# PRÁCTICAS – TEMA 0

## Orden

Los ejercicios del tema 0 se desarrollarán en C:

Para que los programas queden ordenados lo ideal es que los llaméis Ejercicio1.c, Ejercicio2.c, etc.

Añadid estos ejercicios en una carpeta dentro de la máquina virtual.

## Ejercicios

1. Hacer un programa al cual introduzca un número y me diga si es mayor que 5 o menor
2. Crea un programa que pida dos números al usuario y muestre sus posiciones de memoria.

Para imprimir las posiciones de memoria hay que usar %p.

Muestra posteriormente cuanto tamaño ocupa en memoria un entero usando la función *sizeof(variable/tipovariable)* que devuelve el número de bytes que ocupa una variable en memoria.

Teniendo en cuenta que cada posición de memoria abarca un byte, intenta explicar si las variables están seguidas en memoria o no.

1. Hacer un programa al cual introduzca tres números por teclado y me los devuelva ordenados
2. Hacer un programa que me solicite dos números y me ofrezca un menú, en el cual las opciones sean las siguientes:
   1. Sumar
   2. Restar
   3. Multiplicar
   4. Dividir
   5. Salir

En función de la opción seleccionada se obtendrá el resultado conveniente.

Tened en cuenta que el buffer almacena el enter que pulsamos tras introducir un valor en *scanf*, por tanto, si solicitamos un carácter después de haber pedido otro valor, se rellenará solo son ese enter.

Podéis utilizar la función *getchar()*; o introducir un espacio donde corresponda en el *scanf* para que se vuelque ese carácter del buffer *scanf(" %c", &opcion);.*

1. Modifica el programa anterior de manera que hasta que el usuario no seleccione la opción de salir se repita el comportamiento.
2. Modulariza el programa anterior, de manera que cada opción sea una función.
3. Escribir un programa que me pida un número del 1 al 20, si el usuario introduce mal este valor debe volver a solicitarlo.

El programa ha de mostrar la tabla de multiplicar del número.

1. Escribe un programa que pida al usuario 10 caracteres.

Una vez introducidos, el programa ha de comprobar si el usuario ha introducido el carácter “k” y en el caso de que sea así ha de indicarlo de manera eficiente.

Es decir, si el carácter k es el segundo introducido por el usuario, el programa no necesitará analizar los números introducidos por el usuario posteriormente.

Para usar booleanos es necesario incluir una biblioteca: *#include<stdbool.h>*

1. Crea un programa que pida al usuario una palabra y la invierta .

Se puede usar la función *strlen(cadenaDeCaracteres)* para saber el número de caracteres que tiene la cadena a analizar.

Se encuentra dentro de la biblioteca *<string.h>*.

1. Crea un programa que pida al usuario dos dimensiones (cuyo valor máximo ha de ser 10 y el mínimo 25 ).

El programa ha de generar dos arrays con números aleatorios (entre 0 y 9) de las dimensiones indicadas por el usuario y mostrarlos ordenadamente por pantalla.

El programa multiplicará estos arrays y mostrará el resultado.

Como no tenemos conocimiento en punteros, no es necesario modularizar el programa.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

[Rand siempre devuelve el mismo valor](https://foro.elhacker.net/programacion_cc/iquestpor_que_el_numero_que_me_genera_el_rand_siempre_es_el_mismo-t481711.0.html)