

## Taller de Fundamentos de Programación – Uso de SWITCH

Duración: 20 minutos

Objetivo: Comprender el uso de la estructura de selección múltiple SWITCH en C utilizando condiciones anidadas, basado en el código proporcionado.

### 1. Introducción

En este taller el estudiante analizará un programa en C que identifica si un carácter ingresado es vocal, consonante, número o símbolo. Se utilizará la estructura SWITCH como herramienta principal para clasificar caracteres.

### 2. Código base de análisis

```
#include <stdio.h>

void main()
{
    char car;
    printf("Ingresa caracter: ");
    scanf("%c",&car);
    if (car>='A' && car<='Z')
    {
        switch(car)
        {
            case 'A': case 'E': case 'I':
            case 'O': case 'U': printf("Es vocal mayuscula\n");
            break;
            default: printf("Es consonante mayuscula\n");
            break;
        }
    }
    else
    {
        if (car>='a' && car<='z')
        {
            switch(car)
            {
                case 'a': case 'e': case 'i':
                case 'o': case 'u': printf("Es vocal minuscula\n");
                break;
                default: printf("Es consonante minuscula\n");
                break;
            }
        }
    }
    else
```

```

    {
        if (car>='0' && car<='9')
printf("Es un numero\n");      else
        printf("Es un simbolo\n");
    }
}
}

```

### 3. Actividades del Taller

**Actividad 1:** Explique en sus propias palabras cómo funciona el SWITCH dentro del programa.

el `switch` se utiliza aquí para **diferenciar rápidamente las Consonantes de las vocales** dentro de un conjunto de letras que ya ha sido confirmado como mayúscula (A-Z) o minúscula (a-z) mediante las sentencias `if`

**Actividad 2:** Modifique el programa para que también identifique vocales acentuadas (á, é, í, ó, ú).

Algoritmo sin\_titulo

//INICIO

// Configurar la localización para soportar caracteres acentuados

// Usamos un tipo para soportar caracteres anchos

// Solicitar al usuario que ingrese un carácter

ESCRIBIR "Ingrese un carácter:"

LEER car // Leer un carácter ancho

// Verificar si el carácter es mayúscula

//SI car ESTÁ ENTRE 'A' Y 'Z' ENTONCES

SEGÚN car HACER

CASO 'A':

CASO 'E':

CASO 'I':

CASO 'O':

CASO 'U':

CASO 'Á':

CASO 'É':

CASO 'Í':

CASO 'Ó':

```
CASO 'Ú':  
ESCRIBIR "Es vocal mayúscula"  
FIN SEGÚN
```

```
// Verificar si el carácter es minúscula  
//SINO SI car ESTÁ ENTRE 'a' Y 'z' ENTONCES  
    SEGÚN car HACER
```

```
CASO 'a':  
CASO 'e':  
CASO 'i':  
CASO 'o':  
CASO 'u':  
CASO 'á':  
CASO 'é':  
CASO 'í':  
CASO 'ó':  
CASO 'ú':  
ESCRIBIR "Es vocal minúscula"  
FIN SEGÚN
```

```
// Verificar si el carácter es un número  
//SINO SI car ESTÁ ENTRE '0' Y '9' ENTONCES  
    ESCRIBIR "Es un número"
```

```
// Si no es ninguna de las anteriores, es un símbolo  
  
    ESCRIBIR "Es un símbolo"
```

FinAlgoritmoActividad

### **CODIGO C**

```
#include <stdio.h>
```

```
void main()
```

```
{  
    char car;  
    printf("Ingrese caracter: ");  
    scanf("%c",&car);
```

```
// 1. Bloque de Letras Mayúsculas (A-Z y acentuadas)  
if ((car >= 'A' && car <= 'Z') || car == 'Á' || car == 'É' || car == 'Í' || car == 'Ó' || car == 'Ú')  
{  
    switch(car)
```

```

{
case 'A': case 'E': case 'I': case 'O': case 'U':
// Nuevos casos para vocales acentuadas mayúsculas
case 'Á': case 'É': case 'Í': case 'Ó': case 'Ú':
    printf("Es vocal mayuscula (incluye acentuada)\n");
    break;
default:
    printf("Es consonante mayuscula\n");
    break;
}
}
else
{
// 2. Bloque de Letras Minúsculas (a-z y acentuadas)
if ((car >= 'a' && car <= 'z') || car == 'á' || car == 'é' || car == 'í' || car == 'ó' || car == 'ú')
{
    switch(car)
    {
    case 'a': case 'e': case 'i': case 'o': case 'u':
// Nuevos casos para vocales acentuadas minúsculas
case 'á': case 'é': case 'í': case 'ó': case 'ú':
        printf("Es vocal minuscula (incluye acentuada)\n");
        break;
    default:
        printf("Es consonante minuscula\n");
        break;
    }
}
}
else
{
// 3. Bloque de Números (0-9)
if (car >= '0' && car <= '9')
{
    printf("Es un numero\n");
}
// 4. Bloque de Símbolos
else
{
    printf("Es un simbolo\n");
}
}
}
}
}

```

3: Cree una versión del programa donde el SWITCH identifique colores primarios: R, G, B.

Algoritmo

// Algoritmo para identificar colores primarios R, G, B

// INICIO

// Asumiendo que las configuraciones de caracteres anchos están al inicio del algoritmo

// Solicitud de entrada

ESCRIBIR "Ingrese un carácter (R, G, o B) para identificar un color primario:"

LEER car // Asume que 'car' es la variable que contiene el carácter de entrada

// --- CLASIFICACIÓN USANDO SEGÚN (SWITCH) ---

// La estructura SEGÚN evalúa la variable 'car'

SEGÚN car HACER

// CASO ROJO (Mayúscula y minúscula agrupados)

CASO 'R':

CASO 'r':

ESCRIBIR "El carácter corresponde al color PRIMARIO: Rojo."

