



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* Ing. Karina García Morales

*Asignatura:* Fundamentos de la Programación

*Grupo:* 20

*No. de práctica(s):* 07

*Integrante(s):* Martinez Ordoñez Diego Tonatiah

*No. de lista o brigada:* 30

*Semestre:* 2023-1

*Fecha de entrega:* 20 / noviembre / 2022

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

## Práctica 07: Estructuras de selección

### - Objetivo:

El alumno elaborará programas en lenguaje C que incluyan las estructuras de selección if, if-else, switch y ternaria (o condicional) para la resolución de problemas básicos.

### - Desarrollo:

Las estructuras de control de flujo en un lenguaje especifican el orden en que se realiza el procesamiento de datos. Las estructuras de selección (o condicionales) permiten realizar una u otra acción con base en una expresión lógica. Las acciones posibles a realizar son mutuamente excluyentes, es decir, solo se puede ejecutar una a la vez dentro de toda la estructura. Lenguaje C posee 3 estructuras de selección: la estructura if-else, la estructura switch y la estructura condicional o ternaria.

### - Estructura de control selectiva if.

Tenemos la estructura de flujo if, la cual es considerada la más simple, se muestra a continuación:

```
if (expresión_lógica)
{
    // bloque de código a ejecutar
}
```

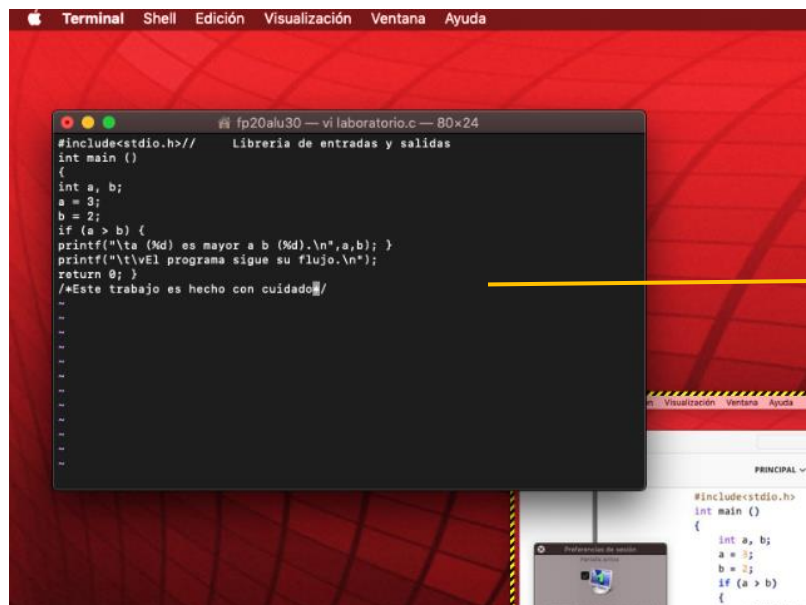
En esta estructura tiene que cumplir las condiciones que se dan, si estas condiciones son cumplidas enseguida se ejecutarán un conjunto de instrucciones correspondientes al bloque en el que se encuentre.

**NOTA 1:** Si el bloque de código a ejecutar consta de una sola línea de código no es necesario el uso de las llaves.

**NOTA 2:** Como ya se explicó en la práctica anterior, la expresión lógica evaluada regresará como resultado un número entero. Dentro de las estructuras de control, el 0 indica que la expresión lógica es falsa y cualquier número diferente de 0 indica que la expresión lógica es verdadera.

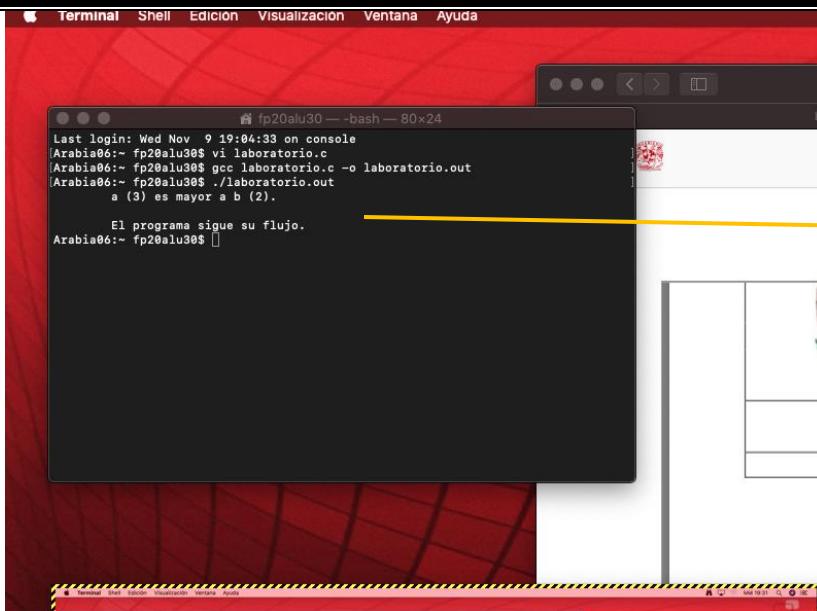
### - Códigos realizados en clase:

