Daniel Cu Sánchez
A01703613

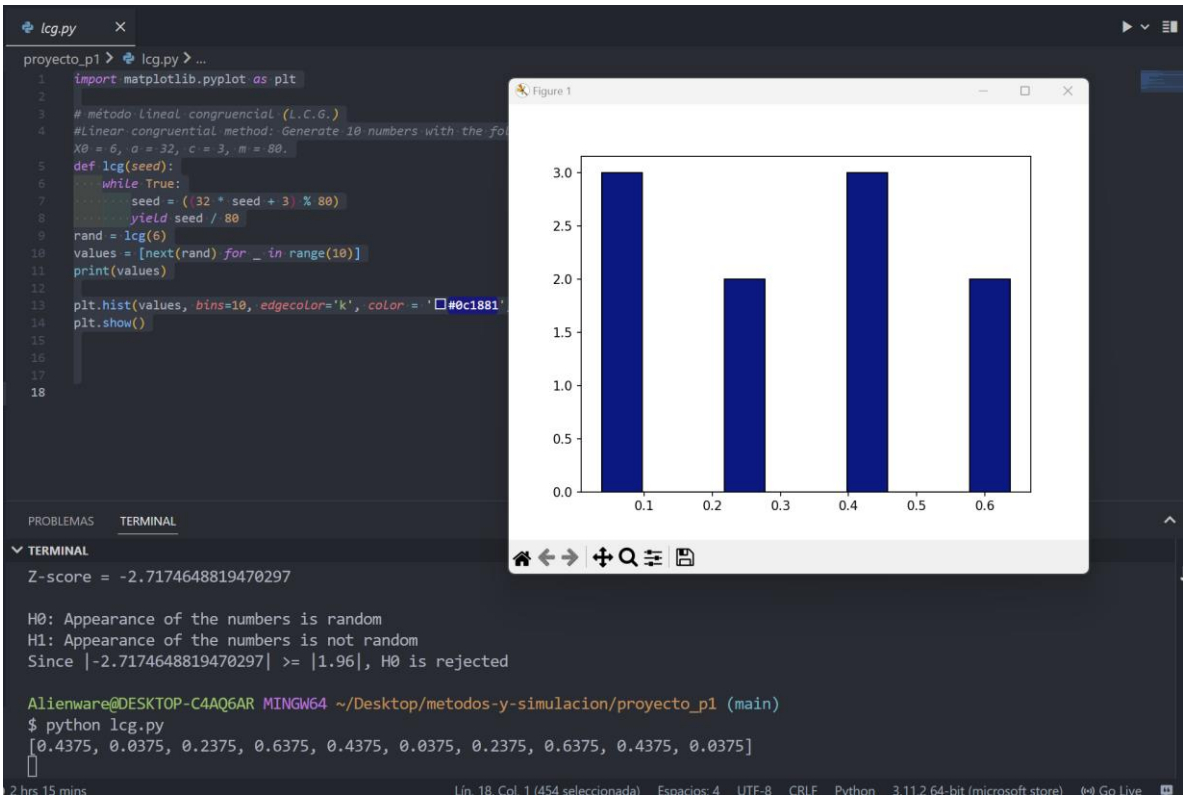
Proyecto parcial 1

Método Lineal Congruencial

Código

```python
import matplotlib.pyplot as plt

# método lineal congruencial (L.C.G.)
#Linear congruential method: Generate 10 numbers with the following parameters: X0 = 6, a = 32, c = 3, m = 80.
def lcg(seed):
    while True:
        seed = ((32 * seed + 3) % 80)
        yield seed / 80
rand = lcg(6)
values = [next(rand) for _ in range(10)]
print(values)

plt.hist(values, bins=10, edgecolor='k', color = '#0c1881', histtype = 'bar')
plt.show()
```