ESCRIBIR "Asociado con la energía y la pasión."

// No se necesita "FIN SEGÚN" aquí si se está usando una estructura SEGÚN

// que tiene implícito el 'break' al encontrar un caso.

// Si el pseudocódigo requiere un "FIN SEGÚN" explícito, se usa al final del bloque.

// CASO VERDE (Mayúscula y minúscula agrupados)

CASO 'G':

CASO 'g':

ESCRIBIR "El carácter corresponde al color PRIMARIO: Verde."

ESCRIBIR "Asociado con la naturaleza y el crecimiento."

// CASO AZUL (Mayúscula y minúscula agrupados)

CASO 'B':

CASO 'b':

ESCRIBIR "El carácter corresponde al color PRIMARIO: Azul."

ESCRIBIR "Asociado con la calma y la estabilidad."

// CASO POR DEFECTO (Cualquier otra entrada)

DE OTRO MODO:

ESCRIBIR "El carácter ingresado NO es un color primario (R, G, o B)."

FIN SEGÚN

// FIN

FinAlgoritmo

## **CODIGO C**

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    char car;
```

```
    printf("Ingrese un caracter (R, G, o B) para identificar un color primario: ");
```

```
    // El espacio en " %c" es una buena práctica para ignorar espacios previos o saltos de línea.
```

```
    scanf(" %c", &car);
```

```
    // Utilizamos SWITCH directamente para clasificar la entrada, ya que solo nos interesan R, G, B
```

```
    switch(car)
```

```
{
```

```
    // Caso ROJO: Agrupa mayúscula y minúscula
```

```
    case 'R':
```

```
    case 'r':
```

```
        printf("Identificado: ROJO (R).\n");
```

```
        break;
```

```
    // Caso VERDE: Agrupa mayúscula y minúscula
```

```
    case 'G':
```

```

case 'g':

    printf("Identificado: VERDE (G).\n");

    break;


// Caso AZUL: Agrupa mayúscula y minúscula

case 'B':

case 'b':

    printf("Identificado: AZUL (B).\n");

    break;


// Caso por Defecto: Si no es R, G, o B.

default:

    printf("El caracter ingresado no es un color primario (R, G, B).\n");

    break;

}

}

```

#### 4. Preguntas de reflexión

1. ¿Por qué es útil el SWITCH en lugar de múltiples if-else?

El SWITCH es preferible a múltiples if-else porque ofrece **mayor claridad y orden** al comparar una sola variable contra muchos valores discretos. En lugar de evaluar secuencialmente cada condición, el SWITCH permite una **clasificación directa** (a menudo implementada internamente como una tabla de saltos, lo que potencialmente mejora el **rendimiento**)

2. ¿Qué sucede si no incluimos la instrucción break en cada caso?

Si omites el **break**, el SWITCH sufre "**fall-through**": el código **no se detiene** tras una coincidencia y **ejecuta consecutivamente** todos los casos siguientes, incluyendo el default, lo que causa un **error lógico** en el programa.



### 3. ¿Cómo se comporta el programa si el usuario ingresa más de un carácter?

El programa, al usar `scanf ("%c", &car)` ;, está estrictamente limitado a procesar **solo el primer carácter** que el usuario ingrese. Si el usuario teclea más de uno (ej., "CASA"), el programa solo clasifica la 'C'. Los caracteres restantes (ASA y el salto de línea) se quedan en el **búfer de entrada**, lo que puede causar problemas si hay más entradas planeadas posteriormente.

### 5. Mini ejercicio práctico

Escriba un programa en C utilizando SWITCH que permita ingresar un número del 1 al 7 y muestre el día de la semana correspondiente.

algoritmo dias

//INICIO

//DECLARAR dia COMO ENTERO

// Solicitar al usuario que ingrese un número del 1 al 7

ESCRIBIR "Ingrese un número del 1 al 7 para conocer el día de la semana:"

LEER dia

// Usar un switch para determinar el día de la semana

SEGÚN dia HACER

1:

ESCRIBIR "Lunes"

2:

ESCRIBIR "Martes"

3:

ESCRIBIR "Miércoles"

4:

ESCRIBIR "Jueves"

5:

ESCRIBIR "Viernes"

6:

ESCRIBIR "Sábado"

7:

ESCRIBIR "Domingo"

OTRO:

ESCRIBIR "Número inválido. Por favor, ingrese un número del 1 al 7."

FIN SEGÚN

FinAlgoritmo

## 6. RUBRICA

Criterio	Descripción del desempeño esperado	Puntaje
Comprensión del funcionamiento del SWITCH	Explica con claridad en sus propias palabras cómo opera la estructura SWITCH dentro del programa base,	0 – 5 pts
	incluyendo casos, break y flujo lógico.	
Modificación del programa para incluir vocales acentuadas	Integra correctamente los caracteres á, é, í, ó, ú en la lógica del programa, manteniendo coherencia, sintaxis y funcionamiento.	0 – 5 pts

Versión alternativa del programa (R, G, B)	Implementa un nuevo SWITCH funcional que identifique colores primarios (R, G, B). Se evalúa creatividad, sintaxis y correcto uso de casos.	0 – 4 pts
Respuesta a preguntas de reflexión	Responde con criterio propio y claridad conceptual a las tres preguntas de reflexión del taller, demostrando comprensión real del tema.	0 – 3 pts
Mini ejercicio: días de la semana	Programa en C un SWITCH que muestre el día de la semana según un número del 1 al 7. El código debe compilar y mostrar todos los casos de forma ordenada.	0 – 3 pts