1. Tenemos el ejercicio 1, nos muestra cual es el mayor valor y cual es el menor.



```
#include<stdio.h>
int main ()
{
    int a, b;
    a = 3;
    b = 2;
    if (a > b) {
        printf("\ta (%d) es mayor a b (%d).\n",a,b);
        printf("\tEl programa sigue su flujo.\n");
    }
    return 0;
}
/*Este trabajo es hecho con cuidado*/
```

Aquí tenemos una condicional simple, en la que estamos declarando las variables y dándoles valores.



```
fp20alu30 ~ -bash — 80x24
Last login: Wed Nov  9 19:04:33 on console
Arabia06:~ fp20alu30$ vi laboratorio.c
Arabia06:~ fp20alu30$ gcc laboratorio.c -o laboratorio.out
Arabia06:~ fp20alu30$ ./laboratorio.out
a (3) es mayor a b (2).

El programa sigue su flujo.
Arabia06:~ fp20alu30$
```

En esta parte, estamos ejecutando el programa y de la misma manera ya nos muestra la impresión de este.

2. En el ejercicio 2, Este programa comprueba que las condiciones son numéricas  $0 \rightarrow$  falso,  $\neq 0 \rightarrow$  verdadero.



```
fp20alu30 ~ vi laboratorio2.c — 80x24
#include<stdio.h>
int main()
{
    if (0)
    {
        printf("porque la condición siempre es falsa (0).\n");
        printf("porque la condición siempre es falsa (0).\n");
    }
    if(30) // El bloque de código de esta estructura if
    // solo consta de una línea porque los comentarios
    // no son tomados en cuenta por el compilador.
    // La condición siempre es verdadera (diferente de 0)
    printf("Esta instrucción siempre se ejecuta.\n");
    return 0;
}
-- INSERT --
```

Aquí se muestra con los valores de la práctica.



```
fp20alu30 ~ vi laboratorio2.c — 80x24
#include<stdio.h>
int main()
{
    if (0)
    {
        printf("porque la condición siempre es falsa (0).\n");
        printf("porque la condición siempre es falsa (0).\n");
    }
    if (-30) // El bloque de código de esta estructura if
    // solo consta de una línea porque los comentarios
    // no son tomados en cuenta por el compilador.
    // La condición siempre es verdadera (diferente de 0)
    printf("Esta instrucción siempre se ejecuta.\n");
    return 0;
}
-- INSERT --
```

En esta parte, se muestra con nuestros valores.

### Estructura de control selectiva if-else.

En esta estructura podemos observar que primero se evalúa si es que esta bien la expresión y la condición, si la condición es falsa se ejecuta el bloque de código que está entre las llaves después de la palabra reservada else.

```
if (expresión_lógica)
{
    // bloque de código a ejecutar
    // si la condición es verdadera
}
else
{
    // bloque de código a ejecutar
    // si la condición es falsa
}
```

## - Códigos en clase:

3. Ejercicio 3: Este programa permite validar si un número es par o impar. El número se lee desde la entrada estándar (el teclado).

