Universidad Rafael Landívar
Laboratorio de Programación
Ingeniería Química
Catedrático: Ing. Edwin Chocoy

PROYECTO PRÁCTICO NO. 2
Dados

Vásquez Ruiz, Nadia María-1140023

Guatemala, 13 de noviembre de 2023

ÍNDICE

INTRODUCCIÓN

En este proyecto se realizó una simulación de dados, utilizando las habilidades y herramientas aprendidas a lo largo del curso de programación. En la simulación, el jugador tenía dos dados de seis caras, cada una con un número 1,2,3,4,5,6, se tiran los dados al mismo tiempo y siguiendo las condiciones establecidas, la suma de los valores resultantes en cada dado indica si el jugador gana o pierde puntos, en el caso de perder la “La casa” gana los puntos.

Para realizar el código se utilizó el ciclo for para generar las partidas y tiros solicitados por el usuario, para asignar los puntos ganados en cada tiro se utilizó la condición if y se utilizaron variables contadoras para llevar el registro de la cantidad de tiros pares, impares, iguales y ganados por el jugador. Además, se utilizó el método random.Next() para generar los valores de los dados en cada tiro. Al final se obtuvo un código que muestra las partidas y los resultados solicitados.

ANÁLISIS

Entrada

Tabla 1: Entradas del código dados

Variable	Descripción
cantidadPartidas	El usuario debe ingresar la cantidad de partidas que desee jugar.
tirosPorPartida	El usuario debe ingresar la cantidad de tiros por partida, cumpliendo con la condición que sea un número par.

Fuente: Elaboración propia 2023.

Procesos

Tabla 2: Procesos del código dados

Variable	Descripción
puntajeJugador	Si la suma de los dados en el primer tiro es igual a 6 o 12, el jugador gana el tiro sumando 12 puntos. Si en el turno del jugador la suma de los dados es 2,3,5,7,8 o 9, se le agrega el resultado de la suma a su puntaje. El jugador puede perder si no lleva ningún punto y la suma de los dados es 11, en este caso la casa gana 6 puntos.
puntajeCasa	Si en el primer tiro la suma de los dados es 4 o 10, la casa gana el tiro sumando 12 puntos. Si en el turno de la casa la suma de los dados es 2,3,5,7,8 o 9, se le agrega el resultado de la suma a su puntaje.
ganaParCasa	Almacena la cantidad de partidas que gana la casa
ganaParJugador	Almacena la cantidad de partidas que gana el jugador

Fuente: Elaboración propia 2023.

