Os códigos completos serão anexados junto do docs da avaliação na entrega da atividade

Questão 1: Considere que está inscrito em um círculo um dodecágono regular cujos vértices estão numerados de 0 até 11. Suponha que uma partícula move-se ao longo dos vértices do polígono, e que a cada segundo ela dá um passo no sentido horário ou anti-horário, com igual probabilidade. Matematicamente, se Xt é a posição da partícula após passados t segundos, entao temos que

$$P(X_t = i + 1 \mid X_{t-1} = i) = P(X_t = i - 1 \mid X_{t-1} = i) = 1/2,$$

para todo i = 0, . . . , 11, considerando $i + 1 \equiv 0$ se $i = 11$ e $i - 1 \equiv 11$ se $i = 0$. Suponha que a partıcula está inicialmente no vértice 0 (ou seja, $X_0 = 0$), e movimenta-se de acordo com a regra acima, por tempo indeterminado. Com base nisso, faça o que se pede abaixo.

Item A - Seja Y a variável aleatória que representa o número de passos feitos pela partícula até que todos os vértices do polígono sejam visitados. Construa um algoritmo para simular valores segundo tal variável aleatória. Note que para isso você também precisará construir um algoritmo para simular do processo $(X_t)_{t \in \mathbb{Z}_{\geq 0}}$.

Resposta - item A

Inicialmente para satisfazer a primeira exigência do problema foi implementado uma rotina que escolhe um número aleatório entre 0 e 1, que, da forma que foi interpretado, estaria representado de forma simples uma probabilidade de 50%, rotina essa que foi implementado como

```
  value = SystemRandom().randint(0, 1)
```

A título de curiosidade, na primeira implementação do código, foi observado que a função simples randint(0, 1), não variava a sua distribuição de variáveis em cada interação que era usada , por isso foi preferível utilizar a função da biblioteca System.Random().

Partindo agora para o processo que descreveria (Xt) $t \in Z \geq 0$, ou seja, a posição da partícula em dado tempo t, inteiro e maior que zero, partícula essa que estaria percorrendo um espaço limitado [0,11],foi implementado como

```
# Criando a função que descreve a variável Y
def Y():
  #Array que armazena os passos, utilidade na plotagem de dados
  Passos=[]
  #Dado que a partícula está inicialmente no vertice 0
```

```
  passo=0
  i=0
                                                   Posicao                    =
[False,False,False,False,False,False,False,False,False,False,False,False,Fa
lse]
  # Variavel condição que vai validar se o vertice foi visitado
  condicao = all(Posicao)
  # Loop que termina quando a condição se faz verdadeira
  while (condicao==False):
  #Decide a direção no inicio do Loop
    dir = SystemRandom().randint(0, 1)
  #Loop para posição zero, caso voltar ir para posição 11
    if i==0:
      if dir==0:
        i=11
      else:
        i+=1
      Posicao[i]=True
  #Loop para posição 11, caso voltar ir para posição 0
    elif i==11:
      if dir==1:
        i=0
      else:
        i+=-1
      Posicao[i]=True
  #Loop para qualquer posição, pode ir e voltar livremente
    else:
      if dir==1:
        i+=1
      else:
        i+=-1
      Posicao[i]=True
  #Append no Array de passos
    passo+=1
    Passos.append(passo)
  #Teste se todas as posições foram visitadas
    condicao = all(Posicao)
  #Retorna o número de passos que levaram para conclusão
  return (len(Passos))
```

Item B - Através do algoritmo construído no item a), obtenha uma aproximação para o valor esperado de Y . Para isso, simule uma grande quantidade y1, . . . , yN de valores segundo Y , digamos N = 5.000, e aproxime E[Y] por $\overline{yN} = \frac{1}{N} \sum_{n=1}^{N}$ yn.

Resposta - item B

Para implementar a solução do item b) basta criar uma função que contém um ciclo que executa a função do item a) N vezes, que no caso é 5000 vezes, a cada execução ele armazena o valor para ao fim retornar um somatório dos valores e dividi-los pelo número de iterações N.

```
# Criando a função Ey, que vai armazenar o valor esperado de Y em N
iterações
def Ey(N):
  #Array que funciona com o espaço amostral de Y
  OmegaY=[]
  #Loop que executa a função Y N vezes
  for n in range(N+1):
    OmegaY.append(Y())
  return sum(OmegaY) / N
```

Item C - Justifique porque o procedimento do item b) funciona para aproximar o valor esperado de Y .

