1.

```
[1]: # Generador Congruencial
a = int(input("Digite el valor de a: "))
c = int(input("Digite el valor de c: "))
x0 = int(input("Digite el valor de x0: "))
m = int(input("Digite el valor de m: "))
lista = []
x1 = (a * x0 + c) % m
lista.append(x0)
lista.append(x1)

while True:
  x1 = (a * x1 + c) % m
  if x1 in lista[:-1]:
    break
  else:
    lista.append(x1)
cont = len(lista)
print(lista)
print(cont)
```

```
Digite el valor de a:  21
Digite el valor de c:  15
Digite el valor de x0:  21
Digite el valor de m:  31
[21, 22, 12, 19, 11, 29, 4, 6, 17, 0, 15, 20, 1, 5, 27, 24, 23, 2, 26, 3, 16, 10, 8, 28, 14, 30, 25, 13, 9, 18]
30
```

```
Digite el valor de a:  13
Digite el valor de c:  9
Digite el valor de x0:  7
Digite el valor de m:  128
[7, 100, 29, 2, 35, 80, 25, 78, 127, 124, 85, 90, 27, 104, 81, 38, 119, 20, 13, 50, 19, 0, 9, 126, 111, 44, 69, 10, 11, 24, 65, 86, 103, 68,
125, 98, 3, 48, 121, 46, 95, 92, 53, 58, 123, 72, 49, 6, 87, 116, 109, 18, 115, 96, 105, 94, 79, 12, 37, 106, 107, 120, 33, 54, 71, 36, 93, 6
6, 99, 16, 89, 14, 63, 60, 21, 26, 91, 40, 17, 102, 55, 84, 77, 114, 83, 64, 73, 62, 47, 108, 5, 74, 75, 88, 1, 22, 39, 4, 61, 34, 67, 112, 5
7, 110, 31, 28, 117, 122, 59, 8, 113, 70, 23, 52, 45, 82, 51, 32, 41, 30, 15, 76, 101, 42, 43, 56, 97, 118]
128
```

```
Digite el valor de a:  17
Digite el valor de c:  0
Digite el valor de x0:  23
Digite el valor de m:  31
[23, 19, 13, 4, 6, 9, 29, 28, 11, 1, 17, 10, 15, 7, 26, 8, 12, 18, 27, 25, 22, 2, 3, 20, 30, 14, 21, 16, 24, 5]
30
```

```
Digite el valor de a:  1
Digite el valor de c:  121
Digite el valor de x0:  17
Digite el valor de m:  256
[17, 138, 3, 124, 245, 110, 231, 96, 217, 82, 203, 68, 189, 54, 175, 40, 161, 26, 147, 12, 133, 254, 119, 240, 105, 226, 91, 212, 77, 198, 6
3, 184, 49, 170, 35, 156, 21, 142, 7, 128, 249, 114, 235, 100, 221, 86, 207, 72, 193, 58, 179, 44, 165, 30, 151, 16, 137, 2, 123, 244, 109, 2
30, 95, 216, 81, 202, 67, 188, 53, 174, 39, 160, 25, 146, 11, 132, 253, 118, 239, 104, 225, 90, 211, 76, 197, 62, 183, 48, 169, 34, 155, 20,
141, 6, 127, 248, 113, 234, 99, 220, 85, 206, 71, 192, 57, 178, 43, 164, 29, 150, 15, 136, 1, 122, 243, 108, 229, 94, 215, 80, 201, 66, 187,
52, 173, 38, 159, 24, 145, 10, 131, 252, 117, 238, 103, 224, 89, 210, 75, 196, 61, 182, 47, 168, 33, 154, 19, 140, 5, 126, 247, 112, 233, 98,
219, 84, 205, 70, 191, 56, 177, 42, 163, 28, 149, 14, 135, 0, 121, 242, 107, 228, 93, 214, 79, 200, 65, 186, 51, 172, 37, 158, 23, 144, 9, 13
0, 251, 116, 237, 102, 223, 88, 209, 74, 195, 60, 181, 46, 167, 32, 153, 18, 139, 4, 125, 246, 111, 232, 97, 218, 83, 204, 69, 190, 55, 176,
41, 162, 27, 148, 13, 134, 255, 120, 241, 106, 227, 92, 213, 78, 199, 64, 185, 50, 171, 36, 157, 22, 143, 8, 129, 250, 115, 236, 101, 222, 8
7, 208, 73, 194, 59, 180, 45, 166, 31, 152]
256
```

```
Digite el valor de a:  15
Digite el valor de c:  903
Digite el valor de x0:  21
Digite el valor de m:  64
[21, 2, 37, 50, 53, 34, 5, 18]
8
```

