**Números aleatorios reales:**

Los números aleatorios reales, a diferencia de los pseudoaleatorios, son aquellos que **no siguen ningún patrón determinista y, por lo tanto, son impredecibles por naturaleza**. Se obtienen a partir de **fenómenos físicos** que involucran procesos aleatorios intrínsecos, como la desintegración radiactiva o el ruido térmico.

Esta característica los hace ideales para aplicaciones que requieren un alto nivel de seguridad e imprevisibilidad, como la **criptografía**, el **juego** o la **investigación científica**.

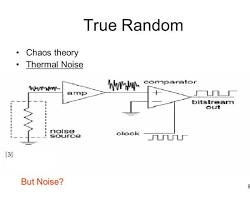
**¿En base a qué se generan?**

La generación de números aleatorios reales se basa en la **explotación de fenómenos físicos** que presentan un comportamiento inherentemente aleatorio. Algunos ejemplos de fuentes para estos números incluyen:

* **Desintegración radiactiva:**La desintegración de átomos radiactivos es un proceso aleatorio en el que no se puede predecir el momento exacto en que un átomo decaerá. Se puede medir el tiempo entre desintegraciones y utilizarlo como base para generar un número aleatorio.



**Ruido térmico:**El movimiento aleatorio de los electrones en un conductor produce ruido térmico. La amplificación y el procesamiento de este ruido pueden generar números aleatorios.



* **Eventos atmosféricos:**Ciertos eventos atmosféricos, como los rayos o el movimiento de las nubes, también presentan un comportamiento aleatorio. Se pueden utilizar sensores para detectar estos eventos y generar números aleatorios a partir de sus características.

**Números Pseudo aleatorios:**

**método del cuadrado medio** o **método de los cuadrados centrales(middle-square method**

):Fue desarrollado por John Von Neumann en la década de 1940.

**¿Cómo funciona?**

1. **Selección de la semilla:**
   * Se elige un número inicial de **2n** dígitos, donde **n** es el número deseado de dígitos en el número aleatorio final.
   * Este número inicial se denomina **semilla**.
   * Es importante que la semilla sea un número aleatorio o al menos un número con un alto grado de imprevisibilidad.
   * La elección de una mala semilla puede resultar en números aleatorios predecibles y no uniformes.
2. **Elevación al cuadrado de la semilla:**
   * Se eleva al cuadrado la semilla, obteniendo un número de **4n** dígitos.
   * Si el resultado tiene menos de 4n dígitos, se le deben agregar ceros a la izquierda hasta completar la longitud deseada.
3. **Extracción del número aleatorio:**
   * Se extrae un subconjunto de **n** dígitos del centro del resultado del cuadrado.
   * Los dígitos que se extraen son aquellos que se encuentran en las posiciones **n/2** a **n/2 + n - 1**, inclusive.
   * Este número central se convierte en el primer número aleatorio generado.
4. **Repetición del proceso:**
   * Se utiliza el número aleatorio generado en el paso 3 como nueva semilla y se repiten los pasos 2 y 3 para generar nuevos números aleatorios.
   * Cada vez que se repite el proceso, se genera un nuevo número aleatorio.

El **método congruencial lineal (LCG)** es uno de los métodos pseudoaleatorios más utilizados para generar números aleatorios. Fue introducido por Enrico Fermi en la década de 1940 y ha sido ampliamente utilizado en diversas aplicaciones, incluyendo simulaciones, juegos, criptografía y pruebas estadísticas.

**¿Cómo funciona el LCG?**

El LCG se basa en una fórmula matemática simple que combina tres valores para generar un nuevo número aleatorio:

1. **Semilla:** Un número inicial que se utiliza como punto de partida para el cálculo.
2. **Módulo:** Un número entero positivo que determina el rango de los números aleatorios generados.
3. **Multiplicador:** Un número entero constante que se utiliza para multiplicar la semilla.
4. **Incremento:** Un número entero constante que se agrega al resultado de la multiplicación.

