COIS 4470H: Modelling and Simulation Winter 2017

Assignment 2

Due: Thursday, Mar. 9, 2017

1. **Inventory system:** An automobile dealership uses a weekly periodic inventory review policy. Assume the maximum space for cars is S=80 and the minimum inventory level is s=20. Operation costs are assumed as:
   * Holding cost (C\_holding) - $25 per car per week
   * Shortage cost (C\_Shortage) - $700 per car per week
   * Set up cost (C\_SetUp) - $1000 per order
   * Unit cost (C\_Unit) - $8000 each car ordered
2. Modify the program **sis1.c** to compute all four components of the total average cost per week.
3. Use your program to compute and complete the following table (S=80):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *s* | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
| Average holding cost/week | 854.55 | 917.4 | 917.4 | 955.71 | 1060.03 | 1144.44 | 1207.93 | 1262.14 | 1277.42 |
| Average shortage cost/week | 795.77 | 374.34 | 374.34 | 345.50 | 172.47 | 15.88 | 1.48 | 0.29 | 1.19 |
| Average setup cost /week | 320 | 340 | 340 | 350 | 390 | 440 | 470 | 500 | 510 |
| Sum of the three costs/week | 1970.31 | 1631.84 | 1631.84 | 1651.21 | 1622.51 | **1600.32** | 1679.41 | 1762.43 | 1788.61 |

1. What could be the optimum value for *s*? Explain.

As it is indicated in the table above, we can conclude that the S holding of 25 will have the least expense in average for the company. It is important to note while none of the average holding, shortage or setup is optimum, however it is sum of cost with value of 1600.32 in average per week.

1. Redo (a)-(c). Instead of reading the demands from the input file (sis1.dat), using random-variate generation techniques. Assume demands are uniformly distributed in the same range as the data in file sis.dat.

**Data needed for Part d and e, taken from sis1.dat**

|  |  |  |
| --- | --- | --- |
| Maximum | Minimum | Average |
| 48 | 17 | 29.29 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S | **Average holding cost/week** | **Average shortage cost/week** | **Average setup cost /week** | **Sum of the three costs/week** |
| **0** | 859.37 | 1371.69 | 340 | 2571.34 |
| **5** | 866.641 | 1074.94 | 350 | 2291.590 |
| **10** | 970.20 | 495.64 | 400 | 1865.84 |
| **15** | 1014.84 | 254.19 | 410 | 1676.03 |
| **20** | 1069.76 | 168.54 | 440 | 1678.31 |
| **25** | 1121.90 | 91.93 | 460 | 1673.84 |
| **30** | **111.91** | **32.74** | **470** | **1620.66** |
| **35** | 1212.183 | 5.12 | 520 | 1737.31 |
| **40** | 1334.75 | 0 | 640 | 1974.75 |

\*Please see the source code attached.

1. Redo (a)-(c). Instead of reading the demands from the input file (sis1.dat), using random-variate generation techniques. Assume demands follow Geometric Distribution with the same mean as the data in file sis.dat.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S | **Average holding cost/week** | **Average shortage cost/week** | **Average setup cost /week** | **Sum of the three costs/week** |
| **0** |  |  |  |  |
| **5** |  |  |  |  |
| **10** |  |  |  |  |
| **15** |  |  |  |  |
| **20** |  |  |  |  |
| **25** |  |  |  |  |
| **30** |  |  |  |  |
| **35** |  |  |  |  |
| **40** | 1978.5 | 0 | 10 | 1988.5 |

1. Compare results obtained for different demand distributions.
2. A random number generator can be developed by combining **two** Linear Congruential Generators using the following algorithm:

The first generator has multiplier *a1* and modulus *m1*, The second generator has multiplier *a2* and modulus *m2*.

Step 1: Select seed *X1,0* in the range of [1, *m1*-1] for the first generator and seed *X2,0* in the range of [1, *m2*-1] for the second generator. Set j=0.

Step 2: Evaluate each individual generator:

*X*1, *j*1 *a*1*X*1, *j* mod *m*1 *X*2 *j*1 *a*2 *X*2, *j* mod *m*2

Step 3:

*Xj*1 (*X*1, *j*1 *X*2 *j*1) mod *m*1

Step 4:

Return *R*

*j*1

*Xj*1 , if *X*

*m*1





*j*1

0,

*m*1 1, if *X*

0.

