COMPUTADORES I

Scrabble en C

Documentación

DANIEL PEIRÓ 05302

EDUARDO DÍAZ 08103

BING ZHENG 08467

SANTIAGO MARTÍNEZ 08288

Índice

Explicación general del juego y programa

El funcionamiento del tablero tiene en cuenta todas las reglas del Scrabble tradicional en inglés y comprueba que las jugadas que se realizan cumplen con todas ellas mediante diferentes funciones y bucles.

Las reglas completas del juego se pueden encontrar en:

http://www.redeletras.com/rules/reglamento2007/reglamento2007.pdf

Reglas que el programa tablero debe imponer al jugador:

- En la primera jugada, se debe incluir la casilla central y la palabra debe ser de más de una letra.

En jugadas siguientes:

- No se pueden solapar fichas y la palabra a introducir debe conectar con las existentes en al menos un punto.
- Tras cada jugada, todas las palabras del tablero deben estar en el diccionario.

Funciones de mantenimiento de la partida:

- Llevar la cuenta de los puntos y asignar puntuaciones teniendo en cuenta multiplicadores.
- Llenar y cambiar letras de los atriles.
- Llenar y vaciar la bolsa según sea necesario.
- Contabilizar turnos y número de pases consecutivos
- Evaluar las condiciones de fin de partida.

Además de todo esto, el programa debe evitar, en la medida de lo posible, errores de introducción de valores por teclado y demás errores humanos. El programa se desarrolló en principio para el uso de ordenadores, por tanto los errores humanos solo se contemplan en controles mínimos (rango de letras dentro del tablero 15x15, strings vacíos etc.). Es decir, no es completamente fool-proof.

Servidor

La versión servidor del programa permite jugar en diferentes ordenadores utilizando la tecnología de sockets. Esta versión se compone de un programa tablero y un programa cliente. Se ejecuta en primer lugar el programa tablero (servidor) en un ordenador, se indican cuantos jugadores (clientes) van a jugar y a continuación se ejecutan tantos programas clientes como se han indicado. Mediante sockets se envía lo que necesita ver por pantalla el cliente y se reciben sus inputs por teclado.

El funcionamiento del juego ES EL MISMO que en la versión manual.

Toda la parte relacionada con sockets se ha realizado en base a este tutorial:

http://www.linuxhowtos.org/C_C++/socket.htm

Explicación breve del funcionamiento de la versión servidor:

Para evitar tener que serializar datos en una aplicación sin datos más allá de strings (o enteros convertidos desde strings) se opta por usar simplemente la función sprintf, que imprime con formato como haría printf, pero a un string. Por tanto, cambiando todas las instancias de printf en el programa manual por el sprintf correspondiente y enviando el string resultante por sockets entre servidor y cliente se consigue evitar serializar o complicar demasiado el código, en teoría.

Para lecturas de varios strings, se utilizan bucles que leen del socket hasta que reciben un string de un solo carácter '>' que indica que ha acabado la transmisión. Para resetear los buffers de lectura y escritura, se usa un bucle for que recorre el string asignando '\0'.

No se ha conseguido averiguar cuál es la causa de los fallos en la comunicación por sockets. Teóricamente debería funcionar y al tratarse de un error semi-aleatorio (pasa en distintos momentos haciendo las mismas acciones, incluso usando las mismas letras exactas), no se ha conseguido eliminar el bug. **El tablero en sí, funciona bien, como se puede comprobar con main_manual.c.**

En el código fuente de main_tablero.c se han incluido algunas ideas de porque puede ocurrir este fallo de comunicación.

Juego Manual

La versión del juego manual, main_manual.c , se ha incluido para poder comprobar que las funciones del tablero se realizan sin problemas y que el modo servidor solo tiene fallos debidos a la transmisión por sockets. Es una partida de cuatro jugadores en el mismo ordenador.

Instrucciones de compilación y ejecución

El programa se ha probado en Cygwin y funciona correctamente (salvo los bugs de comunicaciones explicados anteriormente en la versión servidor y detallados en los comentarios del código fuente). Se ha compilado con GCC, pero otros compiladores deberían valer igual (se han hecho includes a los archivos de código .c necesarios para evitar Makefiles y demás).

Compilación:

Main_manual.c se debe compilar en el mismo directorio que la carpeta func, que contiene chkfunc.c y playfuncman.c.

Main_tablero.c se debe se debe compilar en el mismo directorio que la carpeta func, que contiene chkfunc.c y playfunc.c.