The image shows two terminal windows. The left window displays the C code for Ejercicio 3, which prompts the user to enter a number and checks if it is even or odd. A yellow arrow points from a text box to the code. The right window shows the execution of the program, where the user enters the number 2, and the program outputs 'El número 2 es par.' A yellow arrow points from a text box to this output.

```
#include <stdio.h> //Hola, Soy Tona y este es el ejercicio 3
int main() {
    int num;
    printf("Ingrese un número:\n"); scanf("%d",&num);
    printf("Ingrese un número:\n"); scanf("%d",&num);
    if ( num%2 == 0 )
        printf("El número %d es par.\n",num); else
        printf("El número %d es impar.\n",num); return 0;
}
```

Estoy realizando la codificación y con mi comentario arriba.

```
3 errors generated.
Arabia06:~ fp20alu30$ vi laboratorio2.c
Arabia06:~ fp20alu30$ gcc laboratorio2.c -o laboratorio2.out
Arabia06:~ fp20alu30$ ./laboratorio2.out
Esta instrucción siempre se ejecuta.
Arabia06:~ fp20alu30$ ./laboratorio2.out
Esta instrucción siempre se ejecuta.
Arabia06:~ fp20alu30$ vi laboratorio2.c
Arabia06:~ fp20alu30$ gcc laboratorio2.c -o laboratorio2.out
Arabia06:~ fp20alu30$ ./laboratorio2.out
Esta instrucción siempre se ejecuta.
Arabia06:~ fp20alu30$ vi ejercicio1.c
Arabia06:~ fp20alu30$ vi laboratorio3.c
Arabia06:~ fp20alu30$ gcc laboratorio3.c -o laboratorio3.out
Arabia06:~ fp20alu30$ ./laboratorio3.out
-bash: ./laboratorio3.out: No such file or directory
Arabia06:~ fp20alu30$ gcc laboratorio3.c -o laboratorio3.out
Arabia06:~ fp20alu30$ ./laboratorio3.out
Ingrese un número:
2
El número 2 es par.
Arabia06:~ fp20alu30$
```

Aquí podemos ver que ya nos realiza e indica si es par o impar el número.

4. Ejercicio 4: Este programa ordena en forma descendente tres valores enteros dados. Los valores se leen desde la entrada estándar (el teclado).

The image shows two terminal windows. The left window displays the C code for Ejercicio 4, which prompts the user to enter three numbers and sorts them in descending order. A yellow bracket points from a text box to the code. The right window shows the execution of the program, where the user enters the numbers 2, 6, and 8, and the program outputs the sorted values: '8 es mayor a 6 que es mayor a 2'. A yellow arrow points from a text box to this output.

```
#include <stdio.h> //Este es el ejercicio 4
int main() {
    int uno, dos, tres;
    printf ("Ingrese 3 números separados por espacios:\n"); scanf ("%d %d %d", &uno, &dos, &tres);
    if (uno > dos)
    {
        if (dos > tres) {
            printf("%d es mayor a %d que es mayor a %d\n", uno, dos, tres); }
        else
        {
            if (uno > tres)
            {
                printf("%d es mayor a %d que es mayor a %d\n", uno, tres, dos);
            }
            else
            {
                printf("%d es mayor a %d que es mayor a %d\n", tres, uno, dos);
            }
        }
    }
    else
    {
        printf("%d es mayor a %d que es mayor a %d\n", tres, uno, dos);
    }
}
```

La codificación se esta haciendo, con su comentario en la parte superior.

```
Arabia06:~ fp20alu30$ gcc laboratorio4.c -o laboratorio4.out
laboratorio4.c:38:11: error: expected ';'
return 0;
^
laboratorio4.c:2:12: note: to match this '{'
int main() {
^
1 error generated.
Arabia06:~ fp20alu30$ vi laboratorio4.c
Arabia06:~ fp20alu30$ gcc laboratorio4.c -o laboratorio4.out
laboratorio4.c:39:11: error: expected ';'
return 0;
^
laboratorio4.c:3:1: note: to match this '{'
{
^
1 error generated.
Arabia06:~ fp20alu30$ vi laboratorio4.c
Arabia06:~ fp20alu30$ gcc laboratorio4.c -o laboratorio4.out
Arabia06:~ fp20alu30$ ./laboratorio4.out
Ingrese 3 números separados por espacios:
2 6 8
8 es mayor a 6 que es mayor a 2
Arabia06:~ fp20alu30$
```