*m*1

*j*1

Step 5:

Set *j* *j* 1 and go to Step 2.

1. Following this algorithm, develop a Combined Linear Congruential Generator. The seed *X*1,0and *X*2,0 , parameters *a*1, *m*1, *a*2, *m*2 , and number of random numbers generated should be given by the user at run time.

Please Enter x1(Greater than zero)?)

15

Please Enter a1(Greater than zero)?)

11

Please Enter M1(Greater than zero)?)

65

Please Enter x2(Greater than zero)?)

5

Please Enter a2(Greater than zero)?)

12

Please Enter M2(Greater than zero)?)

14

x1=35, x2=4 xfinal=31

 Just added 0.476923076923077 as the new random calculated number

x1=60, x2=6 xfinal=54

 Just added 0.830769230769231 as the new random calculated number

x1=10, x2=2 xfinal=8

 Just added 0.123076923076923 as the new random calculated number

x1=45, x2=10 xfinal=35

 Just added 0.538461538461538 as the new random calculated number

x1=40, x2=8 xfinal=32

 Just added 0.492307692307692 as the new random calculated number

x1=50, x2=12 xfinal=38

 Just added 0.584615384615385 as the new random calculated number

x1=30, x2=4 xfinal=26

 Just added 0.4 as the new random calculated number

x1=5, x2=6 xfinal=1

 Just added 0.0153846153846154 as the new random calculated number

x1=55, x2=2 xfinal=53

 Just added 0.815384615384615 as the new random calculated number

x1=20, x2=10 xfinal=10

 Just added 0.153846153846154 as the new random calculated number

x1=25, x2=8 xfinal=17

 Just added 0.261538461538462 as the new random calculated number

x1=15, x2=12 xfinal=3

 Just added 0.0461538461538462 as the new random calculated number

x1=35, x2=4 xfinal=31

Please see the source code attached to the file

1. Run your program using the input

*X*1,0 7, *X*2,0 8, *a*1 11, *m*1 16, *a*2 3, *m*2 32,

x1=13, x2=24 xfinal=11

 Just added 0.6875 as the new random calculated number

x1=15, x2=8 xfinal=7

 Just added 0.4375 as the new random calculated number

x1=5, x2=24 xfinal=3

 Just added 0.1875 as the new random calculated number

x1=7, x2=8 xfinal=1

 Just added 0.0625 as the new random calculated number

x1=13, x2=24 xfinal=11

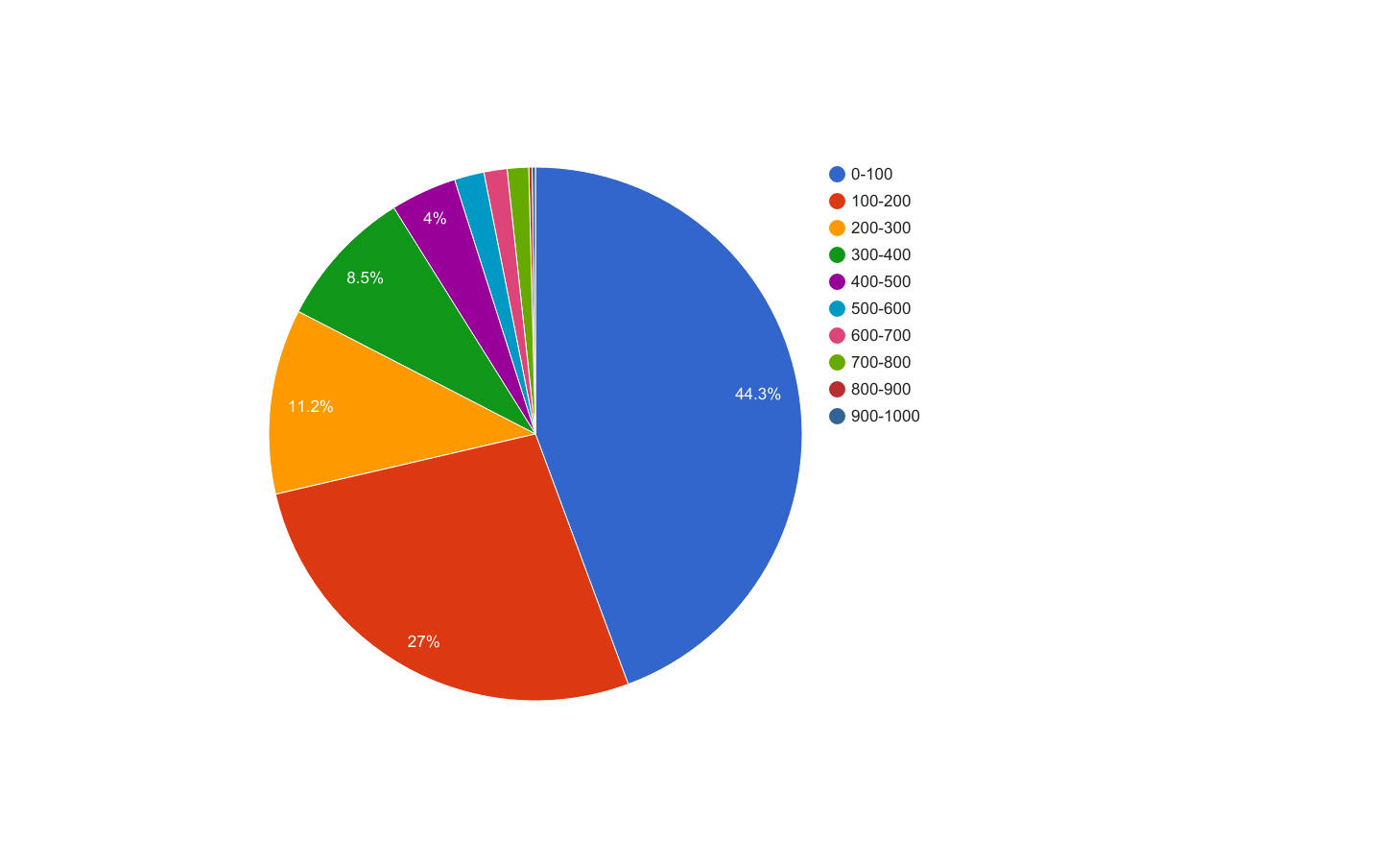
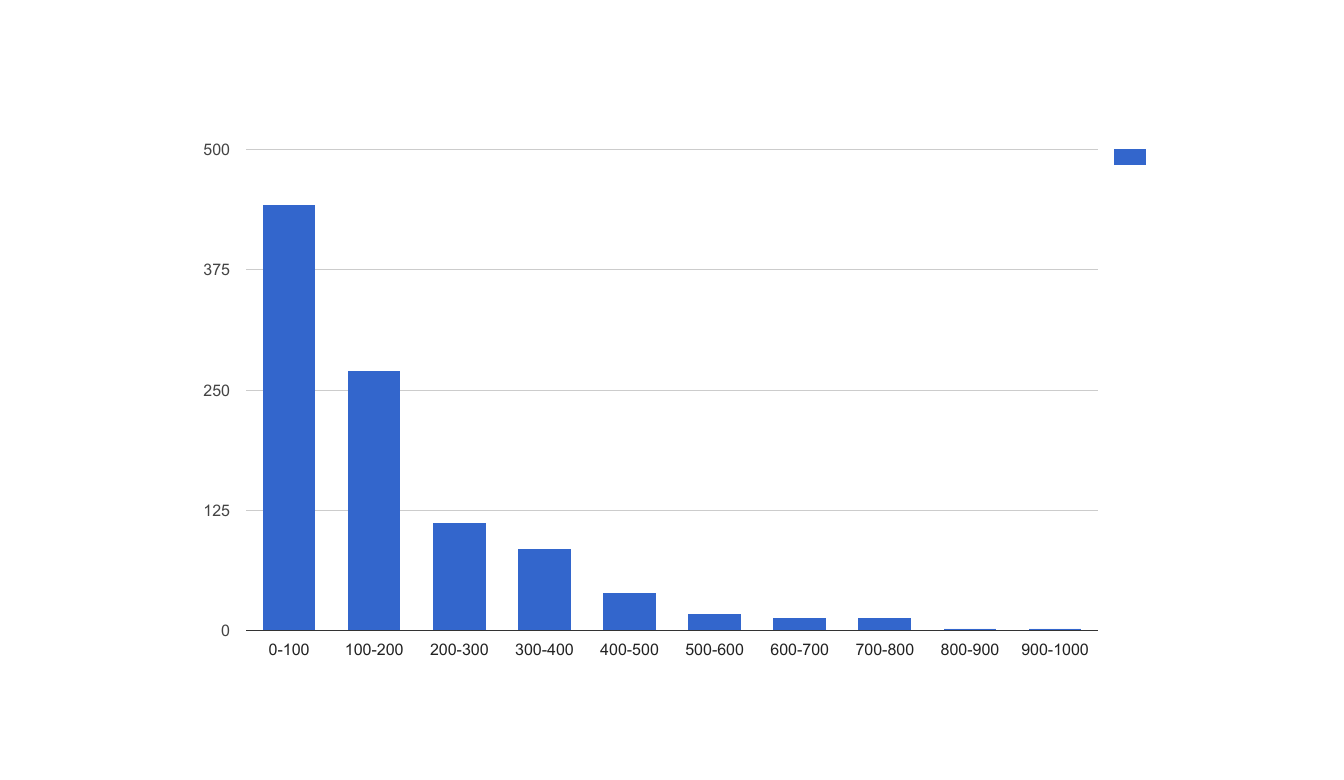
 Just added 0.6875 as the new random calculated number