Main_jugador.c se puede compilar independientemente, sin ningún otro archivo presente (es solo un visor de texto recibido por sockets del servidor y lector de inputs por teclado, enviado por sockets al servidor. Se realiza de esta forma para facilitar su teórica distribución entre jugadores sin necesidad de recompilar, basta con tener jugador.exe al margen de todo lo demás que requiere el servidor).

Ejecución:

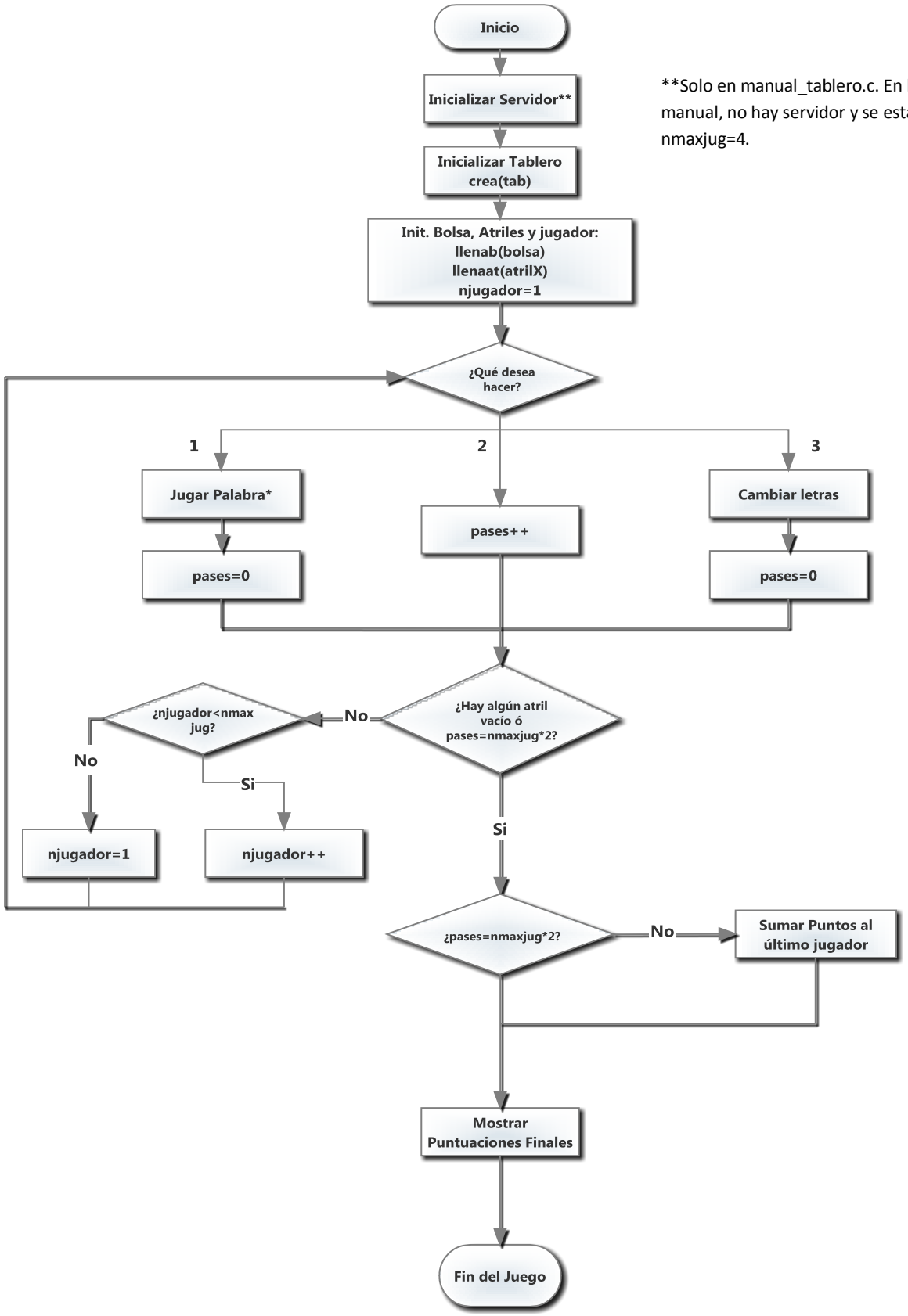
El resultado de compilar main_manual.c, se puede ejecutar sin más en el mismo directorio que dic_en. No tiene argumentos.

El resultado de compilar main_tablero.c, ejecutado en el mismo directorio que dic_en, acepta como argumento el número de puerto que se desea utilizar en la conexión. Sino, asigna 18111 por defecto.