2. **Prueba χ^2**:

Queremos comprobar:

H_0: No hay diferencia entre la distribución de la muestra y la distribución teórica.

Se basa en el agrupamiento de los datos muestrales en clases dentro del intervalo $(0,1)$. Usamos el estadístico:

$$\chi_c^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

Donde:

O_i: Número observado en clase i

E_i: Número esperado en clase i

n: Número de clases

$$E_i = \frac{T}{n}$$

T: Número total de observaciones o datos

Si $\chi_c^2 \leq \chi_{n-1}^2$ con $n-1$ grados de libertad y a un nivel de confianza $1-\alpha$, entonces no se puede rechazar H_0.

3.

```
Ingrese el dato 21: 0.4557
Ingrese el dato 22: 0.1592
Ingrese el dato 23: 0.8536
Ingrese el dato 24: 0.8846
Ingrese el dato 25: 0.3410
Ingrese el dato 26: 0.1492
Ingrese el dato 27: 0.8681
Ingrese el dato 28: 0.5291
Ingrese el dato 29: 0.3188
Ingrese el dato 30: 0.5992
Ingrese el dato 31: 0.9170
Ingrese el dato 32: 0.2204
Ingrese el dato 33: 0.5991
Ingrese el dato 34: 0.5461
Ingrese el dato 35: 0.5739
Ingrese el dato 36: 0.3254
Ingrese el dato 37: 0.0856
Ingrese el dato 38: 0.2258
Ingrese el dato 39: 0.4603
Ingrese el dato 40: 0.5027
Ingrese el dato 41: 0.8376
Ingrese el dato 42: 0.6235
Ingrese el dato 43: 0.3681
Ingrese el dato 44: 0.2088
Ingrese el dato 45: 0.1525
Ingrese el dato 46: 0.2006
Ingrese el dato 47: 0.4720
Ingrese el dato 48: 0.4272
Ingrese el dato 49: 0.6360
Ingrese el dato 50: 0.0954
Frecuencias observadas: [6, 12, 10, 6, 11]
Frecuencias esperadas: [10.0, 10.0, 10.0, 10.0, 10.0]
Chi-cuadrado observado: 3.7
Chi-cuadrado teórico (con 95% de confianza y 4 grados de libertad): 9.487729036781154
No se rechaza la hipótesis nula: los datos siguen una distribución uniforme.
```

4.

Frecuencias observadas: [19, 21, 21, 17, 22]

Frecuencias esperadas: [20.0, 20.0, 20.0, 20.0, 20.0]

Chi-cuadrado observado: 1.6

Chi-cuadrado teórico (con 95% de confianza y 4 grados de libertad): 9.487729036781154

No se rechaza la hipótesis nula: los datos siguen una distribución uniforme.