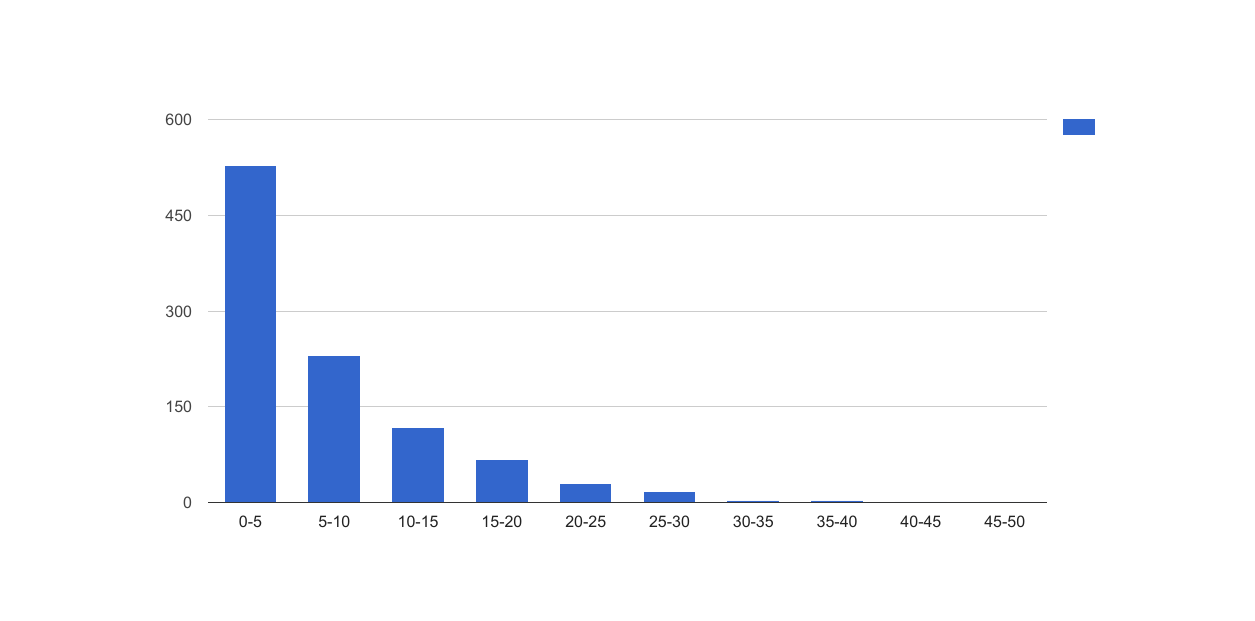
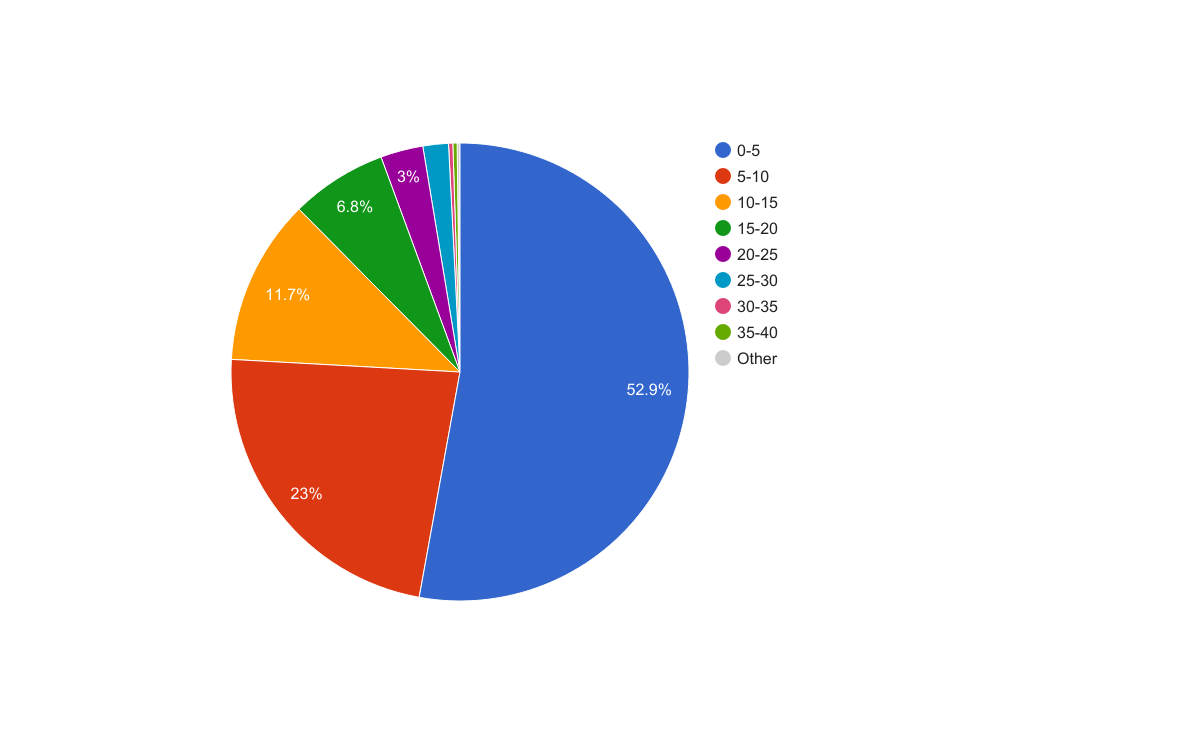
x1=15, x2=8 xfinal=7