La fórmula para generar un nuevo número aleatorio (Xn+1) a partir de un número aleatorio anterior (Xn) es la siguiente:

**Xn+1 = (a \* Xn) + c (mod m)**

Donde:

* **Xn+1** es el nuevo número aleatorio generado.
* **Xn** es el número aleatorio anterior.
* **a** es el multiplicador.
* **c** es el incremento.
* **m** es el módulo.

El **generador de números aleatorios Mersenne Twister (MT19937)** es un algoritmo pseudoaleatorio que utiliza un método complejo para generar secuencias de números enteros de 32 bits con un alto grado de aleatoriedad y uniformidad.A diferencia de otros métodos como el LCG, el MT19937 tiene un período muy largo, lo que significa que la secuencia de números aleatorios generados no se repite durante un ciclo muy extenso. Esto lo hace ideal para aplicaciones que requieren números aleatorios verdaderamente impredecibles.

**Funcionamiento:**

1. **Estado interno:** El MT19937 mantiene un estado interno compuesto por un arreglo de **624 números enteros de 32 bits** (llamado **mt**) y un índice (llamado **mti**). Este estado representa la información necesaria para generar la siguiente secuencia de números aleatorios.
2. **Algoritmo de generación:** Para generar un nuevo número aleatorio, el algoritmo realiza los siguientes pasos:

a. **Extracción:** Se extraen dos números enteros del arreglo **mt**, utilizando las posiciones indicadas por el índice **mti**.Se denominan **x** e **y**.

b. **Mezcla:** Se realiza una operación de mezcla bit a bit entre **x** e **y**, combinando y rotando bits de manera específica.El resultado de la mezcla se almacena en una variable temporal **z**.

c. **Actualización del estado:** Se actualiza el estado interno del algoritmo:

\* Se \*\*decrementa\*\* el valor de \*\*mti\*\* en 1. Si \*\*mti\*\* llega a 0, se reinicia a 624.

\* Se \*\*reemplaza\*\* el elemento del arreglo \*\*mt\*\* en la posición indicada por \*\*mti\*\* con el valor de \*\*z\*\*.

d. **Número aleatorio:** El valor de **z** se **divide** por 2**31** y se **redondea** al número entero más cercano. El resultado final es el nuevo número aleatorio generado.

1. **Repetición:** Se repiten los pasos 2a a 2d para generar nuevos números aleatorios.

**Características del MT19937:**

* **Período largo:** El período del MT19937 es de aproximadamente **2**⁹⁶, lo que significa que la secuencia de números aleatorios generados no se repite durante este ciclo tan extenso.
* **Buena distribución:** Los números generados por el MT19937 se distribuyen uniformemente dentro del rango de valores posibles (0 a 4294967295).
* **Alta velocidad:** El MT19937 es un algoritmo computacionalmente eficiente que puede generar números aleatorios rápidamente.

**Limitaciones del MT19937:**

* **No es criptográficamente seguro:** Aunque el MT19937 es bueno para muchas aplicaciones, no se considera seguro para aplicaciones criptográficas. Su algoritmo puede ser vulnerable a ataques estadísticos sofisticados.
* **Dependencia de la semilla:** Al igual que otros métodos pseudoaleatorios, el MT19937 depende de un valor inicial (semilla) para generar la secuencia de números. Una mala elección de la semilla puede afectar la aleatoriedad de los números generados.

**Xorshift** es un algoritmo pseudoaleatorio para generar secuencias de números aleatorios de 64 bits. Fue desarrollado por Sebastiano Vigna en 2005 y se ha convertido en una opción popular para aplicaciones que requieren números aleatorios de alta calidad.

**¿Cómo funciona Xorshift?**

Xorshift se basa en una operación simple de XOR (operación de exclusiva disyunción) y un desplazamiento de bits. A diferencia de otros métodos como el LCG o el Mersenne Twister, Xorshift no utiliza tablas o arreglos internos, lo que lo hace más compacto y eficiente en cuanto a memoria.