Resposta - item C

Se considerarmos que a variável aleatória Y representa o somatório das variáveis aleatórias Y no espaço amostral com N termos e dividirmos esse Yn por N, obtemos justamente a média aritmética dos valores Y, lembrando que a definição de valor esperado o outro nome também é esperança ou média, essa aproximação fica cada vez mais precisa dado um valor cada vez maior de N.

Item D - Seja Z a variável aleatória que representa a posição da partícula após transcorrido um tempo suficientemente longo, ou seja, no limite quando t → ∞. Encontre a função da massa de probabilidade de Z.

Resposta - item D

Observando o problema temos que um instante t, a partícula pode estar em qualquer uma das 12 posições do dodecágono, logo a função da massa de probabilidade de Z pode ser dada por.

$f(t) = \left\{ \frac{1}{12}, \; t \geq 0 \right\}$ e tem valor zero, caso o contrário

Questão 2: Um inseto se move aleatoriamente em um tabuleiro de tamanho 3×3 de acordo com a seguinte regra: a cada movimento ele tem igual probabilidade de saltar para alguma das casas adjacentes àquela ocupada, onde não considera-se casas na diagonal como adjacentes; porém, nas casas dos cantos superior esquerdo e inferior direito há uma armadilha que imobiliza o inseto. Com base nessa descrição, faça o que se pede abaixo.

Item A- Justifique que uma cadeia de Markov é um modelo adequado para descrever tal situação, e escreva quem é a matriz de probabilidades de transição de tal processo.

Resposta - item A

Uma cadeia de Markov pode descrever um modelo de forma mais simples e adequada visto que temos diversos estados para a movimentação do inseto cada um com suas próprias probabilidades de transição, sendo seu estado anterior indiferente já que o inseto tem probabilidade igual de se movimentar por entre as casas, excluindo somente as movimentações em diagonal.

A matriz de transição do inseto no tabuleiro pode ser escrita por

	Casa 1	**Casa 2**	**Casa 3**	**Casa 4**	**Casa 5**	**Casa 6**	**Casa 7**	**Casa 8**	**Casa 9**
Casa 1	**1**	0	0	0	0	0	0	0	0
Casa 2	**1/3**	0	**1/3**	0	**1/3**	0	0	0	0
Casa 3	0	**1/2**	0	0	0	**1/2**	0	0	0
Casa 4	**1/3**	0	0	0	**1/3**	0	**1/3**	0	0
Casa 5	0	**1/4**	0	**1/4**	0	**1/4**	0	**1/4**	0
Casa 6	0	0	**1/3**	0	**1/3**	0	0	0	**1/3**
Casa 7	0	0	0	**1/2**	0	0	0	**1/2**	0
Casa 8	0	0	0	0	**1/3**	0	**1/3**	0	**1/3**
Casa 9	0	0	0	0	0	0	0	0	**1**

Item B - Construa um algoritmo para simular a posição do inseto, onde é possível selecionar a casa do tabuleiro onde o inseto localiza-se inicialmente.

Resposta - item B

Para a construção desse algoritmo podemos fazer uma analogia dos requerimentos desse experimento com os requerimentos da questão anterior, de forma que podemos descrever a posição do inseto da mesma forma que a posição da partícula, os instantes passados são também os saltos do inseto até alcançar a captura do indivíduo suspeito, que foi o parâmetro escolhido como parada no algoritmo, já que dali ele não passa, pois pela matriz de transição as posições 1 e 9 são os sumidouros da modelagem.

```
# Criação da função Pulos que recebe o parâmetro casa, que é a casa em que o inseto começa
def Pulos(casa):
  #Armadilha ainda desativada
  armadilha=False
  #Contagem de saltos
  Pulo=0
  #Ciclo que acompanha o inseto até ser pego pela armadilha
  while(armadilha==False):
    if casa==1:
      armadilha=True
    elif casa==2:
         #Dado pela matriz de transição a probabilidade nas casas 2,4,6,8 tem probabilidade de 1/3 para as casa adjacentes
      casa= random.SystemRandom().choice([1,3,5])
      Pulo+=1
    elif casa==3:
       #Dado pela matriz de transição a probabilidade nas casas 3,7 tem probabilidade de 1/2 para as casa adjacentes
      casa= random.SystemRandom().choice([2,6])
      Pulo+=1
    elif casa==4:
         #Dado pela matriz de transição a probabilidade nas casas 2,4,6,8 tem probabilidade de 1/3 para as casa adjacentes
      casa= random.SystemRandom().choice([1,5,7])
      Pulo+=1
    elif casa==5:
       #Dado pela matriz de transição a probabilidade na casa 5 tem probabilidade de 1/4 para as casa adjacentes
      casa=random.SystemRandom().choice([2,4,6,8])
      Pulo+=1
    elif casa==6:
         #Dado pela matriz de transição a probabilidade nas casas 2,4,6,8 tem probabilidade de 1/3 para as casa adjacentes
      casa= random.SystemRandom().choice([3,5,9])
      Pulo+=1
```

```
    elif casa==7:
       #Dado pela matriz de transição a probabilidade nas casas 3,7
tem probabilidade de 1/2 para as casa adjacentes
      casa=random.SystemRandom().choice([4,8])
      Pulo+=1
    elif casa==8:
         #Dado pela matriz de transição a probabilidade nas casas
2,4,6,8 tem probabilidade de 1/3 para as casa adjacentes
      casa= random.SystemRandom().choice([5,7,9])
      Pulo+=1
    elif casa==9:
      armadilha=True
  return Pulo
```