Salida

Tabla No.3: Salidas del código dados

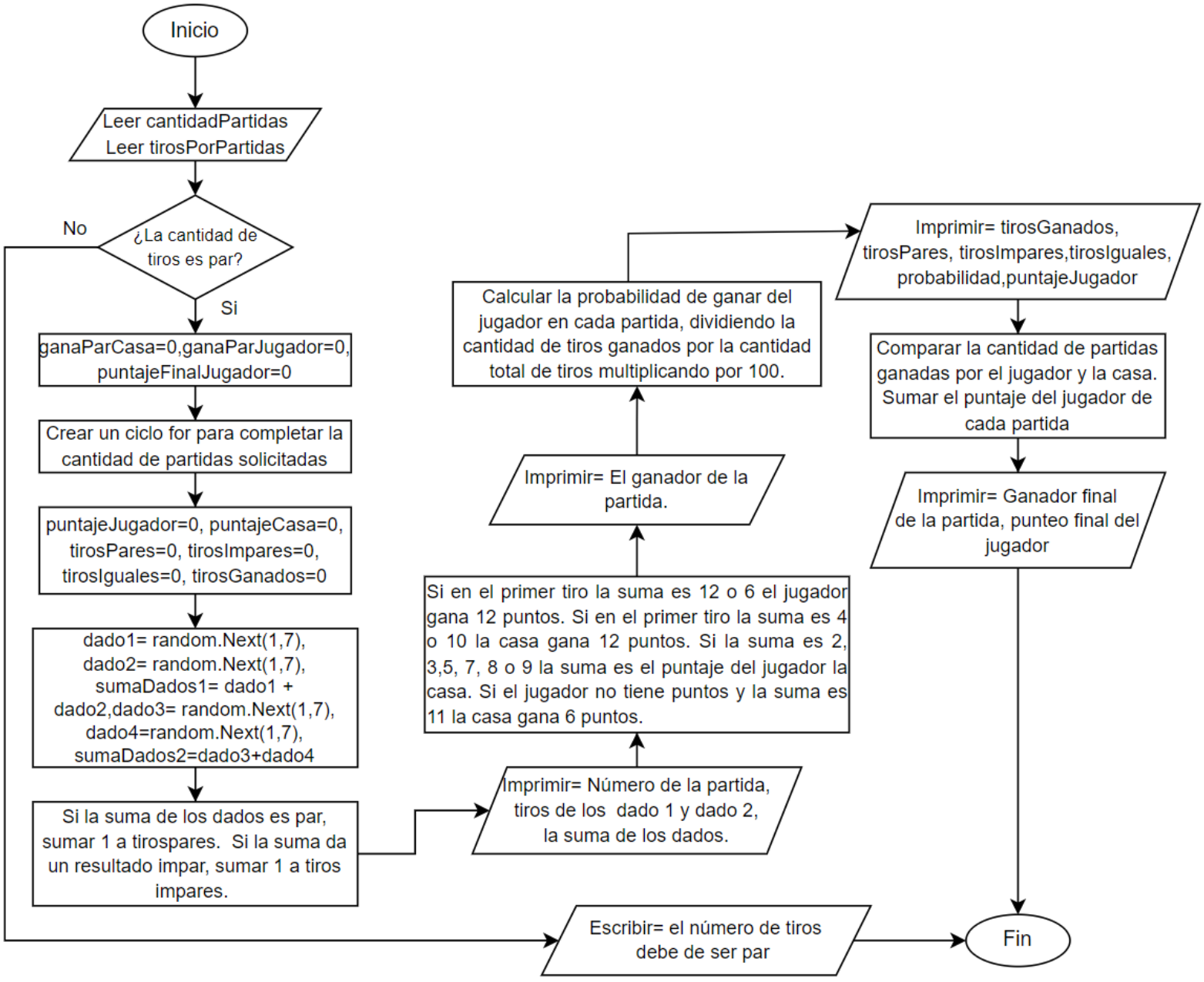
Variable	Descripción
tirosPares	Muestra la cantidad de tiros donde la suma de los dados es un número divisible dentro de 2, siendo un tiro par.
tirosImpares	Muestra la cantidad de tiros donde la suma de los dados es un número no divisible dentro de 2, siendo un tiro impar.
tirosIguales	Muestra la cantidad de tiros donde los valores de los dados coinciden entre sí, siendo un tiro igual.
tirosGanados	Muestra la cantidad de tiros que ganó el jugador.
dado1	Muestra un valor aleatorio de 1 a 6
dado2	Muestra un valor aleatorio de 1 a 6
sumaDados1	Muestra la suma de los dados correspondientes al jugador

dado3	Muestra un valor aleatorio de 1 a 6
dado4	Muestra un valor aleatorio de 1 a 6
sumaDados2	Muestra la suma de los dados correspondientes a la casa
probabilidad	Muestra la probabilidad de ganar del jugador en cada partida
puntajeFinal	Muestra el puntaje final del jugador sumando su puntaje de cada partida.

Fuente: Elaboración propia, 2023.

DISEÑO

Diagrama de flujo No.1: Código Dados



Fuente: Elaboración propia 2023.

Diagrama de clases No.1: Código dados

Clase	Random
Atributos	
Int dado1	
Int dado2	
Int dado3	
Int dado4	
Métodos	
random.Next(1,7)	

Fuente: Elaboración propia, 2023.

CONCLUSIONES

1. La programación nos permite crear simulaciones que cumplen con las características solicitadas, logrando simular cualquier caso de la vida real.

2. El método random.Next() es una herramienta útil ya que nos brinda valores aleatorios en un rango establecido, facilitando la creación de códigos donde se requiera este tipo de datos.

3. Cuando se utilizan condiciones y ciclos para llevar a cabo tareas de forma más sencilla, se evitan los errores en el código y se optimiza la cantidad líneas utilizadas del mismo.

RECOMENDACIONES

1. Se recomienda realizar varias pruebas al código para verificar que cumple con todas las instrucciones solicitadas.

2. Se recomienda utilizar el operador lógico OR para analizar varios datos a la vez.

3. Verificar que la información solicitada al usuario sea la adecuada para realizar la simulación.

REFERENCIAS

1. Universidad Autónoma de Juárez (2023). Fundamentos de Programación. https://www.uacj.mx/CGTI/CDTE/JPM/Documents/IIT/fund_programacion/U4-1.html
   Se consultó para saber el funcionamiento de variables contadoras y acumuladoras.

