

Práctica 2 FP1 2019

Dominó V2

1. Descripción de la funcionalidad de Dominó V2.

En la segunda versión vamos a utilizar dos arrays de enteros para almacenar las fichas que se pueden robar en la mesa (llamada pozo, inicialmente con las 28 fichas posibles) y las fichas del jugador.

Inicio del juego:

Se debe generar el pozo con las 28 fichas posibles.

Se roba al azar una ficha del pozo y se coloca en el tablero.

El jugador roba al azar 7 fichas del pozo.

Las fichas recogidas del pozo se eliminan del pozo.

Desarrollo del juego:

Antes de cada jugada se muestran las fichas colocadas en la mesa (tablero), el número de fichas colocadas por el jugador, el número de fichas robadas por el jugador y las fichas restantes del jugador.

Las opciones que puede elegir el jugador en su turno son:

1) Colocar una de sus fichas a la **izquierda** del tablero. En este caso se pregunta cuál de las fichas del jugador se desea colocar a la izquierda del tablero.

Sólo se puede colocar una ficha si uno de sus valores coincide con el valor del extremo del tablero.

Si se coloca la ficha debe representarse en el tablero en la posición correcta.

La ficha colocada se elimina de la lista de fichas del jugador.

2) Colocar una de sus fichas a la **derecha** del tablero. En este caso se pregunta cuál de las fichas del jugador se desea colocar a la derecha del tablero.

Sólo se puede colocar una ficha si uno de sus valores coincide con el valor del extremo del tablero.

Si se coloca la ficha debe representarse en el tablero en la posición correcta.

La ficha colocada se elimina de la lista de fichas del jugador.

3) Robar una ficha aleatoriamente del pozo que se elimina del pozo y se añade a la lista de fichas del jugador.

No se permite robar si es posible colocar alguna ficha del jugador en el tablero.

4) Salir del juego. En este caso se muestran los puntos del jugador y se termina el programa.

La Figura 1 muestra un ejemplo de ejecución del programa.

```
| TABLERO |
-----
|1-1||1-4||4-5||5-6||6-6|

Fichas colocadas: 4 - Fichas robadas: 1
Fichas del jugador: |0-2| |2-5| |2-3| |1-2|


-----| MENU DE OPCIONES |
-----1. Poner ficha por la izquierda
2. Poner ficha por la derecha
3. Robar ficha nueva
0. Salir

Elija opcion: 1
```

Figura 1: Ejemplo de ejecución

Final del juego:

Una ronda de dominó termina en uno de los siguientes casos:

1) Si el jugador coloca todas sus fichas.

2) Si no quedan fichas para robar en el pozo y el jugador no puede poner ninguna ficha. En este caso se debe mostrar los puntos del jugador (la suma de los dos valores de sus fichas) antes de salir.

3) Si se selecciona la opción de Salir

Funcionalidadesopcionales:

- Salvar la partida en un archivo, para cargarla más adelante y continuar con ella.
- Permitir que el programa se pueda configurar para jugar con otras variantes del juego: entre 6 y 9 como máximo de puntos de las fichas.

2. Detalles de implementación de la versión 2

En la versión 2 el pozo y las fichas del jugador se almacenan en dos arrays de enteros que representan el valor izquierdo y derecho de cada ficha, junto con una variable entera contador de fichas. El tablero se puede seguir almacenando en un string.

```
const int NumFichas = 28;
typedef short int tArray[NumFichas];
```

Además deberás incorporar los subprogramas necesarios decidiendo el tipo de retorno, el tipo de cada argumento y si cada argumento se pasa por valor o por referencia. Algunas de dichas funciones podrían ser las siguientes:

```
void generaPozo(tArray pozol, tArray poz02, int maxValor);

void mostrarTablero(string tablero, short int numColocadas,
short int numRobadas, const tArray fichas1, const tArray
fichas2, short int fichasCont);

void robaFicha(const tArray pozol, const tArray poz02, short
int& cont, short int& fichaN1, short int& fichaN2);

void eliminaFicha (tArray fichas1, tArray fichas2, short int&
fichasCont, short int fichaNum);

bool puedeColocarFicha(const tArray fichas1, const tArray
fichas2, short int fichasCont, string tablero);

short int sumaPuntos(const tArray fichas1, const tArray
fichas2, short int fichasCont);
```

Para desordenar el pozo podemos utilizar el algoritmo de Fisher-Yates, que desordena cualquier array de manera no sesgada en un recorrido:

```
void desordenaPozo(tArray pozol, tArray poz02) {
    int idx, i;
    short int tmp1, tmp2;
    for (int i = NumFichas - 1; i >= 0; i--) {
        idx = rand() % (i + 1);
        if (i != idx) {
            tmp1 = pozol[i];
            tmp2 = poz02[i];
            pozol[i] = pozol[idx];
            poz02[i] = poz02[idx];
            pozol[idx] = tmp1;
            poz02[idx] = tmp2;
        }
    }
}
```