Item C - Utilizando o algoritmo do item b), encontre uma aproximação para as probabilidades do inseto ser capturado por cada uma das duas armadilhas do tabuleiro. Note que tais quantidades serão funções da casa inicialmente ocupada pelo inseto.

Resposta - item C

Visto pelas múltiplas execuções do código, podemos assumir que o inseto sempre será pego pela armadilha, logo, a probabilidade de captura é de 1 ou seja 100%, porém, não se sabe qual das armadilhas efetuou a prisão do meliante, logo a captura das armadilhas A e B serão frações respectivamente A/N e B/N, sendo N o número de interações do código do pulo. Assim como na questão anterior, podemos obter uma boa aproximação por um grande número de interações.

Para isso foi necessário algumas modificações no código, a função agora deve retornar também valores correspondentes para as duas armadilhas. Batizei a casa 1 e a casa 9 de armadilhas A e B, respectivamente. Seguem as modificações

```
  #Contagem de saltos e armadilhas
  Pulo=0
  TrapA=0
  TrapB=0
```

```
    if casa==1:
      armadilha=True
      #Modificação para contagem de capturas pela armadilha A
      TrapA+=1
```

```
    elif casa==9:
      armadilha=True
      #Modificação para contagem de capturas pela armadilha A
      TrapB+=1
```

```
  return Pulo, TrapA, TrapB
```

Agora para implementação da função que vai processar esses dados para o cálculo da porcentagem de captura de cada armadilha

```
# Criando a função de média de capturas de cada armadilha, recebendo
o parâmetro de interações desejadas
def MediaArmadilhas(N):
  #Loop que avança a partir da casa 2 até a casa 9
  for i in range(2,9):
    Saltos=[]
    TrpA=[]
    TrpB=[]
    #Loop que salva os valores das armadilhas
```

```
 for j in range(N):
      #Resgata uma tupla com os valores obtido nas simulações
      Resultado = Pulos(i)
      TrpA.append(Resultado[1])
      TrpB.append(Resultado[2])
    #Calculando a porcentagem de cada armadilha ativada no espaço N
    MediaA=(sum(TrpA) / N)*100
    MediaB=(sum(TrpB) / N)*100
    #Formatando as variaveis para duas casas decimais no print
    FMediaA = "{:.2f}".format(MediaA)
    FMediaB = "{:.2f}".format(MediaB)
    print("Partindo da casa",i,"Com",N,"Interações")
      print("Captura pela Armadilha A =",FMediaA,"% - Captura pela
Armadilha B =",FMediaB,"%")
```

Os resultado obtidos pela simulação com N=5000, assim como na Questão 1) foram os seguintes

```
MediaArmadilhas(5000)
Partindo da casa 2 Com 5000 Interações
Captura pela Armadilha A = 67.50 % - Captura pela Armadilha B = 32.50 %
Partindo da casa 3 Com 5000 Interações
Captura pela Armadilha A = 49.68 % - Captura pela Armadilha B = 50.32 %
Partindo da casa 4 Com 5000 Interações
Captura pela Armadilha A = 66.56 % - Captura pela Armadilha B = 33.44 %
Partindo da casa 5 Com 5000 Interações
Captura pela Armadilha A = 49.96 % - Captura pela Armadilha B = 50.04 %
Partindo da casa 6 Com 5000 Interações
Captura pela Armadilha A = 33.26 % - Captura pela Armadilha B = 66.74 %
Partindo da casa 7 Com 5000 Interações
Captura pela Armadilha A = 50.38 % - Captura pela Armadilha B = 49.62 %
Partindo da casa 8 Com 5000 Interações
Captura pela Armadilha A = 33.80 % - Captura pela Armadilha B = 66.20 %
```