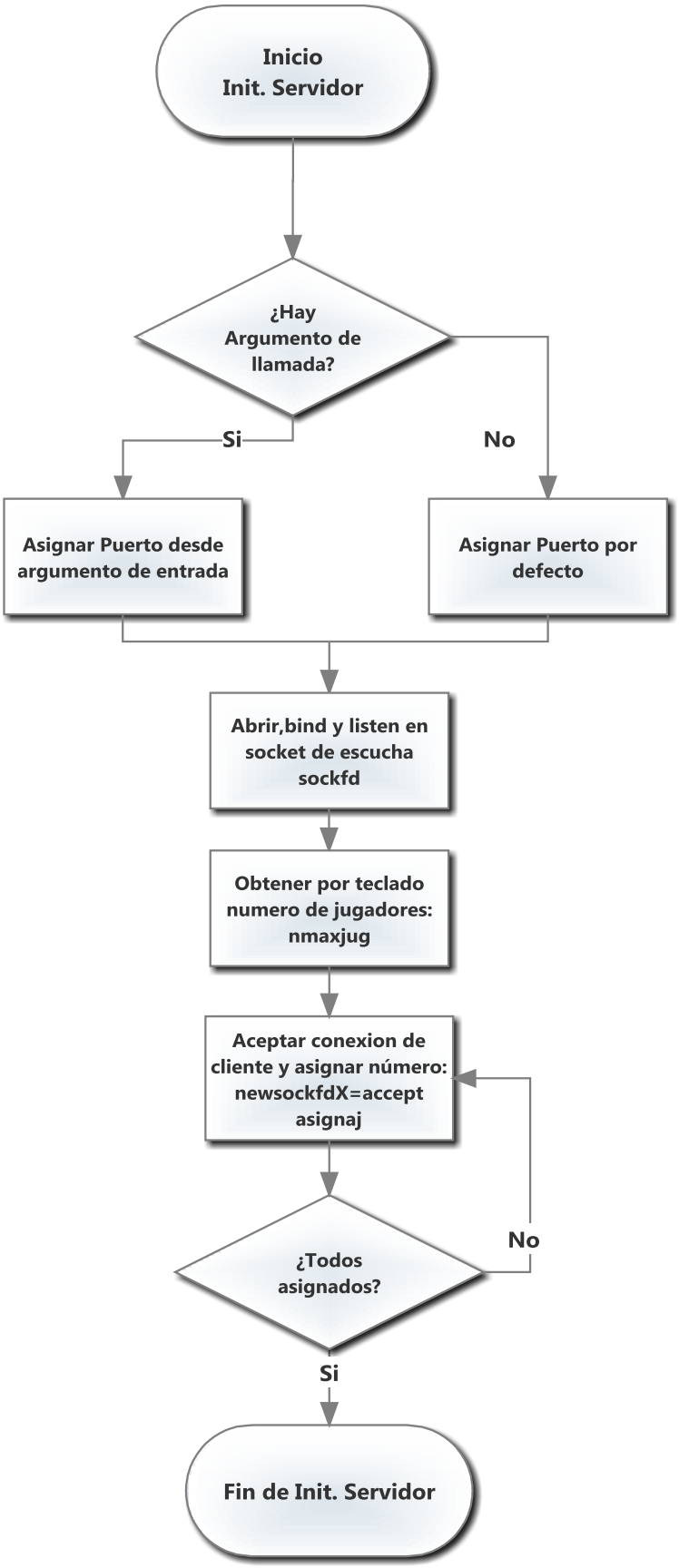
El resultado de compilar main_jugador.c, acepta como argumento host y puerto. Host es la dirección IP del ordenador ejecutando el tablero, en la red en la que están ambos. Puerto es el puerto de conexión, debe ser el mismo que en el tablero. Sino se proporcionan ambos, se establece por defecto localhost (el mismo ordenador) y 18111.

