Report on Simulation Results

# Part 1A

Below is the code for part 1A to generate the answers

#!/usr/bin/env python

# Part 1A, Daniel Jake Davies, C1120627

import math # for math.log

import random # random number 0-1

def nextTime(alpha): # define the function

answer=(alpha\*-1)\*math.log(1-random.random()) # implement the equation

return answer # return it back

I then import this function. Here is the code from my Python session:

>>> total = 0.0

>>> for i in range(10000):

total = total + nextTime(1)

>>> total / 10000

0.9939331101577537

This table shows the results for other experiments, based on iterations. These are passed an argument to the nextTime function. To calculate the average I have looped the code 10 000 times, and divided the answer by that many.

|  |  |
| --- | --- |
| **α** | **Average Result** |
| 1 | 1.0045883771562161 |
| 10 | 9.9688643911873207 |
| 50 | 50.596664453308705 |
| 100 | 100.35999497054409 |

# Part 1B

Below is the code for part 1B to generate the answers

#!/usr/bin/env python

# Part 1B, Daniel Jake Davies, C1120627

import math # for math.log

import random # random number 0-1

# define the function

def partB(alpha,beta):

ta = 0 # next customer arrival

ts = 0 # time until customer has finished

c = 0 # current time

q = 1 # current length of the queue

m = 0 # maximum var

while c <= 480: # time less than or equal to 8 hours

if ta < ts:

ts -= ta

c = c+ta

q += 1

m = max([m,q])

ta = (alpha\*-1)\*math.log(1-random.random())

while q == 0:

q += 1

c = c + ta

ta = (alpha\*-1)\*math.log(1-random.random())

else:

ta = ta - ts

c = c + ts

q -= 1

ts = (beta\*-1)\*math.log(1-random.random())

while q == 0:

q += 1

c = c + ta

ta = (alpha\*-1)\*math.log(1-random.random())

return m

Below is the result of running partB in a interactive Python session:

>>> total = 0.0

>>> for i in range(10000):

total = total + partB(10,4)

>>> total / 10000.0

4.3289

The table below display the other results and respective iterations with parameters passed to the function “partB”. The average, like before, is calculated looping through the code 10,000 and then dividing by that amount.

|  |  |  |
| --- | --- | --- |
| **α** | **β** | **Average Result** |
| 10 | 4 | 4.3520000000000003 |
| 10 | 2 | 3.0144000000000002 |
| 5 | 4 | 10.6092 |
| 5 | 2 | 5.0359999999999996 |