Problem: Using the Monte Carlo method, calculate the mathematical constant e. Do not compute $(1 + 1/n)^n$, calculate the series $\sum 1/n!$, use the logarithm function, etc...

Solution 1: Here's a brilliant and well-established solution. Just repeatedly sample from a uniform random distribution from 0 to 1 and add the sampled numbers. Stop when the sum exceeds 1. The mean number of samples for this process is e. This works because the probability that $x_1 + x_2 + ... + x_n <= 1$ is the volume of a simplex $1/n!$, so when you compute the expectation value of the number of samples, this is equivalent to the infinite series expression for e.

Solution 2: Here's a solution that I came up with that I think is more intuitive, but I don't know how sound all the logic is. Imagine we have a bunch of radioactive particles. When a radioactive particle decays, it emits a photon and somehow becomes a new radioactive particle with exactly the same properties (i.e. it can decay again with the exact same rate). This system models a Poisson process with photon emissions at a constant rate, since the population of radioactive particles does not decrease. We run a stopwatch from time $t = 0$ to $t = 1$, and suppose N photons are emitted at random times uniformly across the time interval. We assume N is large, so $N \approx \lambda$, the photon emission rate.

The time at which each photon is emitted can be modeled by sampling from a random uniform distribution N times (since the system respects time-translation symmetry). Then, if we order the sampled times, and we take the difference between consecutive times, we have $N - 1$ time differences Δt_i. Since this is a Poisson process, these time differences must follow the exponential distribution with mean $1/\lambda$. The cumulative distribution function is $1 - e^{-\lambda \Delta t}$, so if there are R time differences that are smaller than the mean, $R/N = 1 - e^{-1}$, or $e = 1/(1 - R/N)$. I've approximated $N - 1$ as N.

We also get $\ln(2)$ for free because the median of the time differences is $\ln(2)/\lambda$.

Here is a Python script for this simulation:

```python
import numpy as np

def simulate_photon_emissions(N: int) -> tuple[np.ndarray, float, float]:
    """Given N photon emissions, return the time differences
    between consecutive photon emissions, an estimate for e,
    and an estimate for ln(2)"""
    samples = np.random.uniform(0, 1, N)
    samples.sort()
    differences = np.diff(samples)
    e = 1/(1-(differences <= 1/N).sum()/N)
    ln2 = np.median(differences) * N
    return differences, e, ln2

N = 100_000_000
differences, e, ln2 = simulate_photon_emissions(N)

print(f"""The computed e is off by a factor of {(e-np.e)/np.e},
and ln(2) is off by a factor of {(ln2-np.log(2))/np.log(2)}""")
```

We can also compare the sampled time differences against the exponential distribution just to check that they are the same:

```
import matplotlib.pyplot as plt

plt.hist(differences,bins=50, density=True, alpha=0.6, color='g',
         label='Time Difference Between Photon Emissons')
x = np.linspace(0, np.max(differences), 1000)
exp_pdf = N * np.exp(-N * x)
plt.plot(x, exp_pdf, 'r-',
         label=f'Exponential Distribution with Rate {N})')
plt.xlabel('Time Difference')
plt.ylabel('Density')
plt.legend()
plt.show()
```

As a side note, if we wanted a Monte Carlo calculation of $\ln(2)$, a simpler solution would be to exploit $\ln(2) = \int_1^2 dx/x$. One simply samples from a uniform random distribution between 1 and 2, takes the reciprocal of the numbers, and then finds the mean. One can then extend this idea to get a Monte Carlo calculation of e by using a binary search to find t such that $\int_1^t dx/x = 1$.