Midterm for 597ST/697ST (Solution)
10/25/11

Note: In all written assignments, please show as much of your work as you can. Even if you get a wrong answer, you can get partial credit if you show your work. If you make a mistake, it will also help the grader show you where you made a mistake.

Problem 1:
Write a program (in pseudo code) that generates 100000 random variables that are (discretely) uniform distributed as follows:

<table>
<thead>
<tr>
<th>#</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>#33</td>
<td>5%</td>
</tr>
<tr>
<td>#2</td>
<td>10%</td>
</tr>
<tr>
<td>#35</td>
<td>5%</td>
</tr>
<tr>
<td>#4</td>
<td>12%</td>
</tr>
<tr>
<td>#37</td>
<td>25%</td>
</tr>
<tr>
<td>#6</td>
<td>22%</td>
</tr>
<tr>
<td>#39</td>
<td>21%</td>
</tr>
</tbody>
</table>

Solution:
for i = 1:100000
    u = rand (i);
    if u < 0.25
        x(i) = 37;
    elseif u < 0.47
        x(i) = 6;
    elseif u < 0.68
        x(i) = 39;
    elseif u < 0.80
        x(i) = 4;
    elseif u < 0.95
        x(i) = 33;
    else
        x(i) = 35;
    end
end

Problem 2:
If \(x_0 = 3\) and 
\[x_n = (5x_{n-1} + 7) \mod 200\]
find \(x_1, \ldots, x_{10}\).

Solution:
\(x_1=22, x_2=117, x_3=192, x_4=167, x_5=42, x_6=17, x_7=92, x_8=67, x_9=142, x_{10}=117\)
Problem 3:
Use simulation to approximate the following integrals. Present a solution of the simulation in pseudo code that generates 100,000 random variables.

a. $\int_{0}^{7} x^{4} - 1 \, dx$, also calculate the exact solution.
b. $\int_{0}^{\infty} e^{-x^2+2} \, dx$
c. Imagine you were a poor darts player who would (perfectly) randomly throw darts at a square. How could you use this method to determine $\pi$?

Solution:

a. Calculation:
$$\left[\frac{x^5}{5}\right]_{0}^{7} - [x]_{0}^{7} \approx 3354$$
Simulation:
Interval 0 to 5 to 0 to 1:
$$y = \frac{(x-a)}{(b-a)} \, dy = \frac{dx}{(b-a)}$$
$$\int_{0}^{1} f(y(b-a) + a)(b-a)dy = \int_{0}^{1} f(7y) * 7\, dy$$
$$\int_{0}^{1} = \int_{0}^{1} ((7y)^4 - 1) * 7\, dy = \int_{0}^{1} 16807y^4 - 7\, dy$$

for $i = 1:100000$
$$u = \text{rand} (1);$$
$$y = y + 16807u^4 - 7;$$
end
$$y = y/100000$$

b. $\int_{-\infty}^{\infty} e^{-x^2+2} \, dx$

$$y = \frac{1}{x+1}, \, dy = -\frac{1}{(x+1)^2}$$
$$\int_{0}^{\infty} e^{-\frac{1}{(y-1)^2}+2} dy$$

for $i = 1:100000$
$$u = \text{rand} (1);$$
$$y = y + e^\left(-(1/u-1)^2\right)+2/u^2;$$
end
$$y = 2*y/100000$$

c.
If you are a very poor dart player, it is easy to imagine throwing darts randomly at the figure above, and it should be apparent that of the total number of darts that hit within the square, the number of darts that hit the shaded part (circle quadrant) is proportional to the area of that part. In other words,
\[
\frac{\# \text{ darts hitting shaded area}}{\# \text{ darts hitting inside square}} = \frac{\pi r^2}{4r^2}
\]
\[
\pi = 4 \times \frac{\# \text{ darts hitting shaded area}}{\# \text{ darts hitting inside square}}
\]

If you remember your geometry, it’s easy to show that
If each dart thrown lands somewhere inside the square, the ratio of "hits" (in the shaded area) to "throws" will be one-fourth the value of pi. If you actually do this experiment, you'll soon realize that it takes a very large number of throws to get a decent value of pi...well over 1,000. To make things easy on ourselves, we can have computers generate random* numbers.

