SOMATIVA 02

Estudante

In [1]:
```
# Douglas Lemos Bregenski Schiavi
```

Importar bibliotecas

In [2]:
```
import numpy as np
import scipy.stats as st
from filaMMm import FilaMMm
```

Questão 1

Um helpdesk trata apenas 3 categorias de solicitações: (i) problemas de login; (ii) problemas de hardware; (iii) problemas configuração. As solicitações chegam segundo processos de Poisson com as seguintes taxas: L_L = 0.5 solicitações de problemas de login por minuto; L_H = 2.5 solicitações de problemas de hardware por minuto; e L_C = 1.5 solicitações de problemas de configuração por minuto. Qual a probabilidade que em 2 minutos cheguem 3 solicitações, todas de hardware? Dica: para que cheguem 3 solicitações, todas de hardware, é necessário que cheguem 0 solicitações de login, 0 solicitações de configuração, e 3 solicitações de hardware. (Valor 0,75)

In [3]:
```
# Zero solicitações de login em 2 minutos
login = st.poisson.pmf(0, 0.5*2)

# Zero solicitações de configuração em 2 minutos
config = st.poisson.pmf(0, 1.5*2)

# Três solicitações de hardware em 2 minutos
hard = st.poisson.pmf(3, 2.5*2)

# Três solicitações, todas de hardware, em 2 minutos
print(login * config * hard)
```

```
0.002571037585139156З
```

Questão 2

Um sistema de armazenagem consiste em 3 data storages que compartilham uma fila comum. O tempo médio para atender uma solicitação é 50 milissegundos. As solicitações chegam a uma taxa de 30 solicitações por segundo. Calcular:
a) Número médio de tarefas na fila. (Valor 0,35)
b) Probabilidade de o tempo na fila ser menor do que 0,01 segundos. (Valor 0,4)

Utilizar a classe FilaMMm (arquivo filaMMm.py)

In [11]:
```
# Criar fila
lb=1/30
mu=1/0.05
m=3

fila = FilaMMm(lb, mu, m)

# Número médio de tarefas na fila
numMedio = fila.E_Nq

# Probabilidade de o tempo na fila ser menor do que 0,01 segundos
prob = fila.cdf_R(0.01)

print(numMedio)
print(prob)
```

```
4.2843145889988415e-13
0.18126924713376294
```

Questão 3

O tempo de atendimento de chamados em um helpdeks é uma variável aleatória exponencial com média igual a 4 minutos. Um chamado está na fila com 2 chamados à sua frente (na fila) e mais 1 chamado sendo atendido. Qual a probabilidade de o chamado esperar mais do que 6 minutos para começar a ser atendido. (Valor 0,75)

Dicas: Você pode calcular a probabilidade de duas maneiras:
(i) Para o chamado esperar mais do que 6 minutos é preciso que o chamado que está sendo atendido e mais os dois que estão na fila à sua frente demorem mais do que seis minutos. Usar a equação da função gama para calcular $P[S_N > x]$.
(ii) Para o chamado esperar mais do que 6 minutos é preciso que no intervalo de 6 minutos ocorra uma das seguintes possibilidades: nenhum chamado terminou de ser atendido no intervalo, ou 1 terminou, ou 2 terminaram. O terceiro deve terminar depois de 6 minutos. Usar a equação do processo de poisson para calcular $P[N_s = 0] + P[N_s = 1] + P[N_s = 2]$.

Escolher uma delas para responder.

In [5]:
```
# P[SN>x]
t1 = 0
t2 = 6
S = t2-t1
lbda = 0.2
nSim = 50000
x = 3
mu = 1/lbda

deuCerto = 0
for i in range(nSim):
    tempo = 0
    nEventos = 0
    while tempo <= t2:
        if (tempo >= t1):
            nEventos = nEventos + 1
        # sorteia variável exponencial e acumula em tempo
        tempo = tempo + st.expon.rvs(0, mu)
    # se quantidade de eventos = x, deuCerto
    if nEventos == x:
        deuCerto = deuCerto + 1
probS = 1 - (deuCerto/nSim)

print(probS)
```

```
0.7820199999999999
```

