# **Rulett: Programozói Dokumentáció**

## **Bevezetés**

A rulett egy klasszikus kaszinójáték, amelyben a játékosok téteket helyeznek el egy forgó keréken szereplő számok, színek vagy számkategóriák szerint. A programozás során a rulett logikáját kell modellezni és interaktív működést biztosítani a felhasználó számára.

## **Játékmenet Folyamata**

1. **Bejelentkezés**:
   1. A felhasználó beírja a nevét és a feltöltendő kezdő egyenleget.
   2. A felhasználó kiválasztja, hogy hány virtuális ellenfél legyen (maximum 6).
2. **Tétek Megadása**:
   1. A felhasználó kiválasztja, hogy melyik számra, színre, vagy számkategóriára (pl. páros-páratlan) helyezi el a tétet.
   2. Az egyes tétek összege levonásra kerül az egyenlegből.
3. **Kerék Pörgetése**:
   1. Egy véletlenszám-generátor (pl. Random osztály) segítségével meghatározza nyertes számot (és szín).
4. **Eredmények Értékelése**:
   1. Ellenőrizzük, hogy a játékos tétei egyeznek-e a nyertes számmal vagy kategóriával.
   2. Az egyező tétekhez tartozó szorzók alapján kiszámolódik a nyeremény.
5. **Játék Folytatása**:
   1. A felhasználó eldönti, hogy tovább szeretne játszani, vagy kilép a játékból.

## **Szorzók és Típusú Tétek**

* **Egy számra tét** (straight-up): 35-szörös szorzó.
* **Színre tét** (piros/fekete): 2-szeres szorzó.
* **Páros/páratlan**: 2-szeres szorzó.
* **Kisebb/Nagyobb** (1-18 vagy 19-36): 2-szeres szorzó.
* Meg kell ide pár

## **Programozási Összetevők**

### **1. Véletlenszám Generálás**

A nyertes szám meghatározására a C# Random osztályát használjuk:

Random random = new Random();  
int winningNumber = random.Next(0, 37); // 0-36 között generál egy számot

### **2. Tétek Kezelése**

A tétek és szorzók egy összetett adatszerkezetben tárolhatók:

Dictionary<string, int> bets = new Dictionary<string, int>();  
bets["1"] = 100; // 100 egységes tét az 1-es számra  
bets["odd"] = 200; // 200 egységes tét a páratlan számokra

### **3. Eredmények Kiértékelése**

Az állapot és a tétek egyezésének ellenőrzése:

|  |
| --- |
| if (bets.ContainsKey(winningNumber.ToString())) {  int payout = bets[winningNumber.ToString()] \* 35;  Console.WriteLine($"Nyeremény: {payout}"); } if (winningNumber % 2 != 0 && bets.ContainsKey("odd")) {  int payout = bets["odd"] \* 2;  Console.WriteLine($"Nyeremény: {payout}"); } |

## **Felhasználói Felület**

Egy egyszerű **Windows Forms** vagy **WPF** alkalmazás biztosítható, amely lehetővé teszi a tétek grafikus kezelését, illetve az eredmények megjelenítését.

## **Összegzés**

A rulett implementálása egy interaktív programban magában foglalja a véletlenszám-generálást, a tétek logikájának kezelését és a felhasználói felület megtervezését. A szabályok egyszerűek, de a logika és a programozási struktúrák lehetővé teszik a játék komplexitásának növelését.