In [1]:
```python
import numpy as np
import pandas as pd
import scipy
```

In [2]:
```python
def simulate_poisson_process(l,t,mu):
    jumpTimes= np.append(np.zeros(1),np.cumsum(np.random.exponential(scale=1
    values= np.arange(0, t+1)
    compound_values=  np.append(np.zeros(1), np.cumsum(np.random.exponential
    return values, compound_values, jumpTimes

def find_value_at_T(values,jumptimes, T):
    idx = np.where(jumptimes<T)[-1][-1]
    return values[idx]
```

1. For the Poisson process $N = (N(t), t \geq 0)$ with rate $\lambda = 1.2$. Assume τ_k is the time of the kth jump, find:

In [3]:
```python
N = 10000
l = 1.2
mu=2.5
timesteps=1500
```

In [4]:
```python
events=[]
compound_events = []
taus = []
np.random.seed(12345)
for i in range(N):
    a,b,c = simulate_poisson_process(l,timesteps,mu)
    events.append(a)
    compound_events.append(b)
    taus.append(c)
```

In [5]:
```python
values_at_t2 = []

for i in range(N):
    values_at_t2.append(find_value_at_T(events[i], taus[i], 2))

P1= values_at_t2.count(3)/N
```

A. $\mathbb{P}(N(2) = 3)$

In [6]:
```python
print('P(N(2)=3)=', P1)
```

```
P(N(2)=3)= 0.2096
```

B. $\mathbb{P}(N(2) = 3, N(5) = 6, N(10) \geq 9)$

In [7]:
```python
P2_count = 0

values_at_t2 = []
values_at_t5 = []
values_at_t10 = []
```

```
for i in range(N):
    values_at_t2.append(find_value_at_T(events[i], taus[i], 2))
    values_at_t5.append(find_value_at_T(events[i], taus[i], 5))
    values_at_t10.append(find_value_at_T(events[i], taus[i], 10))

    if (values_at_t2[i] ==3) and (values_at_t5[i] ==6) and (values_at_t10[i]
        P2_count+=1

P2=P2_count/N
```

In [8]:
```
print('P((N(2)=3, N(5)=6, N(10)>=9)= ', P2)
```

```
P((N(2)=3, N(5)=6, N(10)>=9)=  0.0414
```

C. $\mathbb{E}[e^{2N(0.5)}]$

In [9]:
```
values_at_t_1half = []

for i in range(N):
    values_at_t_1half.append(find_value_at_T(events[i], taus[i],0.5))
```

In [10]:
```
print('E[e^2N(0.5)]=', np.mean(np.exp(2*np.array(values_at_t_1half))))
```

```
E[e^2N(0.5)]= 44.21546876964519
```

In [21]:
```
np.min(np.array(values_at_t_1half))
```

Out[21]:
```
0
```

D. $\mathbb{E}[\tau_2]$

In [11]:
```
tau2 = []

for i in range(N):
    tau2.append(taus[i][2])
```

In [12]:
```
print('E[tau_2]=', np.mean(tau2))
```

```
E[tau_2]= 1.670915291003944
```

E. $\mathbb{P}(2 < \tau_2 < 3.2)$

In [13]:
```
tau2_series= pd.Series(tau2)
print('P(2 < tau_2 < 3.2)=', len(tau2_series[tau2_series.between(2,3.2, incl
```

```
P(2 < tau_2 < 3.2)= 0.2051
```

2. For the compound Poisson process $X = (X(t), t \geq 0)$ corresponding to the Poisson process from Theory 1, with jumps Z_k having exponential distribution with mean 2.5, find:

A. $\mathbb{E}[X(2)]$

In [14]:
```
values_at_X2 = []

for i in range(N):
    values_at_X2.append(find_value_at_T(compound_events[i], taus[i], 2))
```

In [15]:
```
print('E[X(2)]=', np.mean(values_at_X2))
```

```
E[X(2)]= 5.983706190626745
```

B. $Var(X(2))$

In [16]:
```
print('Var(X(2))=', np.var(values_at_X2))
```

```
Var(X(2))= 30.143141569183378
```

D. $\mathbb{E}[e^{-3X(2)}]$

In [17]:
```
print('E[e^(-3X(2))]=',np.mean(np.exp(-3*np.array(values_at_X2))))
```

```
E[e^(-3X(2))]= 0.1235090615340705
```

3. Using the Central Limit Theorem, for the compound Poisson process from Theory 2, find the approximate value of u such that $X(1000) < u$ with probability 99\%. This u is the 99\% quantile of $X(1000)$

In [18]:
```
values_at_t1000 = []

for i in range(N):
    values_at_t1000.append(find_value_at_T(compound_events[i], taus[i], 1000
```

In [19]:
```
print('u=', np.percentile(values_at_t1000,99))
```

```
u= 3288.081320425806
```