2. Microsoft (2023). *Operadores y expresiones C#. https://learn.microsoft.com/es-es/dotnet/csharp/language-reference/operators/*
   Se utilizó para encontrar un operador que compara dos variables al mismo tiempo.

3. Microsoft (2023). *Random.Next Método. https://learn.microsoft.com/es-es/dotnet/api/system.random.next?view=net-7.0*
   Se consultó para saber el funcionamiento del método Random.Next.

4. Matemáticas (2023). Cálculo de probabilidades. https://www.bartolomecossio.com/MATEMATICAS/clculo_de_probabilidades.html
   Se utilizó para saber cómo calcular la probabilidad de ganar del jugador.

ANEXOS

Manual de usuario:

Para ejecutar el programa, el usuario debe ingresar la cantidad de tiros y partidas que desea jugar en el caso de los tiros debe ser una cantidad par.

Al ingresar la cantidad de tiros par y la cantidad de partidas, el método Random.Next generará los valores de cada dado

```
int dado1 = random.Next(1, 7);
int dado2 = random.Next(1, 7);
int sumaDados1 = dado1 + dado2;
int dado3 = random.Next(1, 7);
int dado4 = random.Next(1, 7);
int sumaDados2 = dado3 + dado4;
```

Se crearon cuatro dados dentro del código para simular los turnos de jugador y la casa. Luego de tener los valores de los dados se suman para analizar que tipo de tiros es, estos pueden ser pares, impares e iguales. Para llevar el conteo de cada tipo de tiro se utilizó if y variables contadoras

```
if (sumaDados1 % 2 == 0)
{
    tirospares++;
}
else
{
    tirosimpares++;
}
```

Para asignar el puntaje correspondiente se utilizó if para verificar que la suma de los dados cumplía las condiciones asignadas. Las condiciones asignadas son las siguientes:

1. Si la suma de los dados es 12 o 6 en el primer tiro, el jugador gana 6 puntos.
2. Si la suma es 4 o 10 en el primer tiro la casa gana 12 puntos.
3. Si la suma es 2,3,5,7,8 o 9 en el tiro, la suma es el punteo del jugador o la casa.
4. Un jugador pierde si la suma tira 11 antes de hacer ganado ningún punto, en este caso la casa gana 6 puntos.

Si alguna de las condiciones donde el jugador gana puntos la variable tiros ganado va aumentando su valor a 1

```
else if (sumaDados1 == 6 && M == 2)
{
    puntajeJugador += 12;
    tirosGanados++;

}
```

Para asignar al ganador del tiro se compara el puntaje del jugador con el puntaje de la casa, el mayor es el que gana la partida.

```
if (puntajeJugador >puntajeCasa)
{
    Console.WriteLine("\nEl jugador gana la partida");
    ganaParJugador++;
}
else if (puntajeJugador == puntajeCasa)
{
    Console.WriteLine("\nHay un empate, no hay ganador");

}
else if (puntajeJugador <puntajeCasa)
{
    Console.WriteLine("\nLa casa gana la partida");
    ganaParCasa++;
}
```

Para calcular la probabilidad de ganar del jugador se dividió la cantidad de tiros ganados por la cantidad de tiros totales, multiplicándolo por 100.

```
double probabilidad = (double)tirosGanados / tirosPorPartida * 100;
```

Al finalizar cada partida el código muestra al ganador de la partida, la cantidad de tiros ganados por el jugador, los tiros pares, impares e iguales. Además de la probabilidad de ganar del jugador y el puntaje del jugador.

Cuando se finalizan todas las partidas, el código muestra al ganador final. Para esta parte se utilizó if para comparar la cantidad de partidas ganadas por el jugador y la casa.

```
if (ganaParJugador>ganaParCasa)
{
    Console.WriteLine("\nGanador final: Jugador");
}
else
{
    Console.WriteLine("\nGanador final: La casa");
}
Console.WriteLine("Punteo Final del Jugador: "+ puntanjeFinalJugador);
```

Para el puntaje final del jugador se utilizó una variable acumuladora que sumaba el puntaje de cada partida.

```
puntanjeFinalJugador += puntajeJugador;
```