Item D - Também usando o algoritmo do item b), encontre uma aproximação para o número médio de saltos que o inseto da antes de ser capturado por qualquer uma das duas armadilhas. Analogamente ao item c), isso também será uma função da casa inicialmente ocupada no tabuleiro. Qual casa inicial maximiza o número médio de saltos do inseto antes de ser capturado?

Resposta - item D

Como observado no item b) da Questão 1) uma boa aproximação para o valor esperado de saltos pode ser dado por uma grande quantidade de simulações, de forma que o valor esperado será a média do número de saltos divido pela quantidade de interações.

```
#  Criando  a  função  de  média  de  saltos,  recebendo  o  parâmetro  de
interações desejadas
def MediaSaltos(N):
  #Loop que avança a partir da casa 2 até a casa 9
  for i in range(2,9):
    Saltos=[]
    #Loop que faz a média de saltos na casa atual
    for j in range(N):
      #Salva a quantidade de saltos de cada interação
      Saltos.append(Pulos(i))
       #Média  salva  somente  o  valor  inteiro  da  média,  pois  não  é
possível dar 4,35 saltos
    Media=int(sum(Saltos) / N)
        print("A  media  de  saltos  dados  a  partir  da  casa",i,"é
de",Media,"Com ",N,"Interações")
```

Os resultado obtidos pela simulação com N=5000, assim como na Questão 1) foram os seguintes

```
A media de saltos dados a partir da casa 2 é de 4 Com  5000 Interações
A media de saltos dados a partir da casa 3 é de 6 Com  5000 Interações
A media de saltos dados a partir da casa 4 é de 5 Com  5000 Interações
A media de saltos dados a partir da casa 5 é de 6 Com  5000 Interações
A media de saltos dados a partir da casa 6 é de 4 Com  5000 Interações
A media de saltos dados a partir da casa 7 é de 6 Com  5000 Interações
A media de saltos dados a partir da casa 8 é de 4 Com  5000 Interações
```

Assumindo que essa simulação não é um devaneio completo por parte do aluno, é possível observar que o inseto partindo das casas 3, 5 e 7 obtemos a maior média de saltos por simulação, que faz certo sentido vendo que são as casas mais distantes e igualmente espaçadas das armadilhas, possibilitando o inseto se mover mais vezes antes de ser pego, pois o crime não compensa.

Item E - Assumindo que a posição inicial do inseto e a casa no canto inferior esquerdo, quantas visitas ele faz, em média, a casa central, antes de ser capturado por alguma das duas armadilhas?

Resposta - item E

Fazemos o código para o item e) análogo aos outros, inicialmente modificando a função Pulos(), as modificações para a contagem foram as seguintes

```
  #Contagem de saltos, armadilhas e visitas a casa 5
  Pulo=0
  TrapA=0
  TrapB=0
  Visita5=0
```

```
    elif casa==5:
       #Dado pela matriz de transição a probabilidade na casa 5 tem probabilidade de 1/4 para as casa adjacentes
      casa=random.SystemRandom().choice([2,4,6,8])
      Pulo+=1
      Visita5+=1
```

```
  return Pulo, TrapA, TrapB, Visita5
```

Agora a implementação da função que vai encontrar a média de visitas à casa central(casa 5) partindo do canto inferior esquerdo(casa 7)

```
def MediaVisita7(N):
  V7=[]
  for i in range(N+1):
      #Resgata uma tupla com os valores obtido nas simulações
      Resultado = Pulos(7)
      #Salva a quantidade de saltos de cada interação
      V7.append(Resultado[3])
  Media=(sum(V7) / N)
   print("O numero de visitas a casa 5 partindo da casa 7 é de em media",Media,"visitas Com ",N,"Interações")
```

Os resultados da interação com N também similar às questões anteriores, ou seja N=5000, obtemos

```
MediaVisita7(10000)
O numero de visitas a casa 5 partindo da casa 7 é de em media 1.0035 visitas Com  10000 Iterações
```