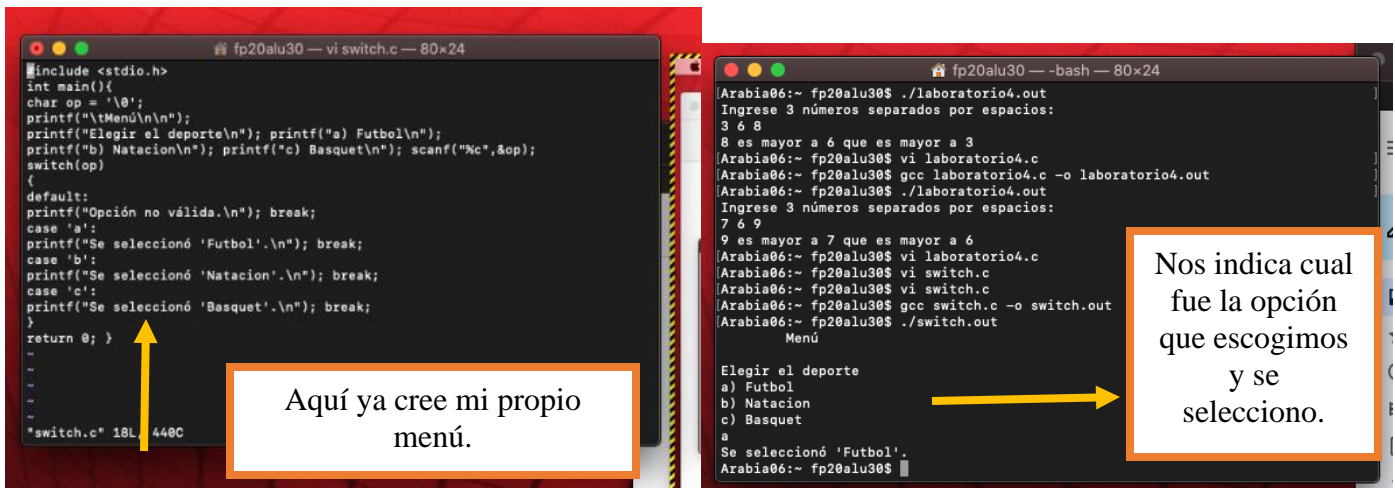
Aquí ya nos imprime los valores y cual es el mayor, el medio y el más chico.

## - Estructura de control selectiva switch-case.

En esta estructura, se evalúa la variable que se encuentra dentro del paréntesis, después del switch. Los tipos de datos que puede evaluar esta estructura son enteros, caracteres y enumeraciones, al final de cada caso se tiene que poner un break, para poder cerrar el caso y seguir con los siguientes.

## - Códigos de clase:

5. Ejercicio 5 y 6: Este programa permite elegir una opción del menú a partir del carácter ingresado. La opción se lee desde la entrada estándar (el teclado).



```
#include <stdio.h>
int main(){
    char op = '\0';
    printf("\tMenú\n\n");
    printf("Elegir el deporte\n"); printf("a) Futbol\n");
    printf("b) Natacion\n"); printf("c) Basquet\n"); scanf("%c",&op);
    switch(op)
    {
        default:
            printf("Opción no válida.\n"); break;
        case 'a':
            printf("Se seleccionó 'Futbol'.\n"); break;
        case 'b':
            printf("Se seleccionó 'Natacion'.\n"); break;
        case 'c':
            printf("Se seleccionó 'Basquet'.\n"); break;
    }
    return 0; }
"switch.c" 18L 440C
```

Aquí ya cree mi propio menú.

```
Arabia06:~ fp20alu30$ ./laboratorio4.out
Ingrese 3 números separados por espacios:
3 6 8
8 es mayor a 6 que es mayor a 3
Arabia06:~ fp20alu30$ vi laboratorio4.c
Arabia06:~ fp20alu30$ gcc laboratorio4.c -o laboratorio4.out
Ingrese 3 números separados por espacios:
7 6 9
9 es mayor a 7 que es mayor a 6
Arabia06:~ fp20alu30$ vi laboratorio4.c
Arabia06:~ fp20alu30$ vi switch.c
Arabia06:~ fp20alu30$ gcc switch.c -o switch.out
Arabia06:~ fp20alu30$ ./switch.out
Menú
Elegir el deporte
a) Futbol
b) Natacion
c) Basquet
a
Se seleccionó 'Futbol'.
Arabia06:~ fp20alu30$
```

Nos indica cual fue la opción que escogimos y se selecciono.

## - Enumeración.

En esta estructura podemos identificar la enumeración, variables que pueden ir una atrás de otra dandoles valores y para crear una enumeración se utiliza la palabra reservada enum, seguida de un identificador (nombre) y, entre llaves se ingresan los nombres de los valores que puede tomar dicha enumeración, separando los valores por coma.