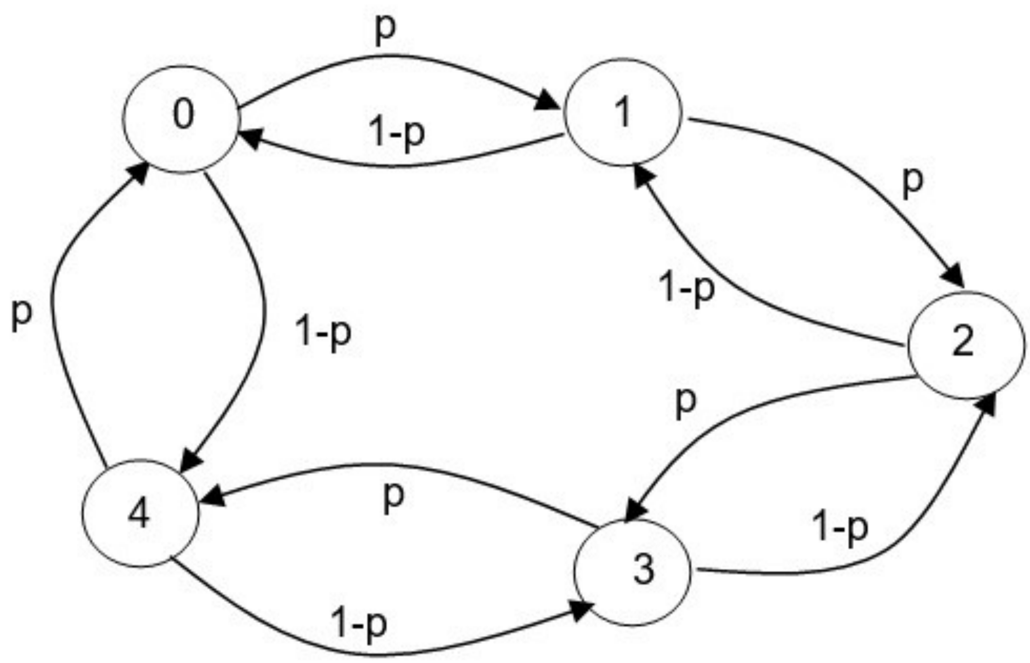
In [6]:
```
# P[Ns=0] + P[Ns=1] + P[Ns=2]
```

Questão 4

Uma partícula move-se em um círculo através de pontos que foram marcados como 0, 1, 2, 3, 4, na direção do ponteiro do relógio. A cada passo existe uma probabilidade p de mover na direção do ponteiro de relógio e $1-p$ de mover na outra direção. Seja X_n a localização da partícula no círculo após o n-ésimo passo. O processo {$X_n, n \geq n_0$} é a cadeia de Markov representada pelo diagrama a seguir.

a) Calcular a matriz de transições de um passo para p=0.7. (Valor 0,4)

b) Utilize a função **cmtdP** (desenvolvida na formativa 11) para calcular as probabilidades do regime permanente (p=0.7). (Valor 0,35)



In [7]:
```
# Função cmtdP
def cmtdP(P):
    [r,c] = P.shape
    if ((r != c) | np.all(np.sum(P, 1) != 1)):
        raise Exception('Matriz P invalida!')

    # Colocar seu código aqui
    A = np.transpose(P) - np.identity(r)
    A = np.vstack((A, np.ones(r)))
    B = np.zeros(r)
    B = np.hstack((B,[1]))
    A_pinv = np.linalg.pinv(A)
    PI = np.dot(A_pinv, B)
    return PI
```

In [8]:
```
# Matriz de transições P
p = 0.7
P = np.array([[0,   p,   0,   0, 1-p],
              [1-p, 0,   p,   0,   0],
              [0, 1-p,   0,   p,   0],
              [0,   0, 1-p,   0,   p],
              [p,   0,   0, 1-p,   0]], dtype=np.float64)
print(P)
```

```
[[0.  0.7 0.  0.  0.3]
 [0.3 0.  0.7 0.  0. ]
 [0.  0.3 0.  0.7 0. ]
 [0.  0.  0.3 0.  0.7]
 [0.7 0.  0.  0.3 0. ]]
```

In [9]:
```
# Probabilidades do regime permanente
print(cmtdP(P))
```

```
[0.2 0.2 0.2 0.2 0.2]
```

In [ ]: