

## **RULETA EUROPEA**

### **I. INTRODUCCIÓN:**

La ruleta europea es uno de los juegos más típicos de los casinos, este a simple vista da la impresión de que es un juego muy complejo, pero el mecanismo de este juego no tiene ninguna complicación.

Juegues donde juegues, siempre lo harás contra el casino en sí. Aquí cada uno de los usuarios reciben fichas con diferentes valores y esto es lo que se irán apostando en cada tirada. Aquí es el crupier el que se encarga de hacer girar la ruleta y tirar la bolita en ella. Además, este también tendrá que comprobar el lugar en el que se esta cae y nombrar al ganador de cada partida.

La ruleta europea también es llamada la ruleta francesa. Esta tuvo sus orígenes en siglo VII, cuando el matemático Pascal creó la primera ruleta. Su nombre proviene del vocablo de origen francés "Roulette".

Este juego posteriormente se expandió por toda Europa de la mano de los hermanos Blanc los cuales fueron los primeros en implantarlo en el casino de Monte Carlo.

Este posteriormente se expandió a América y allí el mecanismo principal del juego sufrió una pequeña modificación.

### **II. IDEA GENERAL DEL ALGORITMO .-**

El programa comienza pidiendo cuantas rondas el jugador desea jugar (se establece que el máximo número de rondas es 20). Una vez que el usuario ingresa el número de rondas, el juego comienza.

Cada ronda consiste en un número determinado de apuestas, en este caso, el máximo número de apuestas es 10. El programa imprime en la consola un menú de opciones de las apuestas donde el usuario puede elegir, luego el programa solicitará al usuario la cantidad de fichas por apuesta..

La ronda culminará cuando el usuario decida dejar de apostar o haya hecho todas las apuestas posibles. Al finalizar la ronda, se mostrará el número que salió en la ruleta , las ganancias y las pérdidas de la mesa en esa ronda.

Y se continuará con la siguiente ronda ( si el usuario indicó más de una ronda).

Al culminar con la totalidad de las rondas, el programa imprimirá: Las ganancias y pérdidas totales de la mesa en todo el juego. Si la mesa es ganadora, no conforme o en esta en problemas. El promedio de la cantidad de apuestas por ronda. Si en más de 5 rondas seguidas la bola cayó en número de igual color y mostrará cuántas veces cayó en un número primo.

### III. ESTRUCTURA DEL PROGRAMA

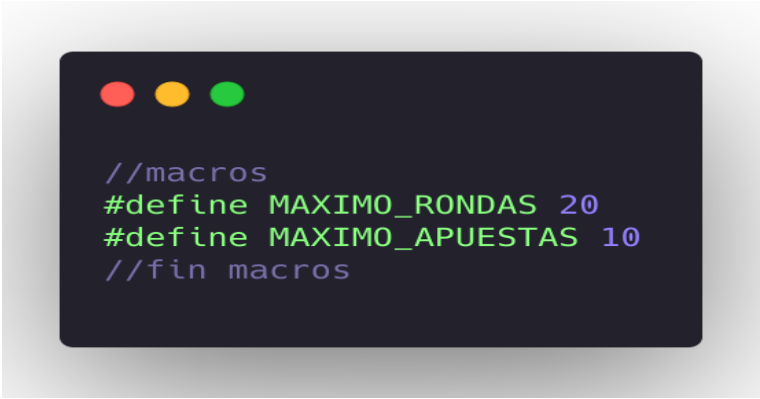
#### MACROS.-

En este programa declare los siguientes macros:


**a) MAXIMO\_RONDAS:** Es el máximo número de rondas que el usuario puede elegir; el usuario puede jugar hasta máximo 20 rondas.

**b) MAXIMO\_APUESTAS:** Es el máximo número de apuestas que el usuario puede realizar en cada ronda; sólo puede realizar 10 apuestas.

**c) terminal\_formats\_active:** Es una constante que debe estar definida para poder usar los formatos de colores de texto en la terminal, en el caso que no esté definida, todas las constantes de formatos no van agregar ningún formato.



```
//macros
#define MAXIMO_RONDAS 20
#define MAXIMO_APUESTAS 10
//fin macros
```



```
#define terminal_formats_active //para activar el formato de salida
```

## VARIABLES.-

En esta parte solo mencionare las variables más importantes, las cuales son a continuación:

**a) rondas:** Es un int, el cual almacena la cantidad de rondas a jugar.

**b) cantidadApuestas:** Es un int que contabiliza el total de la cantidad de las apuestas realizadas.

**c) numerosGanadores:** Es un array de tipo int y con un tamaño de `MAXIMO_RONDAS`; en esta variable se almacena el número que sale cuando se ejecuta la función `lanzarBola`, es el número que salen en la ruleta.

**d) perdidasPorRonda:** Es un array de tipo int y con un tamaño de `MAXIMO_RONDAS`; en esta variable se almacena las pérdidas de la mesa por ronda.

**e) gananciasPorRonda:** Es un array de tipo int y con un tamaño de `MAXIMO_RONDAS`; en esta variable se almacena las ganancias de la mesa por ronda.

**f) totalPerdidas:** Es un int que contabiliza las pérdidas totales de la mesa en el juego.

**g) totalGanancias:** Es un int que contabiliza las ganancias totales de la mesa en el juego.

```
int rondas; // cantidad de rondas
int cantidadApuestas = 0;
int numerosGanadores[MAXIMO_RONDAS];
int perdidasPorRonda[MAXIMO_RONDAS];
int gananciasPorRonda[MAXIMO_RONDAS];
int totalPerdidas = 0;
int totalGanancias = 0;
```

## **FUNCIONES.-**

**El programa tiene 3 funciones principales:**

**a) void JugarRondas()**

**b) void JugarRonda(int ronda, int rondas, int \*p\_cantidadApuesta,int gananciasPorRonda[ ], int perdidasPorRonda[], int numerosGanadores[ ])**

**c) void menuApuestas(int apuesta[ ], int cantidadFichas[ ], int ronda, int rondas, int \*p\_cantidadApuesta)**

```
//prototipos de mis funciones
void JugarRondas();
void JugarRonda(int ronda, int rondas, int *p_cantidadApuesta,int gananciasPorRonda[], int
perdidasPorRonda[], int numerosGanadores[]); // ronda actual que se esta jugando
void menuApuestas(int apuesta[], int cantidadFichas[], int ronda, int rondas, int *p_cantidadApuesta);
//fin prototipos de mis funciones
```

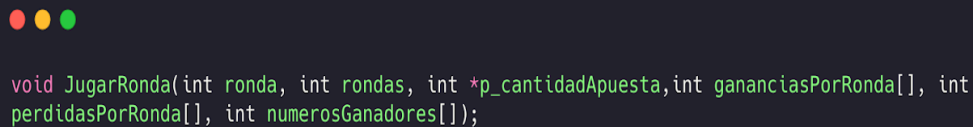
**a) void JugarRondas():** Esta función no retorna nada y no tiene parámetros, todo sucede dentro de la misma. Esta función es el génesis de mi programa.

Solicita al usuario el número de rondas a jugar y las ejecuta, llamando a la función *jugarRonda*.

```
void JugarRondas();
```

- b) ***void JugarRonda(int ronda, int rondas, int \*p\_cantidadApuesta, int gananciasPorRonda[ ], int perdidasPorRonda[ ], int numerosGanadores[ ])***: No retorna nada, recibe como parámetros a int ronda, int rondas, un puntero a int p\_cantidadApuesta, int array gananciasPorRonda[ ], int array perdidasPorRondas[ ] y int array numerosGanadores[ ].

Esta función se encarga de llevar a cabo la ronda. Dentro de ella se llama a la función menuApuestas luego que se ejecuta, se manda a llamar a la función lanzarBola y al finalizar imprime en consola, la cantidad de ganancias y pérdidas de la mesa en esa ronda, dando la por concluida.




```
void JugarRonda(int ronda, int rondas, int *p_cantidadApuesta, int gananciasPorRonda[], int perdidasPorRonda[], int numerosGanadores[]);
```

- c) ***void menuApuestas(int apuesta[ ], int cantidadFichas[ ], int ronda, int rondas, int \*p\_cantidadApuesta)***: Esta función no retorna nada, recibe como parámetros: int array apuestas[ ], int array cantidadFichas[ ], int ronda, int rondas y a un puntero int \*p\_cantidadApuesta.

En cada ronda, la función mostrará por consola cuantas rondas usted está jugando.

Imprime un menú de apuestas, donde se le indica al usuario las opciones que puede apostar y se le da la posibilidad de ingresar sus apuestas.



```
void menuApuestas(int apuesta[], int cantidadFichas[], int ronda, int rondas);
```

**Además, tiene las siguientes funciones 8 secundarias:**

**d) *int obtenerFicha(int \*ficha)*:** Esta función recibe como parámetro un puntero a *int ficha*; tiene un bucle que se ejecutará de acuerdo al número de fichas, solicitará al usuario que ingrese las fichas para apostar a la opción que seleccionó.

El usuario puede apostar a las fichas: 1, 2, 3, 5, 10, 50, 100, el mismo puede ingresar muchas fichas y si quiere dejar de ingresar fichas tiene que presionar el número 0.

Al finalizar se asigna a la variable pasada como parámetro, la suma de las fichas que el usuario ingreso.

**e) *void pedir\_verificarFicha(int arreglo[ ], int contador, int opcion, int apuesta[ ])*:** No retorna nada, recibe como parámetro a un *int array [ ]*, *int contador*, *int opcion* y *int array apuesta [ ]*.

Esta función (como su propio nombre indica), va a verificar y pedir las fichas.

Dentro de un bucle, llama a la función *obtenerFicha*, una vez que se obtiene la ficha se verifica si es mayor a 0, si cumple la condición, la ficha se guardará en un arreglo en la posición[contador], así también verifica si la opción ingresada por el usuario son aquellas que pagan doble (el usuario no puede repetir la cantidad de fichas en las opciones que pagan doble), para verificar esto se llama a la función *perteneceArrYApuestaDoble*, la cual buscará en las apuestas realizadas si existe una apuesta con ganancia doble y con el mismo valor.

**f) *int lanzarBola()*:** Retorna un *int numeroRandom*, no tiene parámetros. Se encarga de generar de manera random el número ganador (donde cae la bola) oscila entre el 0 al 36 y una vez que se obtiene el número ganador, se llama a la función *numeroEsRojo* para verificar si el número es rojo y darle dicho formato, caso contrario se imprime el número en color negro con un fondo blanco.

**g) *int numeroEsRojo(int numero)*:** Retorna un 1, 0 o -1, depende del caso, recibe como parámetro un *int numero*. Esta función se encarga de verificar si el número es rojo, esto se verifica mediante un switch, si el número coincide con algunos de los casos, devolverá 1, si no devolverá un cero o -1 si no es ninguno de los dos.

**h) *perteneceArrYApuestaDoble(int ficha, int arreglo[], int limite, int apuestas[])*:** Retorna 0 o 1 dependiendo del caso, recibe como parámetros un int ficha, array tipo int, int limite, int array apuesta [ ].

Esta función tiene un bucle que recorrerá el arreglo hasta la posición limite y verificará si la ficha se encuentra en el arreglo y si la apuesta en el mismo índice es una ganancia doble (para eso se llama a la función *esApuestaConGananciaDoble*), si esto es verdadero, retornará un 1, caso contrario retornará un 0.

**i) *int esApuestaConGananciaDoble(int opcion)*:** Esta función recibe como parámetro un int opcion. Verifica si el valor pasado como parámetro es una apuesta de ganancia doble, de ser el caso retorna un valor verdadero si no un valor falso.

**j) *int contarColor(int arreglo[], int dimension)*:** Esta función cuenta si el color rojo o negro aparece más de 5 veces o si aparecen los dos en el arreglo pasado como parámetro, ( el arreglo es recorrido hasta la posición indicada en el parámetro dimensión). Retorna 2 si el color rojo y negro aparecen más de 5 veces, retorna el 1 es rojo o 0 si es negro y retorna -1 cuando no aparecen más de 5 veces.

**k) *int contarPrimos(int arreglo[], int dimension)*:** Retorna el contador, recibe como parámetros a un array tipo int y int dimension.

Esta función va a contar cuántos números primos hay en el arreglo que se le pasa como parámetro.

**l) *int esPrimo(int numero)*:** Esta función retorna 0 o 1 dependiendo el caso, recibe como parámetro un int numero.

Esta función verifica si el número que recibe como parámetro es primo, si lo es retorna 1, caso contrario retorna 0.

## LIBRERÍAS

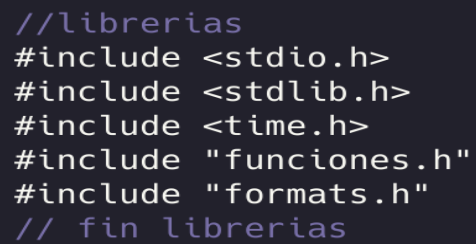
En este programa se utiliza las siguientes librerías:

### LIBRERÍAS EXTERNAS:

- **stdio.h**
- **stdlib.h**
- **time.h**

### LIBRERÍAS INTERNAS:

- **funciones.h**
- **formats.h**



```
//librerias
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "funciones.h"
#include "formats.h"
// fin librerias
```