## - Códigos de clase:

6. Ejercicio 7 y 8:



```
#include <stdio.h> // La verdad se me complica el poder ejecutar y acomodar
int main() {
    // Los valores de una enumeración son enteros y constantes
    enum diasSemana {LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO, DOMINGO}; int op;
    printf("Ingrese el día de la semana.\n");
    printf("1) Lunes\n");
    printf("2) Martes\n"); printf("3) Miércoles\n"); printf("4) Jueves\n"); printf("5) Viernes\n"); printf("6) Sábado\n"); printf("7) Domingo\n"); scanf("%d", &op);
    switch(op-1) {
        case LUNES: case MARTES:
            printf("Inicio de semana.\n");
            break; case MIERCOLES:
            printf("Mitad de semana.\n");
            break; case JUEVES:
            printf("Casi inicia el fin de semana.\n");
            break; case VIERNES:
            printf("Fin de semana.\n"); break;
        case SABADO:
            printf("Día de descanso.\n"); break;
        case DOMINGO:
            // No se necesita default
    }
}
-- INSERT --
```

Podemos ver la codificación que se esta realizando y el cómo se ponen los "break", para finalizar el caso. En la parte superior el comentario.



```
fp20alu30 — programa8.out — 80x24
1 error generated.
Arabia06:~ fp20alu30$ vi programa8.c
Arabia06:~ fp20alu30$ gcc programa8.c -o programa8.out
Arabia06:~ fp20alu30$ ./switch.out
Menú

Elegir el deporte
a) Futbol
b) Natacion
c) Basquet
a
Se seleccionó 'Futbol'.
Arabia06:~ fp20alu30$ gcc programa8.c -o programa8.out
Arabia06:~ fp20alu30$ ./programa8.out
Ingrese el día de la semana.
1) Lunes
2) Martes
3) Miércoles
4) Jueves
5) Viernes
6) Sábado
7) Domingo
```

En la impresión de pantalla nos muestra los días que podemos seleccionar.

### - Estructura de control selectiva condicional.

La estructura condicional (también llamado operador ternario) permite realizar una comparación rápida. Consta de tres partes, una condición y dos acciones a seguir con base en la expresión condicional. Si la condición se cumple (es verdadera) se ejecuta la instrucción que se encuentra después del símbolo '?'; si la condición no se cumple (es falsa) se ejecuta la instrucción que se encuentra después del símbolo ':'.  
- Códigos en clase:

```
fp20alu30 — vi programa9.c — 80x24
#include <stdio.h> //Mi comentario es que ya pude realizar casi todos
int main() {
double a, b, res;
printf("Calcular el error matemático E = |a - b|\n\n"); printf("Ingrese el valor de a:\n");
scanf("%lf",&a);
printf("Ingrese el valor de b:\n");
scanf("%lf",&b);
res = a < b ? b-a : a-b;
printf("El error matemático de\n");
printf("| %lf - %lf | es %lf\n", a, b, res);
return 0;
}
~
~
~
~
~
~
~
~
~
~
"programa9.c" 12L, 384C
```

Se muestra la decodificación y las variables, sus condiciones para que pueda funcionar adecuadamente.

```
fp20alu30 — -bash — 80x24
Ingrese el día de la semana.
1) Lunes
2) Martes
3) Miércoles
4) Jueves
5) Viernes
6) Sábado
7) Domingo
5
¡Fin de semana!
Arabia06:~ fp20alu30$ vi programa8.c
Arabia06:~ fp20alu30$ vi programa9.c
Arabia06:~ fp20alu30$ gcc programa9.c -o programa9.out
Arabia06:~ fp20alu30$ ./programa9.out
Calcular el error matemático E = |a - b|

Ingrese el valor de a:
5
Ingrese el valor de b:
2
El error matemático de
| 5.000000 - 2.000000 | es 3.000000
Arabia06:~ fp20alu30$ vi programa9.c
Arabia06:~ fp20alu30$
```

No muestra la operación, y su respectivo resultado.

### - Conclusiones:

Estamos aprendiendo mas a cerca de como podemos construir nuestros códigos fuente y la codificación, aprendiendo de los errores que son mas comunes y que los pequeños detalles nos pueden llegar a ponernos en problemas.

Tenemos que poner mucha atención en el momento que comenzamos a describir el problema y que a partir de esto podemos hacerlo bien o cometer un error.

Esta parte me parece bastante interesante porque de cierta forma podemos saber realizar diversos procesos para poder realizar codificaciones y obtener los resultados que son los deseados, de igual manera, al saber funciones nuevas de diversas funciones.

### - Referencias:

El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.