



## TRABAJO PRÁCTICO Nº3

Realizá cada ejercicio en lenguaje C siguiendo las indicaciones de la cátedra. En los casos en los que se requiera, desarrollá un planteo de solución (PS) y una prueba de escritorio (PE) antes de compilar y correr el programa. Los primeros ejercicios proponen una definición de las funciones requeridas y la mayoría implica modificar lo que realizaste en los prácticos anteriores, por lo que si no los hiciste... ¡este es el momento!

**Ejercicio 1.** Modificá la resolución del ejercicio 1 del TP Nº 1 (E1TP1) para implementar una función que reciba un arreglo de caracteres como parámetro (*nombre*) e imprima por pantalla el mensaje "¡Hola *nombre*!".

```
void imprimir(char[] nombre);
```

**Ejercicio 2.** Modificá el programa del ejercicio anterior para que use una función que reciba el ingreso por teclado del usuario y devuelva un arreglo de caracteres que represente el *nombre*.

```
char[] ingresarNombre(); o ingresarNombre(char *nombre);
```

**Ejercicio 3.** Modificá el programa del ejercicio E4TP1 para implementar el cálculo de la edad exacta mediante una función.

```
int calcularEdad(char[] fechaNacimiento);
```

**Ejercicio 4.** Modificá el ejercicio E7TP1 para utilizar una función que determine si un número dado es primo (devuelve 1) o no (devuelve 0) (PS) (PE  $N=10$ ).

```
int esPrimo(int numero);
```

**Ejercicio 5.** Modificá el programa del ejercicio E8TP1 para implementar dos funciones, una para calcular el producto de dos números como sumas sucesivas y otra para calcular la potencia entre dos números  $N$  y  $M$  como productos sucesivos. (PE  $N=5$ ,  $M=3$ )

```
int producto(int factor1, int factor2);
```

```
int potencia(int base, int exponente);
```

**Ejercicio 6.** Modificá el programa del ejercicio E12TP1 implementando una función que calcule el  $n$ -ésimo término de la serie de Fibonacci para un parámetro  $n$  dado. Hacé que tú programa llene un arreglo de 10 posiciones con términos de la serie mencionada generando el parámetro  $n$  aleatoriamente entre 1 y 25.

**Ejercicio 7.** Modificá el programa del ejercicio E13TP1 para que llene un arreglo de 10 números de hasta 5 cifras generados aleatoriamente, pero que sólo inserte números capicúa. Implementá para eso una función que devuelva si un número es capicúa.

**Ejercicio 8.** Modificá el programa del ejercicio E15TP1 para que cada operador aritmético esté implementado en una función específica.

**Ejercicio 9.** Modificá el programa del ejercicio E1TP2 para que implemente una función que reciba por parámetro un arreglo de números enteros y un número entero y determine si el número dado está o no en el arreglo.

**Ejercicio 10.** Modificá el programa del ejercicio E3TP2 para implementar una función que reciba por parámetro dos arreglos de números enteros de dimensión 5 y devuelvan otro arreglo del mismo tamaño.



**Ejercicio 11.** Modificá el programa del ejercicio E6TP2 para que utilice dos funciones. Ambas deben recibir un arreglo de cadenas de caracteres que representen fechas y deben devolver la menor fecha del arreglo en un caso y la mayor en el otro.

**Ejercicio 12.** Modificá el programa del ejercicio E7TP2 para que reciba como parámetro un arreglo de cadenas de caracteres y una cadena de caracteres adicional y devuelva el arreglo del primer parámetro con la cadena del segundo insertada en orden alfabético.

**Ejercicio 13.** Revise el programa del ejercicio E10TP2 y proponga al menos tres funciones que podría extraer del código. Determine la declaración de las funciones indicando tipo de retorno, nombre y parámetros de entrada con sus tipos. Si lo desea implemente las funciones.

**Ejercicio 14.** El siguiente programa descompone un número entero como producto de números primos. Por favor indicá cómo lo modificarías para utilizar subrutinas respetando buenas prácticas como el principio de máxima localidad.

```
#include <stdio.h>

int numero;

int main() {
    printf("Ingrese un número entero positivo: ");
    scanf("%d", &numero);

    if (numero <= 1) {
        printf("El número debe ser mayor que 1.\n");
        return 1;
    }

    printf("Los factores primos de %d son: ", n);
    // Iteramos desde 2 (el primer número primo) hasta n
    for (int i = 2; i <= n; i++) {
        while (n % i == 0) {
            // Si n es divisible por i, i es un factor primo
            printf("%d", i);
            n = n / i;
        }

        // Si todavía quedan más factores primos, imprimimos un
        asterisco

        if (n > 1) {
```



---

```
        printf(" * ");  
    }  
}  
  
}  
  
printf("\n");  
  
return 0;  
  
}
```

**Ejercicio 15.** Escribí un programa con subrutinas que lea una frase ingresada por el usuario (máximo 100 caracteres) y luego permita ingresar un número de palabra (según su ubicación en la frase comenzando desde 1) y una nueva palabra que la reemplace. El programa modificará la frase original, buscando la palabra en la posición indicada y reemplazándola por la nueva, imprimiendo luego la frase modificada. El programa termina cuando el usuario ingresa la posición 0. Utilice pasaje de variables por referencia.