PRÁCTICA FUNDAMENTOS DE PROGRAMACIÓN
HACER UN MENÚ DE OPCIONES

La presente práctica consiste en integrar todas las funciones realizadas en al práctica anterior en un único programa. En dicha práctica (la anterior) se plantearon 7 problemas que había que resolver haciendo 7 programas independientes. Ahora se pide hacer un único programa con un menú de opciones, cada una de las cuales deberá ejecutar una función que realice cada una de las tareas de la práctica anterior.

El menú de opciones deberá tener una forma similar a esta:

```
PRACTICA 2:
Alumno: Apellidos, Nombre

Menu:
1.- Contar vocales
2.- Validar un entero
3.- Validar un numero real
4.- Identificar un palindromo
5.- Calculos sobre un vector de enteros
6.- Quitar subcadena
7.- Validar fecha
8.- Salir

Elegir opcion: _
```

A diferencia de la práctica anterior, cada una de estas opciones se implementará en una función que recibirá uno o varios parámetros de entrada con los que tendrá que operar y hacer la tarea especificada. Los prototipos de dichas funciones deberán ser como se especifica a continuación:

Opción 1:
```
int ContarVocales(const char *cad);
```

Recibe como parámetro de entrada una cadena de caracteres y devuelve un valor entero con el número de vocales (mayúsculas y minúsculas) encontradas en la cadena.

Opción 2:
```
int EsEntrero(const char *cad);
```

Recibe una cadena y devuelve verdadero o falso (1 o 0) en función de si esa cadena constituye un número entero válido o no.

Opción 3:
```
int EsReal(const char *cad);
```

Ídem a la anterior pero verificando si la cadena es un número real o no.

Opción 4:
```
int EsPalindromo(const char *cad);
```

Ídem a la anterior pero verificando si la cadena es un palíndromo o no.

Opción 5:
```
void Calculos(const int *vector, int longitud);
```

Recibe un vector de enteros y la longitud de dicho vector e imprime en pantalla el resultado de los cálculos realizados (ver enunciado de la práctica anterior). Esta función no devuelve ningún valor y es la única que debe imprimir algo en pantalla (las demás no imprimen nada en pantalla pero devuelven un resultado).

Opción 6:
```
int QuitarSubcadena(char *cad1, const char *cad2);
```

Recibe dos cadenas, 'cad1' y 'cad2' y quita de 'cad1' todas las apariciones que haya en ella de la cadena 'cad2' (notar que 'cad2' está declarada como 'const' y 'cad1' no). La función devuelve el número de veces que se ha quitado 'cad2' de 'cad1'.

Opción 7:
```
int ValidarFecha(int d, int m, int a);
```

Recibe tres datos enteros, 'd' (día), 'm' (mes) y 'a' (año) y verifica si constituyen o no una fecha correcta. Devuelve verdadero (1) si la fecha es correcta o falso (0) si es incorrecta.

Opción 8:
Aquí no se llama a ninguna función ni se ejecuta ninguna tarea específica, simplemente el programa termina.

Tras la elección de cualquiera de las opciones el usuario deberá introducir por teclado los datos necesarios para posteriormente llamar a la función correspondiente. En el caso de las opciones 1, 2, 3 y 4 el usuario simplemente deberá introducir una cadena y llamar a la función correspondiente para que realice su tarea.

La opción 5 requiere que el usuario indique primero cuantos números enteros va a introducir (longitud del vector) y a continuación deberá introducir tantos valores como haya indicado, verificando que efectivamente son enteros válidos, en caso de que alguno no lo sea, el programa deberá avisarlo con un mensaje y pedirá que se vuelva a introducir (en la opción 2 tenéis una función que puede hacer esa verificación). Para acotar el número de enteros, se exigirá que el usuario indique como número de valores a introducir en el vector 2 como mínimo y 10 como máximo; para cualquier otro valor o cantidad el programa avisará diciendo que no es un valor válido y volverá a pedir que se introduzca.

La opción 6 requiere que el usuario introduzca dos cadenas. Aquí no hay que verificar nada, simplemente llamar a la función correspondiente y esperar la respuesta.

