```python
'''
      Inhomogeneous Poisson Process via Homogeneous Poisson Process
      Homework 3: Notes Method 2 (Thinning?)
      Author: Michael Schniepp
'''
import numpy as np
import matplotlib.pyplot as plt

def l(t):
      y = 2*t + 2
      return y

firstEvents = []
T = 5
L = l(T)
events = np.random.poisson(T*L)

# Note: The code inside the loop executes a single non-homogeneous p.p.
for i in range(0,1000):
      Vtimes = []
      Utimes = []
      numEvents = []
      for i in range(0,events):
            Vtimes.append(np.random.uniform(0,T))

      for i in range(0,events):
            Utimes.append(np.random.uniform(0,1))

      sUtimes = sorted(Utimes)
      nhpp = []

      for i,j in zip(sUtimes,Vtimes):
            if i <= l(j)/float(L):
                  nhpp.append(j)

      snhpp = sorted(nhpp)
      N = []
      for i in range(0,len(nhpp)):
            N.append(i)

      firstEvents.append(snhpp[0])

mu = np.mean(firstEvents)

def var(data):
      for i in range(0,len(data)):
            data[i] = (data[i] - mu)**2
      var = sum(data)/float(len(data))
      return var

# Note: The theoretical values were derived by setting up the integrals
# by hand
# and then approximating the value using my own extrapolated quadrature
# for both the first and second moments.
print 'Theoretical E[T_1]:',0.3789
print 'Simulated E[T_1]:',np.mean(firstEvents)
```

```
print 'Theoretical Var[T_1]:',0.09853
print 'Simulated Var[T_1]:',var(firstEvents)
```

EXAMPLE OUPUT:

```
Theoretical E[T_1]: 0.3789
Simulated E[T_1]: 0.370232692212
Theoretical Var[T_1]: 0.09853
Simulated Var[T_1]: 0.0966106458012
```