5.

```
# Prueba Ji Cuadrada
import numpy as np
from scipy.stats import chi2

def generar_clases():
    clases = []
    limite_inferior = 0
    limite_superior = 0.19
    for _ in range(5):
        clases.append((limite_inferior, limite_superior))
        limite_inferior += 0.2
        limite_superior += 0.2
    return clases

def calcular_frecuencias(datos, clases):
    frecuencias = [0] * len(clases)
    for dato in datos:
        for i, (lim_inf, lim_sup) in enumerate(clases):
            if lim_inf <= dato <= lim_sup:
                frecuencias[i] += 1
                break
    return frecuencias

def prueba_chi_cuadrado(frecuencias_obs, frecuencias_esp):
    chi_cuadrado = sum((obs - esp) ** 2 / esp for obs, esp in
zip(frecuencias_obs, frecuencias_esp))
    return chi_cuadrado

def main():
    n = int(input("Ingrese la cantidad de datos que desea ingresar:
"))
    datos = []
    for i in range(n):
        dato = float(input(f"Ingrese el dato {i+1}: "))
        datos.append(dato)

    clases = generar_clases()
    frecuencias_esp = [n/5] * 5
    frecuencias_obs = calcular_frecuencias(datos, clases)
    chi_cuadrado_obs = prueba_chi_cuadrado(frecuencias_obs,
frecuencias_esp)
    grados_libertad = len(clases) - 1
```

```
    chi_cuadrado_teorico = chi2.ppf(0.95, grados_libertad)

    print("Frecuencias observadas:", frecuencias_obs)
    print("Frecuencias esperadas:", frecuencias_esp)
    print("Chi-cuadrado observado:", chi_cuadrado_obs)
    print("Chi-cuadrado teórico (con 95% de confianza y",
grados_libertad, "grados de libertad):", chi_cuadrado_teorico)
    if chi_cuadrado_obs < chi_cuadrado_teorico:
        print("No se rechaza la hipótesis nula: los datos siguen
una distribución uniforme.")
    else:
        print("Se rechaza la hipótesis nula: los datos no siguen
una distribución uniforme.")

if __name__ == "__main__":
    main()
```

6.

```
# Generador Congruencial
a = int(input("Digite el valor de a: "))
c = int(input("Digite el valor de c: "))
x0 = int(input("Digite el valor de x0: "))
m = int(input("Digite el valor de m: "))
lista = []
x1 = (a * x0 + c) % m
lista.append(x0)
lista.append(x1)

while True:
  x1 = (a * x1 + c) % m
  if x1 in lista[:-1]:
    break
  else:
    lista.append(x1)
cont = len(lista)
print(lista)
print(cont)
```

```
Digite el valor de a: 4
Digite el valor de c: 9
Digite el valor de x0: 15
Digite el valor de m: 98
[15, 69, 89, 71, 97, 5, 29, 27, 19, 85, 55, 33, 43, 83, 47, 1, 13, 61, 57, 41, 75]
21
```

```
Ingrese la cantidad de datos que desea ingresar: 21
Ingrese el dato 1: 0.15
Ingrese el dato 2: 0.69
Ingrese el dato 3: 0.89
Ingrese el dato 4: 0.71
Ingrese el dato 5: 0.97
Ingrese el dato 6: 0.05
Ingrese el dato 7: 0.29
Ingrese el dato 8: 0.27
Ingrese el dato 9: 0.19
Ingrese el dato 10: 0.85
Ingrese el dato 11: 0.55
Ingrese el dato 12: 0.33
Ingrese el dato 13: 0.43
Ingrese el dato 14: 0.83
Ingrese el dato 15: 0.47
Ingrese el dato 16: 0.01
Ingrese el dato 17: 0.13
Ingrese el dato 18: 0.61
Ingrese el dato 19: 0.57
Ingrese el dato 20: 0.41
Ingrese el dato 21: 0.75
Frecuencias observadas: [5, 3, 5, 4, 4]
Frecuencias esperadas: [4.2, 4.2, 4.2, 4.2, 4.2]
Chi-cuadrado observado: 0.6666666666666665
Chi-cuadrado teórico (con 95% de confianza y 4 grados de libertad): 9.487729036781154
No se rechaza la hipótesis nula: los datos siguen una distribución uniforme.
```