El algoritmo funciona de la siguiente manera:

1. **Estado inicial:** Se define un estado inicial con dos valores de 64 bits, denominados **x** e **y**. Estos valores representan el punto de partida para la generación de números aleatorios.
2. **Generación de un número aleatorio:** Para generar un nuevo número aleatorio, se realiza la siguiente operación:

a. **XOR y desplazamiento:** Se realiza una operación **XOR** entre **x** e **y**, y luego se desplaza el resultado **12 bits**hacia la izquierda. El resultado de esta operación se almacena en una variable temporal **z**.

b. **Nuevo estado:** Se actualizan los valores de **x** e **y** de la siguiente manera:

\* \*\*x\*\* se reemplaza por el valor de \*\*y\*\*.

\* \*\*y\*\* se reemplaza por el valor de \*\*z\*\*.

c. **Número aleatorio:** El valor de **z** se devuelve como el nuevo número aleatorio generado.

1. **Repetición:** Se repiten los pasos 2a a 2c para generar nuevos números aleatorios.

**Características de Xorshift:**

* **Sencillez:** El algoritmo Xorshift es muy simple y fácil de implementar. No requiere operaciones matemáticas complejas ni tablas internas.
* **Eficiencia:** Xorshift es un algoritmo computacionalmente eficiente que puede generar números aleatorios rápidamente.
* **Período largo:** El período del Xorshift es de aproximadamente **2**¹²⁸, lo que significa que la secuencia de números aleatorios generados no se repite durante este ciclo tan extenso.
* **Buena distribución:** Los números generados por Xorshift se distribuyen uniformemente dentro del rango de valores posibles (0 a 2\*\*⁶⁴ - 1).

**Limitaciones de Xorshift:**

* **No es criptográficamente seguro:** Aunque Xorshift es bueno para muchas aplicaciones, no se considera seguro para aplicaciones criptográficas. Su algoritmo puede ser vulnerable a ataques estadísticos sofisticados.
* **Dependencia de la semilla:** Al igual que otros métodos pseudoaleatorios, Xorshift depende de dos valores iniciales (semillas) para generar la secuencia de números. Una mala elección de las semillas puede afectar la aleatoriedad de los números generados.

**Criterio paraa determinar Aleatoriedad:**

**La Prueba Chi-Cuadrado para evaluar la Aleatoriedad**

La **prueba Chi-Cuadrado** (también conocida como prueba de ji-cuadrado o prueba χ²) es una herramienta estadística utilizada para determinar si una **distribución de frecuencias observada** se ajusta a una **distribución de frecuencias esperada**. En otras palabras, se utiliza para evaluar si una muestra proviene de una población con una distribución específica.

**¿Cómo funciona la prueba Chi-Cuadrado para evaluar la aleatoriedad?**

La prueba Chi-Cuadrado se basa en la comparación de las frecuencias observadas en una muestra con las frecuencias esperadas bajo la hipótesis de que la muestra proviene de una población con una distribución uniforme. Si las frecuencias observadas difieren significativamente de las frecuencias esperadas, se puede rechazar la hipótesis de aleatoriedad.

**1. Definir las hipótesis:**

* **H₀ (hipótesis nula):** La distribución de la muestra es aleatoria (es decir, se ajusta a una distribución uniforme).
* **H₁ (hipótesis alternativa):** La distribución de la muestra no es aleatoria (es decir, no se ajusta a una distribución uniforme).

**2. Recopilar datos:**

* + Obtener una muestra de datos.
  + Registrar la frecuencia de cada categoría o valor posible en la muestra.

3. **Calcular las frecuencias esperadas:**

* + Suponiendo que la distribución es aleatoria (H₀ es verdadera), calcular las frecuencias esperadas para cada categoría o valor posible en la muestra.

1. **Calcular el estadístico Chi-Cuadrado:**
   * Utilizar la fórmula del estadístico Chi-Cuadrado para calcular la diferencia entre las frecuencias observadas y las frecuencias esperadas, ajustando por las diferencias esperadas.
2. **Determinar el valor p:**
   * Utilizar una tabla de distribución Chi-Cuadrado con el número de grados de libertad (n-1) para encontrar el valor p asociado al estadístico Chi-Cuadrado calculado.
3. **Tomar una decisión:**
   1. Si el valor p es menor que el nivel de significancia (α), se rechaza la hipótesis nula (H₀) y se concluye que la distribución de la muestra no es aleatoria.
   2. Si el valor p es mayor que el nivel de significancia (α), no se rechaza la hipótesis nula (H₀) y no se puede concluir que la distribución de la muestra no sea aleatoria.

