", LET_B, LET_Z = "Z" } Letras;

typedef enum { LETRA_A = 'A', LETRA_B, LETRA_Z = 'Z' } Letras2;
```

9. Resuelva:
  - a. Defina un enumerativo que permita representar los días de la semana (domingo a sábado).
  - b. Realice dos implementaciones diferentes de una función que imprima el texto asociado a un día de la semana del enumerativo.

*Nota: si bien una función para la impresión no es necesaria, favorece la reutilización, encapsulamiento, validación del rango del dato y minimiza la posibilidad de errores futuros*

10. Un teléfono móvil tiene varios bits de configuración que permite controlar el encendido y apagado de dispositivos con el objetivo, generalmente, de ahorrar energía. Cada bit se asocia a un módulo diferente donde un 1 significa encendido y un 0 apagado. Comenzando desde el bit más significativo, la descripción de estos bits es la siguiente: Bluetooth, Wifi, GPS, Datos, Frontal, Trasera, Linterna y Vibrar.

Bluetooth	Wifi	GPS	Datos	Frontal	Trasera	Linterna	Vibrar
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Resuelva:

- Utilice un tipo enumerativo para representar los distintos módulos que tiene el teléfono, asignándole a cada integrante un valor que sea potencia de 2. Esta estrategia permite combinar varios valores en una misma variable, y de esta forma modelar que un módulo está activo cuando el valor asociado está presente.
- Defina 2 constantes donde la primera contiene a todos los módulos y otra donde no contenga ninguno.
- Dada una variable del tipo definido en a), describa expresiones que permitan:
  - Marcar un módulo como activo
  - Marcar un módulo como inactivo
  - Invertir el estado actual de un módulo
  - Determinar si un módulo está activo
- Implemente un programa donde aplique todos los puntos anteriores.

## Campos de Bits

- Se requiere la implementación de un tipo de datos para almacenar el horario de manera compacta utilizando 16 bits. Debido a que el tamaño de los datos es más importante que la precisión, puede admitirse pérdida de precisión en los segundos. Desarrolle un programa que implemente este tipo de datos y funciones para leer la hora por teclado e imprimirla por pantalla.
- El formato de color RGB es el más extendido para uso en imágenes como en displays. Este formato utiliza 3 componentes (Red, Green, Blue) de 8 bits para codificar un píxel de color, requiriendo 3 bytes o 24 bits de memoria. Hace muchos años cuando surgieron los primeros teléfonos a color se utilizó un formato RGB reducido a 2 bytes o 16 bits (Red:5 bits, Green:6 bits, Blue: 5 bits) para reducir el espacio de almacenamiento y mejorar las velocidades de transferencia. Actualmente en el ámbito de los microcontroladores estos displays siguen siendo utilizados. Teniendo en cuenta lo anterior implemente un programa que:
  - Defina el tipo RGB24 y RGB16 de forma eficiente.
  - Implemente 2 funciones de conversión de un formato a otro y viceversa.
- Reimplemente todos los ítems del ejercicio 10 utilizando campos de bits para definir el estado de cada módulo del dispositivo.

## Integración de Conceptos

Utilice todos los conceptos vistos en la práctica para implementar los siguientes ejercicios

- Reimplemente el ejercicio 4 de las barajas españolas utilizando los conceptos vistos en la práctica para mejorar tanto el modelado del problema como la eficiencia.
- Desarrolle un programa que utilice un tipo de datos que modele las figuras geométricas bidimensionales: círculo, elipse, triángulo, cuadrado y rectángulo con sus propiedades (radio, lado/s, base, etc.). Implemente una única función que permita calcular la superficie de cualquier figura. El programa debe leer por teclado los datos de una figura e imprimir su superficie.

16. El siguiente programa maneja un dispositivo que recibe comandos a través del teclado y genera tonos musicales (sin implementar). Cada comando indica si el dispositivo debe permanecer encendido y la frecuencia del tono a reproducir. Encuentre los errores de compilación que presenta el programa:

```
#include <stdio.h>

// Frecuencia de tonos musicales
typedef enum tono {BASE=440, DO=262, RE=294, MI=330, FA=349, SOL=392, LA=440,
SI=494} tono_t;

// Estructura de comando
typedef struct comando {
    unsigned encendido : 1,
    unsigned frecuencia : 15
} comando_t;

// Encendido y apagado del dispositivo
#define NO 0
#define SI 1

/* Prototipos */
comando recibir_comando();

void generar_sonido(comando_t);

int main(){
    comando_t c;
    printf("Encendiendo el dispositivo...\r\n");
    c = recibir_comando();
    while(c.encendido == SI){
        // Es un valor de frecuencia válido
        if(c.frecuencia == DO || c.frecuencia == RE || c.frecuencia == MI ||
           c.frecuencia == FA || c.frecuencia == SOL || c.frecuencia == LA ||
           c.frecuencia == SI)
            generar_sonido(c); // Generar tono correspondiente
        else
            printf("ERROR: Valor de frecuencia incorrecto.\r\n");
        c = recibir_comando();
    }
    printf("Apagando el dispositivo...\r\n");
    return 0;
}

// Leer comando desde la entrada estándar con el formato "encendido.frecuencia"
comando recibir_comando(){
    comando_t c;
    printf("Ingrese comando: ");
    scanf("%d.%d", &c.encendido, &(c.frecuencia)); // Leer comando
    return c;
}

// Generar tono recibido por parámetro (sin implementar)
void generar_sonido(comando_t c){
    printf("Tono: %d Hz\r\n", c->frecuencia);
    // Generar sonido ...
    return;
}
```