---

# \*\*Guía para Agregar Nuevas Funcionalidades a un Proyecto Modularizado\*\*

## \*\*Instrucciones para Agregar un Nuevo Juego/Funcionalidad\*\*

### \*\*Modificaciones en `main.c`:\*\*

1. \*\*Incluir los nuevos encabezados:\*\*

- Utiliza las líneas \*\*15 a 17\*\* para agregar los `#include` necesarios de tu nueva funcionalidad.

2. \*\*Actualizar el menú principal:\*\*

- En las líneas \*\*26 a 28\*\*, añade un nuevo `case #` en el menú principal, asegurándote de modificar los números de los casos existentes.

3. \*\*Incorporar la lógica del juego en el bucle principal:\*\*

- En el `while` de la línea \*\*107\*\*, agrega un bloque como este:

```c

case #:

/\* Código del juego \*/

break;

```

- Asegúrate de ajustar la condición de salida si es necesario.

4. \*\*Integrar la función principal del nuevo juego:\*\*

- Cambia el nombre de la función principal de tu nuevo juego (no puede llamarse `main`) y llámala desde el `main.c`. Hay tres enfoques posibles:

- \*\*Uso directo:\*\* Las variables internas no afectan las globales; solo asegúrate de incluir un mecanismo para salir.

- \*\*Con parámetros específicos:\*\* Pasa `&jugador1` para modificar los valores de nombre y puntaje en el archivo de registro/ranking.

- \*\*Con todos los parámetros:\*\* Usa la estructura y variables de `jugarPartida` en la línea \*\*128\*\*.

---

### \*\*Modificaciones en la Carpeta del Proyecto:\*\*

1. \*\*Crear los archivos necesarios:\*\*

- Crea al menos dos archivos:

- `cambiar\_nombre.c`: Contendrá la lógica principal del juego.

- `cambiar\_nombre.h`: Declarará las funciones públicas del juego.

2. \*\*Estructura en `cambiar\_nombre.c`:\*\*

- La función principal debe renombrarse y declararse como `elejirNombreFuncion(parametros)`.

3. \*\*Estructura en `cambiar\_nombre.h`:\*\*

- Incluye al menos la declaración de:

```c

void elejirNombreFuncion(parametros);

```

- Asegúrate de incluir todas las dependencias necesarias.

4. \*\*Verificación de compatibilidad:\*\*

- Comprueba que no existan conflictos con funciones o estructuras ya implementadas.

---

### \*\*Ejemplo de Encabezado para `cambiar\_nombre.h`:\*\*

```c

#ifndef CAMBIAR\_NOMBRE\_H

#define CAMBIAR\_NOMBRE\_H

#define SALDO\_INICIAL 1000

#include "config.h"

#include "mazo.h"

#include "logica.h"

#include "jugador.h"

#include "scoreboard.h"

#include "record.h"

// Declaración de funciones públicas

void nuevaFuncionalidad();

void nuevoJuego(myDeck \*mazo, Jugador \*jugador1);

#endif // CAMBIAR\_NOMBRE\_H

```

---

## \*\*PENDIENTES / Funcionalidades Posibles\*\* \*(Actualizado al 19/11)\*

### \*\*Mejoras Lógicas:\*\*

1. Lógica para la opción de \*\*doblar apuesta\*\*.

2. Recuperar apuesta en caso de \*\*derrota con blackjack\*\*.

3. Implementar lógica para \*\*dividir cartas\*\*.

4. Revisar y solucionar problemas con el índice en `manoInicial`.

5. Leer e imprimir reglas del blackjack europeo desde un archivo `.txt`.

6. Exportar el registro del juego a formato \*\*TXT/CSV\*\*.

7. Validar la integridad de los archivos con \*\*hashing\*\*.

8. Implementar función para importar y fusionar registros de otros jugadores.

9. Incorporar funciones de búsqueda y ordenamiento para un registro extenso.

### \*\*Mejoras en el Proyecto:\*\*

1. Verificar la compilación con archivos distribuidos en subcarpetas.

2. Crear un archivo `CMakeLists.txt`.

3. Empaquetar el proyecto en un instalador.

4. Considerar soporte para otros sistemas operativos.

5. Decidir entre configuración mediante `.env` o `.txt`.

---

## \*\*Otras Rutas de Desarrollo\*\*

### \*\*Opción A: Mejoras Gráficas\*\*

- Implementar gráficos con bibliotecas externas.

- Mejorar la experiencia en terminal mediante representaciones visuales (como cartas en ASCII).

- Revisar e integrar funciones con `printf` existentes para soportar estas mejoras.

### \*\*Opción B: Nuevos Juegos y Funciones\*\*

1. Agregar variantes de blackjack (europeo y americano).

2. Incluir nuevos juegos con baraja francesa, como póker.

3. Soporte para juegos que utilicen múltiples mazos.

4. Implementar un sistema de \*\*ID de usuario\*\* para retomar partidas.

5. Mejorar funciones actuales con conceptos avanzados:

- Paridad

- Algoritmo de Hamming

- Algoritmo de Luhn

- Módulo 11

- Hashing

- Regresión lineal

- Tablas de salto (`jump table`)

- Listas enlazadas y doblemente enlazadas

---

## \*\*Estructura del Proyecto Sugerida\*\*

```plaintext

blackjack/

├── include/

│ ├── config.h

│ ├── jugador.h

│ ├── logica.h

│ ├── mazo.h

│ ├── record.h

│ ├── scoreboard.h

├── src/

│ ├── config.c

│ ├── jugador.c

│ ├── logica.c

│ ├── mazo.c

│ ├── record.c

│ ├── scoreboard.c

├── .env

├── main.c

├── README/

│ ├── Changelog.txt

│ ├── Modularizar.txt

│ ├── Compilacion.txt

│ └── Implementacion\_Funciones.txt

└── Makefile

```

---