**La Prueba de Carreras (Runs Test) para Aleatoriedad**

La prueba de carreras, también conocida como prueba de rachas o **Wald-Wolfowitz runs test**, es una prueba estadística no paramétrica utilizada para evaluar la aleatoriedad de una secuencia de datos binarios.

**¿Qué evalúa la prueba de carreras?**

Esta prueba analiza la frecuencia de **cambios** entre dos valores posibles en una secuencia de datos. Un conjunto de datos aleatorio se espera que tenga una cantidad balanceada de **carreras**, donde una carrera es una secuencia de valores iguales (ya sea 1 o 0) seguida de un cambio de valor.

**Funcionamiento de la prueba de carreras:**

1. **Definir las hipótesis:**
   * **H₀ (hipótesis nula):** La secuencia de datos es aleatoria.
   * **H₁ (hipótesis alternativa):** La secuencia de datos no es aleatoria (presenta patrones o tendencias).
2. **Contar las carreras:**
   * Identificar y contar el número total de carreras en la secuencia de datos. Una carrera inicia con un valor (1 o 0) y termina cuando aparece el otro valor.
3. **Calcular el número esperado de carreras:**
   * Se utiliza una fórmula que considera la longitud de la secuencia (n) y la probabilidad de cada valor (p para 1 y 1-p para 0).
4. **Comparar las carreras observadas y esperadas:**
   * Se calcula un estadístico basado en la diferencia entre el número de carreras observadas y el número esperado de carreras.
5. **Determinar el valor p:**
   * Se utiliza una tabla o software estadístico para encontrar el valor p asociado al estadístico calculado.
6. **Tomar una decisión:**
   * Si el valor p es menor que el nivel de significancia (α), se rechaza la hipótesis nula (H₀) y se concluye que la secuencia de datos no es aleatoria (presenta patrones o tendencias).
   * Si el valor p es mayor que el nivel de significancia (α), no se rechaza la hipótesis nula (H₀) y no se puede concluir que la secuencia no sea aleatoria.

## Prueba de Kolmogorov-Smirnov (K-S)

La prueba de Kolmogorov-Smirnov (K-S) es una prueba estadística no paramétrica que se utiliza para determinar si dos muestras provienen de la misma distribución de probabilidad. Se puede utilizar de dos maneras:

1. **Prueba de K-S de una muestra:** Esta versión comprueba si una muestra proviene de una distribución de referencia específica (distribución teórica).
2. **Prueba de K-S de dos muestras:** Esta versión comprueba si dos muestras provienen de la misma distribución (independientemente de si es conocida o desconocida).

La prueba K-S funciona comparando las **funciones de distribución empírica (ECDF)** de las dos muestras. La ECDF representa la probabilidad de que una variable sea menor o igual que un cierto valor.

**Funcionamiento de la prueba K-S:**

1. **Calcular la ECDF para cada muestra:**
   * La ECDF se calcula para cada muestra determinando la proporción de puntos de datos menores o iguales a cada valor en la muestra.
2. **Calcular la diferencia absoluta entre las ECDF:**
   * La estadística K-S es la máxima diferencia absoluta entre las dos ECDF. Esto representa la mayor discrepancia entre las distribuciones de las dos muestras.
3. **Determinar el valor p:**
   * En base a la estadística K-S y al tamaño de las muestras, se calcula un valor p. El valor p representa la probabilidad de observar una estadística tan extrema o más extrema que la calculada, suponiendo que la hipótesis nula (las muestras provienen de la misma distribución) es cierta.
4. **Interpretar los resultados:**
   * Un valor p bajo (típicamente menos de 0.05) indica que la diferencia observada entre las ECDF es estadísticamente significativa, lo que nos lleva a rechazar la hipótesis nula. Esto sugiere que las muestras probablemente provienen de diferentes distribuciones.
   * Un valor p alto significa que no se rechaza la hipótesis nula. No hay evidencia suficiente para concluir que las muestras provienen de diferentes distribuciones.

