



**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

ALUMNO:

JONATHAN SAID GOMEZ MARBAN

PROFESOR:

JUAN JESÚS GUTIÉRREZ GARCÍA

**FUNDAMENTOS DE PROGRAMACIÓN**

PRÁCTICA 1 UNIDAD 2 (P21):

**“Función Swap”**

El propósito de esta práctica es que con una función se logre intercambiar el valor de cinco variables entre sí, de tal forma que, el valor inicial de cada una de las variables sea distinto al terminar dicha función. El ejemplo más sencillo y similar es recibiendo solamente dos parámetros, el parámetro tipo apuntador a carácter (char\*) "a" con el valor 'A' y el otro tipo char\* "b" con el valor 'B', al final el valor que tendrá "a" = 'B' y el valor de "b" = 'A'.

En un archivo llamado fun.h donde se crean los prototipos de todas las funciones que se quieran definir y posteriormente usar. Para poder resolver el problema en el cual solamente se reciben dos parámetros, se crea el prototipo de la función de la siguiente forma:

```
10 #ifndef FUN_H
11 #define FUN_H
12
13 void swap2 (char*,char*);
```

Imagen 1. Prototipo swap2.

Como se logra observar en la imagen 1, la función es llamada swap2 debido a que swap en inglés significa intercambio y el número 2 alude a que recibe 2 parámetros, de esta forma estamos dando un nombre descriptivo a nuestra función lo cual es una buena práctica; Esta función recibe dos parámetros de tipo apuntador a carácter esto con la finalidad de hacer paso de parámetros por referencia, con lo cual, se pasa un apuntador a la variable original y se puede modificar su valor dentro de la función. La función regresa void, dicho de otra forma, la función swap2 regresa nada, esto debido a que nosotros requerimos que la función haga los cambios de valores entre las variables correspondientes, y no queremos que solamente regrese valor en específico como lo haríamos usando la palabra reservada return.

Para el problema principal (intercambiar el valor de cinco variables entre sí), el prototipo de la función sería así:

```
14 void swap5 (char*,char*,char*,char*,char*);
```

Imagen 2. Prototipo swap5.

Al igual que como se explicó con el prototipo de la función swap2, esta función swap5 tiene un nombre descriptivo, recibe como parámetros 5 apuntadores a carácter (char\*), para poder hacer paso de parámetros por referencia y el void nos indica que la función no regresa nada.

Como se mencionó con anterioridad todos los prototipos de las funciones se colocan en el archivo fun.h; la extensión de nombre de archivo .h representa el tipo de archivo del archivo de cabecera (.h), además son parte de un proyecto de programación. Al incluir un archivo .h a través de la declaración "#include", el contenido del archivo .h (prototipos) se puede combinar con el código principal cuando sea necesario. Y de esta forma es que en un archivo nuevo llamado fun.c incluimos al archivo fun.h, de esta forma:

```
1 #include "fun.h"
2
```

Imagen 3. Inclusión del archivo fun.h.

En nuestro archivo llamado fun.c es en donde haremos las definiciones de las funciones de las cuales ya tenemos el prototipo que están contenidas en el archivo fun.h, de la siguiente manera:

```
3 void swap2 (char* x, char* y) {
4     char aux;
5     aux = *x;
6     *x = *y;
7     *y = aux;
8     return;
9 }
10
11 /*Este es el ejercicio que me fue asignado para realizar
12    el día de la práctica, es el número 41.*/
13
14 void swap5 (char* a, char* b, char* c, char* d, char* e) {
15     char aux;
16     aux = *a;
17     *a = *b;
18     *b = *d;
19     *d = aux;
20     aux = *c;
21     *c = *e;
22     *e = aux;
23     return;
24 }
```

Imagen 4. Definiciones de las funciones en el archivo fun.c.

Como se nota en la imagen 4, la función swap2 hace el cambio de valor con la ayuda de una variable de tipo char nombrada aux que se le asigna el valor del primer parámetro para que este no se pierda al asignarle un nuevo valor que será el del segundo parámetro y

finalmente al segundo parámetro se le asigna el valor de aux el cual contiene el valor del primer parámetro, de esta forma se hace el cambio de valores entre variables.

Lo explicado con anterioridad se puede explicar con lo siguiente:

```
aux ← a  
a ← b  
b ← aux
```

De la misma forma se explica que es lo que hace la función swap5 que se presenta en la imagen 4:

```
aux ← a  
a ← b  
b ← d  
d ← aux  
aux ← c  
c ← e  
e ← aux
```

Este proceso es el que sigue la función swap5, lo que nos da la siguiente combinación:

No	A	B	C	D	E
41	B	D	E	A	C


Finalmente, hace falta invocar la función swap5, eso se hará en un nuevo archivo llamado `usa_fun.c` en el cual estarán incluidos el archivo de cabecera `fun.h` y la biblioteca de entrada y salida estándar. Y dentro de la función `main` se declaran 5 variables nombradas `a`, `b`, `c`, `d` y `e` y se le asignan los valores de 'A', 'B', 'C', 'D' y 'E', respectivamente. Usando `printf` y se le dan 5 especificaciones de formato para carácter, la cual es `%c` y un salto de línea; se imprimirán en pantalla todas las variables con sus valores iniciales. Posteriormente se invoca a la función `swap5` y se hace un paso de parámetros por referencia, luego se usa otro `printf` exactamente igual que el primero solo que esta vez las variables ya contienen los valores que les fueron asignados al pasar por la función `swap5`.

Toda la explicación anterior se detalla en la siguiente imagen:

```
1 #include "fun.h"
2 #include <stdio.h>
3
4 int main(){
5     char a,b,c,d,e;
6     a = 'A'; b = 'B'; c = 'C'; d = 'D'; e = 'E';
7     printf("%c %c %c %c %c\n",a,b,c,d,e);
8     /*Coloca de aquí el código que te permita obtener la
9     configuración que te corresponde*/
10    swap5(&a,&b,&c,&d,&e);
11    /*hasta aquí*/
12    printf("%c %c %c %c %c\n",a,b,c,d,e);
13    return 0;
14 }
```

Imagen 5. Archivo usa\_fun.c.

Con esto terminamos de hacer el código fuente del problema inicial. Ahora se procede a compilarlo sin que genere warnings, y cumpliendo con el propósito del cambio de valores entre variables:

 Símbolo del sistema

```
C:\Users\jona-\Downloads\swap\swap1>gcc usa_fun.c fun.c

C:\Users\jona-\Downloads\swap\swap1>a
A B C D E
B D E A C
```

Imagen 6. Compilar y ejecutar.

Como se logra observar en la imagen 6, usando el comando gcc y agregando los archivos usa\_fun.c y fun.c compilaremos. Se contempla que no genera warnings y cumple con el propósito para el cual fue creado, ya que, regresa la combinación requerida con anterioridad:

No	A	B	C	D	E
41	B	D	E	A	C

Con esto se da por terminada la práctica y el reporte de la misma.