As you can see above the cycled repeat itself after generating 4 distinguished number

1. Apply the Gap Test with the interval (0.2, 0.5) to determine if the random variates generated are independent (α = 5%.).
2. Apply the formulas discussed in class to develop a random variate generator for exponential distribution. Run the program using 7 and 170 to generate 1000 random variate respectively. Plot histograms for the obtained random variates.

**Q=170**



**Q=7**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0-100** | **100-200** | **200-300** | **300-400** | **400-500** | **500-600** | **600-700** | **700-800** | **800-900** | **900-1000** |
| 443 | 270 | 112 | 85 | 40 | 18 | 14 | 13 | 2 | 2 |
| **0-5** | **5-10** | **10-15** | **15-20** | **20-25** | **25-30** | **30-35** | **35-40** | **40-45** | **45-50** |
| 528 | 230 | 117 | 68 | 30 | 18 | 3 | 3 | 1 | 1 |

1. Applying the Frequency Test to test the following sequence of numbers for uniformity, using s = 10 subintervals and α = 5%.

0.594, 0.928, 0.515, 0.055, 0.507, 0.351, 0.262, 0.797, 0.788, 0.442, 0.097, 0.798,

0.227, 0.127, 0.474, 0.825, 0.007, 0.182, 0.929, 0.852

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 0-0.1 | 2 | 3 |
| 0.1-0.2 | 2 | 2 |
| 0.2-0.3 | 2 | 2 |
| 0.3-0.4 | 2 | 1 |
| 0.4-0.5 | 2 | 2 |
| 0.5-0.6 | 2 | 3 |
| 0.6-0.7 | 2 | 0 |
| 0.7-0.8 | 2 | 3 |
| 0.8-0.9 | 2 | 2 |
| 0.9-1.0 | 2 | 2 |
|  | 20 | 20 |

**Note:**

Give all your answers and discussions in a pdf file named: yourLastName-A2. Submit both the answer file and all programs on Blackboard by 11:59pm on the due date.