**Resultados de la Simulación**:

**middle\_square:**

Chi-Square Test: Statistic=22499671.001880012, p-value=0.0

Runs Test: Statistic=333.31788907494234, p-value=0.0

Autocorrelation Test: lag=1, autocorrelation=3.972123257358777e-06

Kolmogorov-Smirnov Test: Statistic=0.24999800000000005, p-value=0.0

**Prueba de Chi-Cuadrado**: El valor del estadístico de prueba es 22499671.001880012 y el valor p asociado es 0.0. Esto significa que la diferencia entre las frecuencias observadas y esperadas es significativamente grande, lo que indica que los números generados no siguen una distribución uniforme.

**Prueba de Runs Test**: El valor del estadístico de prueba es 333.31788907494234 y el valor p asociado es 0.0. Esto indica que hay una falta significativa de aleatoriedad en la secuencia de números generados.

**Prueba de autocorrelación**: El valor de la autocorrelación para un retraso de 1 es de 3.972123257358777e-06. Esto significa que no hay una correlación significativa entre los números generados y sus valores anteriores.

**Prueba de Kolmogorov-Smirnov**: El valor del estadístico de prueba es 0.24999800000000005 y el valor p asociado es 0.0. Esto indica que hay una discrepancia significativa entre la distribución empírica de los números generados y la distribución uniforme teórica.

Los resultados de las pruebas indican que el generador "middle\_square" no genera números pseudoaleatorios que se comporten como se esperaría en términos de aleatoriedad y distribución uniforme

**LCG:**

Chi-Square Test: Statistic=98.93000400000003, p-value=0.5114587110393721

Runs Test: Statistic=-0.9490009480028384, p-value=0.342620129922921

Autocorrelation Test: lag=1, autocorrelation=0.0006363724407508575

Kolmogorov-Smirnov Test: Statistic=0.0011441068840027935, p-value=0.14573515867175302.

**Prueba de Chi-Cuadrado**: El valor del estadístico de prueba es 98.93000400000003 y el valor p asociado es 0.5114587110393721. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que los números generados siguen una distribución uniforme. Esto indica que los números generados por el LCG parecen seguir una distribución uniforme según la prueba de Chi-Cuadrado.

**Prueba de Runs Test**: El valor del estadístico de prueba es -0.9490009480028384 y el valor p asociado es 0.342620129922921. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que la secuencia de números generados es aleatoria. Esto indica que la secuencia generada por el LCG parece tener propiedades de aleatoriedad según la prueba de rachas.

**Prueba de autocorrelación**: El valor de la autocorrelación para un retraso de 1 es de 0.0006363724407508575. Un valor cercano a cero sugiere que no hay una correlación significativa entre los números generados y sus valores anteriores. Esto indica que los números generados por el LCG parecen ser independientes entre sí según la prueba de autocorrelación.

**Prueba de Kolmogorov-Smirnov**: El valor del estadístico de prueba es 0.0011441068840027935 y el valor p asociado es 0.14573515867175302. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que la secuencia de números generados sigue una distribución uniforme. Esto indica que los números generados por el LCG parecen seguir una distribución uniforme según la prueba de Kolmogorov-Smirnov.

los resultados sugieren que los números generados por el LCG pasan las pruebas de aleatoriedad y parecen seguir una distribución uniforme.

**mersenne\_twister**:

Chi-Square Test: Statistic=93.2536, p-value=0.6439083473562558

Runs Test: Statistic=0.21300021400064764, p-value=0.8313267957644119

Autocorrelation Test: lag=1, autocorrelation=0.001840070946965809

Kolmogorov-Smirnov Test: Statistic=0.0008185839492792779, p-value=0.513958451466643

**Prueba de Chi-Cuadrado**: El valor del estadístico de prueba es 93.2536 y el valor p asociado es 0.6439083473562558. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que los números generados siguen una distribución uniforme. Esto indica que los números generados por Mersenne Twister parecen seguir una distribución uniforme según la prueba de Chi-Cuadrado.

