1) Circle one of the following (-50 points): I come to class at 8:00 am 12:45 pm
2) Please remove all hats. If it is part of your head, turn it around backwards.
3) Please note that if your work is not legible, or if I cannot follow your logic at a glance, it will receive zero credit. If your code is scattered and messy I will not be able to follow it, so leave yourself time to recopy it if necessary. Better yet, just leave a lot of space between lines of code as you write it to fill in things later on that you forgot.
4) You MUST fully comment each and every line of code you write, specifying who was born, why they are waiting, why they are closing the gate, everything! Any line that is not commented will not be graded.

Problem 1) (10 points) Give me your best estimate for when the next person will arrive, for the random number = 0.7, if the following statement is given. ARRIVE{TIME = 0 MAX NORMAL(12,3)}; I realize that you don’t have the equation to map a random number into a normal distribution, and that you don’t have a set of Z tables. That’s why it will be an estimate. But make it your best estimate and fully justify your answer with words, graphs, whatever you can to get credit. The closer it comes to the truth, and the better you justify what you get, the better your grade.

Problem 2) (5 points) For the distribution shown, determine the next interarrival time for a random number of 0.8:

Problem 3) (10 points) I have started getting a check for $1800/month from Social Security and am going to buy me one of these. It costs $80,000, nothing down, at 6%, and can be obtained overnight from Stephan at Russki Auto Werks, during the night. If I am willing to devote my full check to paying for this car, about how long will it take me to pay it off? I don’t want it exact, just about.

Problem 4) (35 points) A concrete batch plant makes concrete at a rate of 100 cy/hour. It starts the day at 8:00 am with no reserves. The concrete reserve tank holds 500 cy. The plant serves the Bryan/College Station area which has 2 sizes of trucks. 30% are big (20 cy) and the rest are 12 cy. These trucks arrive at the plant exponentially distributed with a mean of 15 minutes. They take from 10 to 15 minutes to fill and leave, normally distributed. There are two loading docks at the plant.

The plant stops making concrete at 3:00 pm and remains open for offloading until trucks haul the last of their reserves away. You do not need to worry about what to do with the trucks that come after all the concrete is gone. Just let them sit there. They will finally get the idea and go home on their own.

Write me a BOSS program to study this system. You MUST fully comment each and every line of code specifying who was born, why they are waiting, why they are closing the gate, everything!
Problem 5) (5 points) For the program listed below, determine the value of ducks at the end of the run. You may assume that the program runs.

PROGRAM
  "duck counter"

  DEFINITION
  LABELS = {fowl};
  ducks=0;

  CONTROL
  STOPTIME=32.567;

  LOGIC
  ARRIVE {TIME=1};
  fowl: ducks = ducks + 1;
  DEPART{ }; END.

Problem 6) (35 points) I will be tutoring a group of students for the F.E. exam in room 110 from 8:00 am to 12:00 noon. There are 117 seats in the room. Students arrive exponentially distributed with a mean of 1 minute from 8:00 until 9:00. They arrive normally distributed with a mean of 2 minutes and a sigma of 0.5 minute from 9:00 until 11:00, and they arrive continuous uniform (3 to 4 minutes) from 11:00 until 12:00. If they come and the room is full, they wait for about one minute (sigma = 0.2 minutes) and check again. If the room is still full, they wait one more minute (sigma = 0.2 minutes), and check again. If the room is still full, they leave. Once in the room, a student will listen for about an hour, (sigma = 10 minutes) and then leave.

Write a BOSS program to study my tutoring session. I am especially interested in how many students went away for lack of space in the room.
Interarrival Times

For a normal distribution:

<table>
<thead>
<tr>
<th>Random Number</th>
<th>0.7</th>
<th>0.254</th>
<th>0.607</th>
<th>0.633</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Sigma</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Time</td>
<td>13.5732</td>
<td>10.01413</td>
<td>12.81453</td>
<td>13.01943</td>
</tr>
</tbody>
</table>

For an exponential distribution:

<table>
<thead>
<tr>
<th>Random Number</th>
<th>0.5</th>
<th>0.896</th>
<th>0.254</th>
<th>0.904</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
</tr>
<tr>
<td>Time</td>
<td>0.086643</td>
<td>0.282921</td>
<td>0.036629</td>
<td>0.292926</td>
</tr>
</tbody>
</table>

For a uniform distribution:

<table>
<thead>
<tr>
<th>Random Number</th>
<th>0.5</th>
<th>0.812</th>
<th>0.211</th>
<th>0.637</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left end value</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Right end value</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>Time</td>
<td>13</td>
<td>13.624</td>
<td>12.422</td>
<td>13.274</td>
</tr>
</tbody>
</table>