La opción 7 pedirá al usuario que introduzca en una cadena con el formato de fecha "*DD/MM/AAAA*" (2 dígitos para el día, 2 para el mes y 4 para el año, separados por el carácter '/'). Introducida la cadena el programa deberá verificar que tiene el formato indicado y si dicho formato es correcto se extraen de la cadena los valores 'd', 'm' y 'a' y se llama la función correspondiente para validarlos. Notar que en este caso la entrada pasa por dos validaciones, la primera es para ver si tiene el formato "*DD/MM/AAAA*" y la segunda es para ver si los valores 'd', 'm' y 'a' son coherentes; esta segunda validación la hace la función "*ValidarFecha(...)*", la primera habrá que programarla fuera de dicha función.

Con carácter general, durante la entrada de datos por parte del usuario previa a la llamada a cualquiera de las funciones que se han especificado, si el usuario introduce en algún momento la cadena "*" (un asterisco) el programa interrumpirá la entrada de datos y, sin llamar a ninguna de las funciones, volverá al menú principal, independientemente de en que opción se encuentre y cual sea el dato que debía introducirse.

Tras la llamada a cada opción (salvo la 5) el programa mostrará en pantalla un mensaje indicando el resultado de la operación, dicho mensaje deberá ser claro y conciso (algo como: "Se han contado X vocales" o "El número introducido No es un real valido" o "la cadena SÍ es un palíndromo", etc.). Además, para todas las opciones, al mostrar el resultado por pantalla, se deberá detener la ejecución del programa y esperar que el usuario pulse [INTRO] antes de volver a mostrar el menú principal (excepto para la opción 8, "*Salir*").

A la hora de evaluar la práctica:

- Se valorará positivamente:
  - La claridad con la que se describa TODO el desarrollo y contenido de la práctica (código bien tabulado, etc.).
  - La robustez del programa: Que no se cuelgue ni realice cosas "extrañas" durante su funcionamiento aunque el usuario introduzca por teclado datos erróneos o que no sigan el formato que se esperaba (leer sugerencia al final).

- Se valorará negativamente:
  - Mensajes de *warning* no resueltos al compilar.
  - **Funciones no implementadas**. Sobre este particular caber hacer una puntualización. Cuando se ejecuta un código (una función *main* por ejemplo) y se llama a una función no implementada obviamente, el programa no compilará o se colgará o producirá algún error de funcionamiento. Eso es lo peor que podéis hacer. Si el último día antes de entregar la práctica os queda una función por hacer (o más de una) y no os va a dar tiempo ha implementarla correctamente os recomiendo que la implementéis así.

```
void FuncionSinImplemtear()
{
      printf("Función 'FuncionSinImplemtear' sin implentar\n");
}
```

Aún más, si dicha función devolviera algún valor, ponedle un "**return**" para que devuelva algo acorde con el tipo del propio método (por ejemplo, si es '**int**' haced

"**return 0;**"). De esta forma evitaréis que el programa se cuelgue o de *warnings* y en cualquier caso se ejecutará hasta el final. Haciendo esto, la penalización por no terminar de implementar alguna función será algo menor.

- Se suspenderá directamente con un cero en estos casos:
  - Prácticas con errores que no se lleguen a ejecutar.
  - Practicas copiadas (ambas).

NOTA:

El programa final deberá contener al menos ocho funciones, las siete que se han descrito en esta práctica más la función '*main*'. Se deja a elección del alumno el implementar alguna función más que considere que puede servirle para realizar mejor su práctica. Recordad la teoría vista en el tema 5 sobre programación modular y sobre descomposición de un problema grande en varios problemas más pequeños y más fáciles de abordar y resolver.

SUGERENCIA:

Para la entrada de datos por teclado, si se quiere hacer un programa robusto, la función "*scanf(...)*" se desaconseja por completo ya que cuando se introducen espacios en blanco, más o menos parámetros de los esperados o parámetros que no son del tipo esperado esta función tiene un comportamiento impredecible. En su lugar se puede utilizar la función "*gets(...)*" en combinación con otras que validen el dato introducido.