**Prueba de Runs Test**: El valor del estadístico de prueba es 0.21300021400064764 y el valor p asociado es 0.8313267957644119. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que la secuencia de números generados es aleatoria. Esto indica que la secuencia generada por Mersenne Twister parece tener propiedades de aleatoriedad según la prueba de rachas.

**Prueba de autocorrelación**: El valor de la autocorrelación para un retraso de 1 es de 0.001840070946965809. Un valor cercano a cero sugiere que no hay una correlación significativa entre los números generados y sus valores anteriores. Esto indica que los números generados por Mersenne Twister parecen ser independientes entre sí según la prueba de autocorrelación.

**Prueba de Kolmogorov-Smirnov**: El valor del estadístico de prueba es 0.0008185839492792779 y el valor p asociado es 0.513958451466643. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que la secuencia de números generados sigue una distribución uniforme. Esto indica que los números generados por Mersenne Twister parecen seguir una distribución uniforme según la prueba de Kolmogorov-Smirnov.

Los resultados sugieren que los números generados por Mersenne Twister pasan las pruebas de aleatoriedad y parecen seguir una distribución uniforme.

**xorshift:**

Chi-Square Test: Statistic=90.1628, p-value=0.7257451505305985

Runs Test: Statistic=0.9770009780029397, p-value=0.32856866818738917

Autocorrelation Test: lag=1, autocorrelation=-0.000762945577263051

Kolmogorov-Smirnov Test: Statistic=0.0011589122546397101, p-value=0.13613658627762626.

**Prueba de Chi-Cuadrado**: El valor del estadístico de prueba es 90.1628 y el valor p asociado es 0.7257451505305985. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que los números generados siguen una distribución uniforme. Esto indica que los números generados por xorshift parecen seguir una distribución uniforme según la prueba de Chi-Cuadrado.

**Prueba de Runs Test**: El valor del estadístico de prueba es 0.9770009780029397 y el valor p asociado es 0.32856866818738917. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que la secuencia de números generados es aleatoria. Esto indica que la secuencia generada por xorshift parece tener propiedades de aleatoriedad según la prueba de rachas.

**Prueba de autocorrelación**: El valor de la autocorrelación para un retraso de 1 es de -0.000762945577263051. Un valor cercano a cero sugiere que no hay una correlación significativa entre los números generados y sus valores anteriores. Esto indica que los números generados por xorshift parecen ser independientes entre sí según la prueba de autocorrelación.

**Prueba de Kolmogorov-Smirnov**: El valor del estadístico de prueba es 0.0011589122546397101 y el valor p asociado es 0.13613658627762626. Un valor p alto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que la secuencia de números generados sigue una distribución uniforme. Esto indica que los números generados por xorshift parecen seguir una distribución uniforme según la prueba de Kolmogorov-Smirnov.

Los resultados sugieren que los números generados por xorshift pasan las pruebas de aleatoriedad y parecen seguir una distribución uniforme.

### **Semilla Variable(Time)**

La función **time.time()** devuelve el tiempo actual en segundos desde el "epoch" (01 de enero de 1970 00:00:00 UTC). Al convertir este valor en un entero (**int(time.time())**), obtenemos un número que cambia constantemente con el tiempo. Esto significa que cada vez que se ejecuta el programa, se genera una semilla diferente, garantizando que las secuencias de números pseudoaleatorios generadas sean distintas en cada ejecución.

Bibliografia:

Links: <https://www.lawebdefisica.com/apuntsmat/num_aleatorios/>

Links: <https://www.icfo.eu/es/noticias/2187/generadores-de-numeros-aleatorios-cuanticos-super-rapidos-permiten-una/>

Links: <https://hackaday.io/project/4628-nuclear-random-number-generator>

Links: <https://www.monografias.com/trabajos107/generacion-numeros-seudo-aleatorios-ii/generacion-numeros-seudo-aleatorios-ii#google_vignette>