Några Pythontips för laboration 1

Q ska ersättas med namnet på instansen av klassen queue som du skapat i MM1.py. På samma sätt ska S ersättas av namnet på instansen av klassen sink.

# Uppgift 3.1

Programmet i MM1.py är inte komplett. Du måste lägga till fem rader kod kod innan while-slingan i programmet. Se programmet som finns i filen exempel.py som finns på Canvassidan. Observera dock att det behövs fler parametrar (en för i generator och en för i queue) än vad som finns i exempel.py.

När du ska plotta antalet kunder som finns i kösystemet så kan du använda kommandot:

plt.plot(Q.measuredValues)

Du kan skriva raden ovan sist i simuleringsprogrammet eller skriva den som ett kommando i Spyders kommandofönster när du har kört simuleringsprogrammet. Om du inte ser något diagram i Spyder så prova att tryck Control+Shift+g.

# Uppgift 3.2

För att beräkna medeltalet kunder så använd kommandot:

np.mean(Q.measuredValues)

Kan läggas till sist i filen MM1.py eller skrivas in i Spyders kommandofönster.

# Uppgift 3.4

Lägg till följande rader kod i slutet av MM1.py eller skriv in dem i kommandofönstret för att rita ett histogram med 15 staplar:

a = list(range(0, 15))

plt.hist(Q.measuredValues, density=False, bins=a)

Det är lagom att ha 15 staplar I histogrammen i denna laboration.

# Uppgift 3.5

För att rita ett normerat histogram så ändra det ovan till:

a = list(range(0, 15))

plt.hist(Q.measuredValues, density=True, bins=a)

det vill saga density=False blir i stället density=True.

# Uppgift 3.6

För att rita ett histogram och kurvan som fås från programmet pkMM1.py gör så här:

1. Kör programmet pkMM1.py och välj ankomstintensitet = 7, betjäningsintensitet = 10 och maximalt k-värde att plotta till 15.
2. Klistra in följande tre rader i kommandofönstret:  
   plt.hist(q.measuredValues, density=True, bins=a)  
   plt.plot(pk)  
   plt.show()
3. Flytta markören till efter plt.show() om den inte redan är där och slå return.

# Uppgift 3.12

Kommandot ska vara:

plt.plot(S.T[2:101], S.T[1:100], ’\*’)

Om du gör copy/past på denna rad in i Spyder så måste du sudda bort ’\*’ i Spyder och skriva det igen i Spyder. Words tecken ’ kan Spyder inte tolka.

för T finns ju i klassen sink.

# Uppgift 4.1

Den deterministiska fördelningen innebär att alla betjäningstider är lika långa.

Den likformiga (uniform) fördelningen får man med följande kod:

def serviceTime(self):

return simTime + random.uniform(0, 2\*mean)

där mean är det önskade medelvärdet, ska ersättas med värdet som är aktuellt i denna uppgift.

Exponentialfördelningen finns redan i MM1.py.

Den hyperexponentiella fördelningen får man i två steg: först drar man ett slumptal för att välja en exponentialfördelning. När man har gjort det så drar man ett slumptal från den valda exponentialfördelningen. Kod för detta:

def serviceTime(self):

if random.random() < alpha:

return simTime + random.expovariate(mu1)

else:

return simTime + random.expovariate(mu2)

Ersätt alpha, mu1 och mu2 med lämpliga värden (finns i labbhandledningen).

# Uppgift 5.2

Gör detta genom att i simuleringsprogrammet räkna hur många kunder som kommer till kösystemet och hur många kunder som spärras. Dividera sedan för att få en skattning av sannolikheten att en kund spärras.