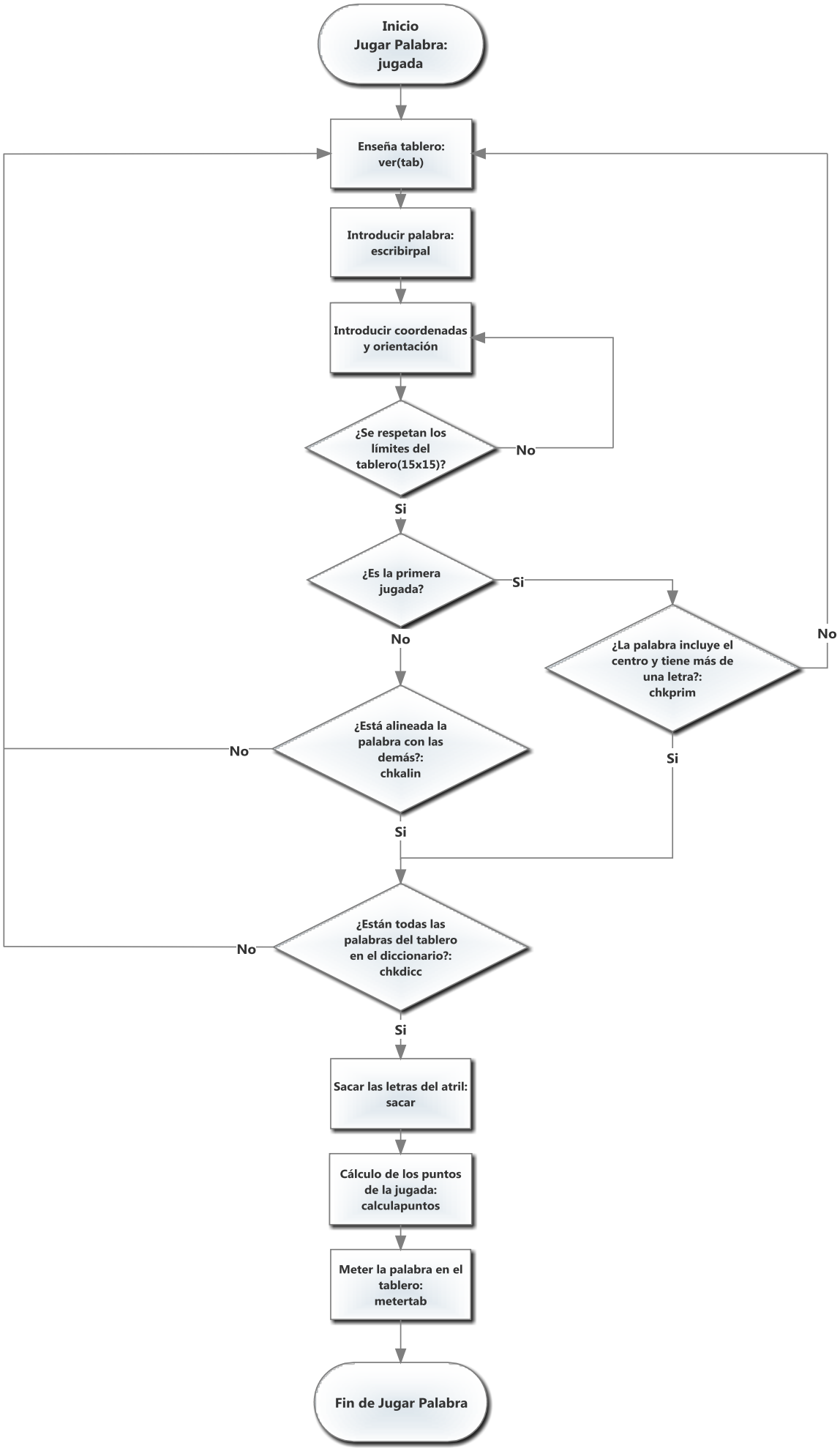
Diagramas de flujo del programa

En las siguientes hojas se explica el flujo del programa principal, asi como de algunas subrutinas mediante diagramas de flujo simplificados para mantener la claridad del desarrollo del juego. Los bloques con un asterisco al final (Init. Servidor y Jugada) tienen su propio diagrama por ser demasiado extenso para una sola página.



**Solo en manual_tablero.c. En la versión manual, no hay servidor y se establece nmaxjug=4.


Inicio
Init. Servidor
¿Hay Argumento de llamada?
Si
No
Asignar Puerto desde argumento de entrada
Asignar Puerto por defecto
Abrir,bind y listen en socket de escucha sockfd
Obtener por teclado numero de jugadores: nmaxjug
Aceptar conexion de cliente y asignar número: newsockfdX=accept asignaj
¿Todos asignados?
No
Si
Fin de Init. Servidor


Inicio
Jugar Palabra:
jugada
Enseña tablero:
ver(tab)
Introducir palabra:
escribirpal
Introducir coordenadas
y orientación
¿Se respetan los
límites del
tablero(15x15)?
No
Si
¿Es la primera
jugada?
Si
No
¿La palabra incluye el
centro y tiene más de
una letra?:
chkprim
No
Si
¿Está alineada la
palabra con las
demás?:
chkalin
No
Si
¿Están todas las
palabras del tablero
en el diccionario?:
chkdicc
No
Si
Sacar las letras del atril:
sacar
Cálculo de los puntos
de la jugada:
calculapuntos
Meter la palabra en el
tablero:
metertab
Fin de Jugar Palabra

Explicación breve de las funciones

Se incluyen a continuación unas breves explicaciones de las funciones y rutinas escritas para el programa. Una versión más específica se puede encontrar al inicio del código fuente de cada una de ellas, con detalles más relacionados con el propio código que las compone además de su funcionalidad. Las funciones en playfunc.c son prácticamente iguales que las de playfuncman.c solo que incluyen código específico de transmisión de datos por sockets. Por esto no se incluyen en las explicaciones, al ser iguales en funcionamiento a las de playfuncman.c.