Diagram:

- Diagram anything between 13.6 and 15.
Problem 2

\[ \text{Need an additional 100% of area} \]

\[ \text{Area} = \frac{3(1)}{2} + (6)(1.0) = 7.5 \text{ total} \]

20% is inside 3 to 6 mins

\[ \text{need a total} = 7.5 \times 0.8 = 6 \]

Already got \( \frac{(6-3)(1)}{2} = 1.5 \)

\[ \text{Need additional 4.5} \]

\[ (t_{\text{final}} - 6)(1) + \frac{(6-3)(1)}{2} = (7.5)(0.8) \]

\[ t_{\text{final}} - 6 = (7.5)(0.8) - 1.5 \]

\[ t_{\text{final}} = 10.5 \text{ mins} \]
$P = A(P/A, i, m) - 3$

$\frac{P}{A} = \frac{80000}{1800} = 44.4 - 3$

$50\text{ months}$
PROGRAM
"concrete trucks"

DEFINITION
dock:RESOURCE={CAPACITY=2}; "born two unloading docks"
cconcreteintank = 0;
cconcretedelivered = 0;
LABELS = {makemore, isitfull, waitone, tryagain, loadit};
ATTRIBUTES = {size=12};

CONTROL
STOPTIME = 9*60; "shut plant doors at 5:00 pm"

LOGIC
ARRIVER={TIME = 0, LIMIT = 1}; "born a concrete maker"
makemore: concreteintank = concreteintank+100/60; "add 1 minute of concrete to tank"
  IF {CLOCKTIME>=7*60} THEN DEPART(); "stop making concrete but allow trucks to keep unloading"
waitone: WAIT={TIME = 1}; "wait one minute"
isitfull: IF concreteintank >= 499 THEN GOTO waitone; "check to see if tank is full"
  GOTO makemore; "ok to go make more concrete"

  ARRIVER={TIME = EXPD(15)}; "born concrete trucks every 15 minutes"
  IF RANDOM >= 0.7 THEN size = 20; "reset size of this truck"
  SEIZE={NAME = dock}; "take over one of the docks"
tryagain: IF(concreteintank >= size) THEN GOTO loadit; "see if enough concrete to load truck"
  WAIT={TIME = 1}; "wait a minute for more concrete to be made"
  GOTO tryagain;
loadit: concreteintank = concreteintank - size; "reserve size cubic yards of concrete for you"
  WAIT={TIME = 0 MAX NORMAL(12.5, 5/6)}; "load your reserved concrete"
  RELEASE={NAME=dock}; "let next truck have the dock"
  cconcretedelivered = cconcretedelivered + size; "not asked, but concrete delivered"
  DEPART(); "leave"

END.
PROGRAM
"duck counter"

DEFINITION
LABELS = {fowl};
ducks=0;
CONTROL
STOPTIME=20;
LOGIC
ARRIVE {TIME=1};
fowl: ducks = ducks + 1;
DEPART{ };


PROGRAM
"f.e. exam"

DEFINITION
ATTRIBUTES={mytries=0};
seat:RESOURCE = {CAPACITY = 117};
wentawaymad = 0;
LABELS = {tryagain, classfull, im_mad};

CONTROL
STOPTIME = 4*60;

LOGIC

ARRIVE{TIME = EXPD(0.01)}; "born students before 9 am"
IF(CLOCKTIME >= 1*60) THEN DEPART{}; "get rid of students coming after 9:00 am"
GOTO tryagain;
ARRIVE{TIME = 0 MAX NORMAL(2,0.5)}; "born students for between 9 and 11 am"
IF(CLOCKTIME NOTIN [1*60,3*60]) THEN DEPART{}; "get rid of students before 9 or after 11"
GOTO tryagain;
ARRIVE{TIME = CUNIFORM(3,4)}; "born students after 11 am"
IF(CLOCKTIME <= 3*60) THEN DEPART{}; "get rid of students coming before 11:00 am"
GOTO tryagain;

tryagain: SEIZE{NAME = seat, REXCESS = classfull}; "grab seat unless full, else go to wait"
WAIT{TIME = 0 MAX NORMAL(60,10)}; "listen to lecture"
RELEASE{NAME = seat}; "release your seat in class"
DEPART{}; "go home"

classfull: WAIT{TIME=1}; "wait one minute"
mytries = mytries+1; "keep count of your personal number of tries"
IF (mytries >= 3) THEN GOTO im_mad; "give up in despair"
GOTO tryagain; "see if class is still full"

im_mad: wentawaymad = wentawaymad + 1;
DEPART{};

END.

□