TALLER DE PROGRAMACIÓN

Escribir un programa en C que permita jugar al ahorcado, de la siguiente forma:

Se dispone de 4 ficheros de texto, con palabras de diferentes temáticas: Animales.txt, Profesiones.txt, Nombres.txt, Paises.txt, en cada línea de estos ficheros hay una única palabra cuyo tamaño máximo es de 25 caracteres, en ninguno de los ficheros hay más de 50 líneas.

Primero, el usuario determinará con qué tipo de palabras quiere jugar (animales, profesiones, nombres o países) y el programa abrirá el fichero correspondiente.

A continuación, el programa, de forma reiterada, generará un número aleatorio que determinará una palabra del fichero, ésta será la palabra que el usuario deberá acertar.

Se rellenará una cadena con tantos caracteres '-' como longitud tenga la palabra a adivinar.

Después el programa pedirá que se introduzca un carácter y comprobará si pertenece a la palabra que se debe adivinar. Si es así, se escribirá dicho carácter en la cadena anterior en las mismas posiciones en las que se encuentra en la palabra a adivinar sustituyendo a los caracteres '-'.

Previamente a la lectura de cualquier carácter el programa debe mostrar la siguiente información:

Número de fallos: xx

Caracteres introducidos: x/x/x/x…/x

Representación gráfica de la situación (como las que se anexan o similares)

Progreso de la cadena en la que se busca la palabra: x - - x

Solicitud al usuario para Resolver o Introducir un nuevo carácter: *x*

Este proceso se repetirá hasta que:

- se acierte la palabra
- se agoten los intentos, es decir, se haya completado la figura del ahorcado
- el usuario decida resolver la palabra

Si se agotan los intentos o el usuario decide resolver sin haber acertado la palabra, se escribirá un mensaje indicándolo y se mostrará la palabra que se debía haber acertado, en otro caso se escribirá un mensaje de felicitación.

Para realizar el programa, al menos se deben codificar y emplear las siguientes funciones:

- Función **cabecera** que imprime NOMBRE y NUM. MATRICULA del autor del programa y que debe ser ejecutada nada más empezar el programa.
- Función **borraConsola** que limpia la consola realizando tantos saltos de línea como sea necesario.
- Función **longCad** que a partir de una cadena de caracteres devuelve la longitud de dicha cadena, es decir, el número de caracteres que contiene la palabra sin contar el carácter de final de cadena.
- Función **cadenasIguales** que dadas dos cadenas devuelve un 1 si son iguales y un 0 en caso contrario.

- Función **iniciaCad** que partiendo de un número que indica la longitud de una palabra, devuelve una cadena rellena con el carácter '-'tantas veces como sea la longitud dada.
- Función **actualizaCad** que a partir de la palabra a adivinar, un carácter y la cadena que se quiere completar, devolverá dicha cadena con el carácter en las posiciones donde se encuentra en la palabra a adivinar.
- Función **actualizaLista** que a partir de la lista de caracteres introducidos por el usuario añade una nueva entrada.
- Función **imprimeFigura** que a partir del número de fallos cometidos por el usuario imprime la figura correspondiente (ver ANEXO).

Especificación de diseño:

- La única librería que se puede utilizar es stdio.h por tanto no se admitirá el uso de la librería string.h.
- El fichero de palabras seleccionado solo se puede recorrer una vez.

ANEXO para la *Representación gráfica de la situación*

```
0 fallos:
        printf("\t.......\n");
        printf("\t.      .\n");
        printf("\t.      .\n");
        printf("\t.      .\n");
        printf("\t.      .\n");
        printf("\t.      .\n");
        printf("\t.      .\n");
        printf("\t.......\n");

1 fallo:
        printf("\n");
        printf("\n");
        printf("\n");
        printf("\n");
        printf("\n");
        printf("\t------\n");

2 fallos:
        printf("\n");
        printf("\t|      \n");
        printf("\t|      \n");
        printf("\t|      \n");
        printf("\t|      \n");
        printf("\t------\n");

3 fallos:
        printf("\t_____\n");
        printf("\t|/     \n");
        printf("\t|      \n");
        printf("\t|      \n");
```

```
        printf("\t|     \n");
        printf("\t------\n");

4 fallos:
        printf("\t_____\n");
        printf("\t|/ O  \n");
        printf("\t|     \n");
        printf("\t|     \n");
        printf("\t|     \n");
        printf("\t------\n");

5 fallos:
        printf("\t_____\n");
        printf("\t|/ O  \n");
        printf("\t|  |  \n");
        printf("\t|  |  \n");
        printf("\t|     \n");
        printf("\t------\n");

6 fallos:
        printf("\t_____\n");
        printf("\t|/ O  \n");
        printf("\t| /|  \n");
        printf("\t|  |  \n");
        printf("\t|     \n");
        printf("\t------\n");

7 fallos:
        printf("\t_____\n");
        printf("\t|/ O  \n");
        printf("\t| /|\\\n");
        printf("\t|  |  \n");
        printf("\t|     \n");
        printf("\t------\n");

8 fallos:
        printf("\t_____\n");
        printf("\t|/ O  \n");
        printf("\t| /|\\\n");
        printf("\t|  |  \n");
        printf("\t| /   \n");
        printf("\t------\n");

9 fallos:
        printf("\t_____\n");
        printf("\t|/ O  \n");
        printf("\t| /|\\\n");
        printf("\t|  |  \n");
        printf("\t| / \\\n");
```

```
        printf("\t------\n");
```