Además de estas explicaciones textuales, **todas las funciones tienen comentarios paso a paso en su código, para facilitar la comprensión de las mismas.**

CHKFUNC.C

Estas funciones son las encargadas de hacer las comprobaciones necesarias para que el programa se siga ejecutando de acuerdo a las normas del Scrabble.

1. **chkpalab**: Se encarga de comprobar si la palabra está en el atril, recorriéndolo y comparando con la palabra letra por letra. Si una de las letras de la palabra es el espacio, no comprueba ese carácter.

2. **chkdicc:** Comprueba si la palabra que se pasa como argumento está en el diccionario. Es importante destacar que se pasa la palabra a minúsculas porque el archivo de texto dónde están las palabras tiene su contenido escrito en minúsculas.

3. **chkalin:** Se asegura de que haya continuidad en la formación de las palabras. Es decir, que no haya choques entre letras de diferentes palabras, y que la nueva palabra tenga una letra de otra palabra.

4. **chktab:** Comprueba que todas las palabras que hay sobre el tablero están en el diccionario.

5. **chkprim**: Comprueba si en la primera jugada se ocupa la casilla central y si la palabra tiene más de una letra.

6. **atrilvacio:** Comprueba si el atril está vacío.

PLAYFUNCMAN.C

Estas funciones son las que forman las acciones que debe tomar el tablero para mantener el flujo del programa:

1. **crea:** Crea el tablero, entendiéndolo como una matriz de 15x15.Cada casilla tiene incluida la puntuación correspondiente y un elemento visual para facilitar su caracterización.

2. **ver:** Simplemente imprime por pantalla el tablero con indicaciones de dobles y triples puntos.

3. **llenab:** Llena un array de tipo ficha con las letras necesarias para repartir a los jugadores: La bolsa de fichas.

4. **llenaat:** Asigna una ficha aleatoria de la "bolsa" a los elementos vacíos del atril, comprobando que cada elemento está vacío.

5. **sacar:** Saca la palabra formada como un array auxiliar. Esta función tiene dos formas de operación. Una es en modo de comprobación, donde no tocamos el atril, utilizando un atril auxiliar. El otro modo es el formato "ya comprobado", donde lo que hacemos es quitar las fichas del atril.

6. **metertab:** Introduce la palabra en el tablero. Las comprobaciones de posibles errores ya se han hecho y no competen a esta rutina.

7. **calculapuntos:** Calcula los puntos de la jugada teniendo en cuenta la formación de otras palabras aparte de la que formamos originalmente, y los multiplicadores que haya.

8. **escribirpal:** Obtiene la palabra que introduce el jugador. También se comprueba que la palabra se puede formar con elementos del atril y que no está formada exclusivamente por espacios.

9. **`jugada:`** Es la función principal. Es la que conduce el programa con todos los datos de los que dispone. Hace las comprobaciones necesarias, enseña el tablero, pide los datos necesarios, saca las letras del atril, modifica el tablero y calcula los puntos. Todo esto se hace con las funciones anteriores, sólo que en esta función se va haciendo que cada elemento de la jugada siga al siguiente. Se puede ver de la siguiente manera: las rutinas distintas de jugada son los procesos mentales que efectuamos al hacer una jugada o representaciones de procesos inherentes al juego como llenar el atril o ver el tablero en todo momento, mientras que la jugada sería la ejecución de estas ideas.

10. **`cambiar:`** Cambia las letras del atril por otras de la bolsa.

Diccionario

El diccionario se ha implementado de la forma más básica posible, con un diccionario en texto (una palabra por línea) y una función que realiza una comparación entre su argumento y cada una de las líneas de dicho diccionario (archivo dic_en). Se intentó en un primer momento implementar librerías como hunspell y aspell pero resultaron ser difíciles de instalar y a nuestro juicio un poco excesivas para las necesidades tan básicas que se requerían para el programa. Así, se optó por este método más simple, aunque seguramente más ineficiente en cuanto a utilización de recursos.

El resto de la documentación se puede encontrar en el código fuente.