Prueba



Daniel Cu Sánchez
A01703613

Chi-squared test

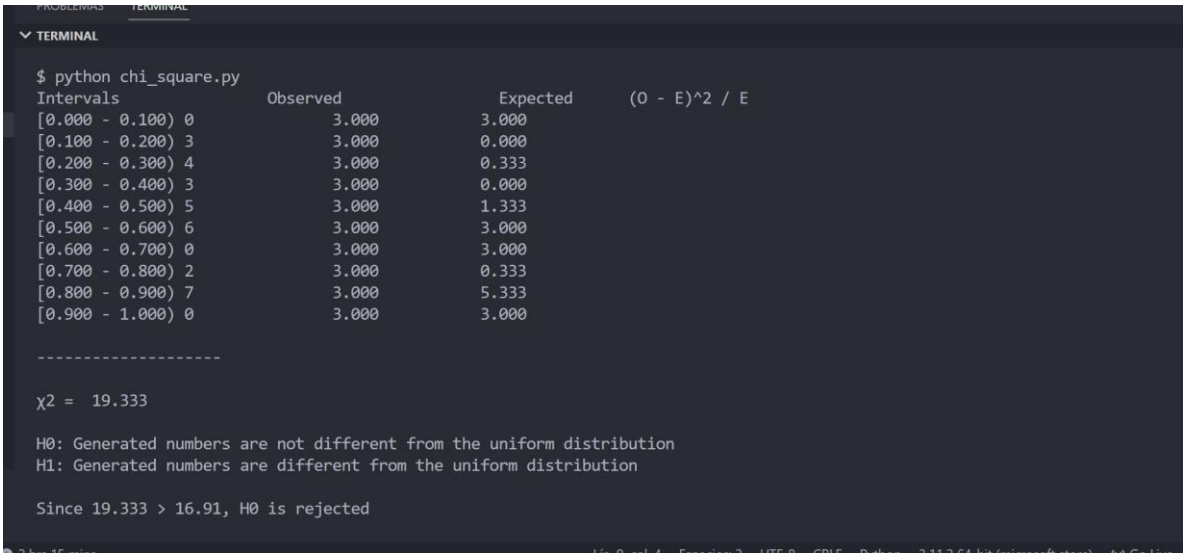
Código

```
import numpy as np
from scipy.stats import chisquare
# prueba de aleatoriedad (Prueba de Chi-cuadrada)
"""
La tabla de frecuencias (con C = 10 y W = 0.1), mostrando los intervalos, frecuencias
observadas y esperadas (no es obligatorio mostrar las operaciones, aunque yo las muestre
en mi ejemplo)
el valor de χ2
las hipótesis H0 and H1
la conclusión acerca de H0 (es decir, si fue rechazada o no y por qué)
"""
# Run the chi-squared test with the following numbers: chi_data.txt

# import the observed values
values = np.loadtxt("chi_data.txt", dtype=float)
# round the values to 3 decimals
observed = np.around(values, 3)
# define the expected values
expected = np.repeat(3.0, 10)
# define the intervals
intervals = np.linspace(0, 1, 11)
# calculate the observed frequencies
observedfrequencies, _ = np.histogram(observed, intervals)
# we can not calculate the expected

# calculate the chi-square statistic and p-value
chisquared, pvalue = chisquare(observedfrequencies, expected)
# calculate the chi-square contributions for each interval
chisquaredcontributions = ((observedfrequencies - expected)**2) / expected

# print the table
print("Intervals  \t\t Observed  \t\t  Expected\t(O - E)^2 / E")
for i in range(len(intervals) - 1):
  print("[{:.3f} - {:.3f})\t{}\t\t{:.3f}\t\t{:.3f}"
    .format(
    intervals[i],
    intervals[i+1],
    observedfrequencies[i],
    expected[i],
    chisquaredcontributions[i]
    )
  )

print()
print("--------------------")
print()
print("χ2 = {: 0.3f}".format(chisquared))
print()

print("H0: Generated numbers are not different from the uniform distribution")
```

Daniel Cu Sánchez
A01703613

```
print("H1: Generated numbers are different from the uniform distribution")

#Conclusion
if chisquared > 16.91:
  print()
  print("Since {:0.3f} > 16.91, H0 is rejected".format(chisquared))
else:
  print()
  print("Since {:0.3f} <= 16.91, H0 is not rejected".format(chisquared))
```

Prueba



Daniel Cu Sánchez
A01703613

Runs test

Código

```python
from scipy.stats import norm
import numpy as np

# prueba de aleatoriedad (Prueba de Rachas)
"""
Los signos generados (en caso de que sean muchos, el usuario decidirá si se muestran
todos o no)
La cantidad de Rachas calculadas
Los parámetros μ, σ y Zr
Las hipótesis H0 and H1
La conclusión acerca de H0 (es decir, si fue rechazada o no y por qué)
"""

numbers = np.loadtxt("runs_data.txt", dtype=float)

# Generate the signs
signs = np.sign(numbers - np.mean(numbers))
signs[signs == 0] = 1

# Calculate the total number of signs and runs
total_signs = len(signs)
total_runs = len(np.where(np.diff(signs) != 0)[0]) + 1

# Calculate the mean and standard deviation of the number of runs
miu = (2 * total_signs - 1) / 3
sigma = np.sqrt((16 * total_signs - 29) / 90)

# Calculate the z-score and p-value
z_score = (total_runs - miu) / sigma
p_value = 2 * (1 - norm.cdf(abs(z_score)))

# Print the results
print("Generated signs:\n", " ".join(["+" if s == 1 else "-" for s in signs]))
print()
print("Total signs:", total_signs)
print("Total runs:", total_runs)
print()
print("Statistics")
print("Miu =", miu)
print("Sigma =", sigma)
print("Z-score =", z_score)
print()
print("H0: Appearance of the numbers is random")
print("H1: Appearance of the numbers is not random")
if abs(z_score) < 1.96:
    print("Since |{}| < |1.96|, H0 is not rejected".format(z_score))
else:
    print("Since |{}| >= |1.96|, H0 is rejected".format(z_score))
```

Daniel Cu Sánchez
A01703613

Prueba

```
Alienware@DESKTOP-C4AQ6AR MINGW64 ~/Desktop/metodos-y-simulacion/proyecto_p1 (main)
$ python runs.py
Generated signs:
 - - - - + - - - - + + + + -

Total signs: 14
Total runs: 5

Statistics
Miu = 9.0
Sigma = 1.4719601443879744
Z-score = -2.7174648819470297

H0: Appearance of the numbers is random
H1: Appearance of the numbers is not random
Since |-2.7174648819470297| >= |1.96|, H0 is rejected
```

Video de evidencia en drive

https://drive.google.com/drive/folders/1FFy0UHJYJkEQ7ufW2nmqqS4_eqIfDysE?usp=sharing