Problem 4:
Provide the inverse transform algorithm for generating a random variable X with mean \( \lambda \) if
\[
p_i = P\{X = i\} = e^{-\lambda} \frac{\lambda^i}{i!}, \quad i = 0, 1, ...
\]
and make use of the recursive identity
\[
p_{i+1} = \frac{\lambda}{i+1} p_i.
\]

Solution:
STEP 1: Generate a random number U.
STEP 2: \( i=0, \quad p=e^{-\lambda}, \quad F=p. \)
STEP 3: If \( U<F, \) set \( X=i \) and stop.
STEP 4: \( p=\lambda p/(i+1), \quad F=F+p, \quad i=i+1. \)
STEP 5: Go to Step 3.

Problem 5:

a. Use the inverse transform method to generate a random variable having distribution function
\[
F(x) = 1 - e^{-x}, \quad 0 \leq x \leq 1
\]
and present and algorithm (in pseudo code), which generates 100,000 random variables of this distribution.
b. Use the inverse transform method to generate a random variable having probability density function:

\[ f_x(x) = \begin{cases} 
2x & 0 \leq x \leq 1 \\
0 & \text{otherwise} 
\end{cases} \]

As a first step to solve this problem, calculate the probability distribution of \( x \)

\[ F(x) = \begin{cases} 
\int_0^x 2t \, dt = x^2 & 0 \leq x \leq 1 \\
1 & x > 1 
\end{cases} \]

Present and algorithm (in pseudo code), which generates 100,000 random variables of this distribution.

**Solution:**

a. \( u = 1 - e^{-x} \)

\[ u - 1 = -e^{-x} \]

\[ 1 - u = e^{-x} \]

\[ f_x(x) = \begin{cases} 
2x & 0 \leq x \leq 1 \\
0 & \text{otherwise} 
\end{cases} \]

\[ F(x) = \begin{cases} 
0 & x < 0 \\
1 & x < 1 \\
1 & x > 1 
\end{cases} \]

\[ U = X^2 \]

\[ X = \frac{F^{-1}(U)}{U^{1/2}} \]

**Problem 6:**

We are interested in simulating a queuing system with ONE queue and TWO parallel servers to determine such quantities as the average time a customer spends in the system and the average time past \( T \) that the last customer departs.

We use the following variables:

- **Time Variable** \( t \)
- **System State (SS) Variable**
  \((n, i_1, i_2)\): if there are \( n \) customers in the system, \( i_1 \) is with server 1 and \( i_2 \) is with server 2. \( \text{SS}=(0) \) when system empty, \( \text{SS}=(1,j,0) \) or \( (1,0,j) \) when only customer is \( j \) and served by either server 1 or 2.
- **Counter Variables**
  \( N_a \): the number of arrivals by time \( t \)
  \( C_j \): the number of customers served by \( j \), by time \( t \)
• **Output Variable**
  \( A(n) \): arrival time of customer \( n, n \geq 1 \)
  \( D(n) \): departure time of customer \( n, n \geq 1 \)

• **Event List** \( t_A, t_1, t_2 \)
  \( t_A \): time of next arrival
  \( t_i \): service completion time at server \( i \)

The simulation will begin by the initialization of the variables and the event times \( t_A \) and \( t_B \).

**Initialize**
Set \( t=N_A=C_1=C_2=0 \).
Set \( SS=(0) \).
Generate \( T_0 \) and set \( t_A=T_0, t_1, t_2=\infty \).

Fill in the missing lines for cases 1-3!

**Case 1:** \( SS=(n, i_1, i_2) \)  \( t_A=min(t_A, t_1, t_2) \)
Reset: \( t=t_A \).
Reset: \( N_A=N_A+1 \).
Generate \( T_B \) and set \( t_A=T_B, t_1, t_2=\infty \).
Collect output data \( A(N_A)=t \).
IF \( SS=(0) \):
  Reset: \( SS=(1, N_A, 0) \).
  Generate \( Y_1 \) and reset \( t_1=t+Y_1 \).
IF \( SS=(1, j, 0) \):
  Reset: \( SS=(2, j, N_A) \).
  Generate \( Y_2 \) and reset \( t_2=t+Y_2 \)
IF \( SS=(1, 0, j) \):
  Reset: \( SS=(2, N_A, j) \).
  Generate \( Y_1 \) and reset \( t_1=t+Y_1 \)
IF \( n>1 \):
  Reset: \( SS=(n+1, i_1, i_2) \).

**Case 2:** \( SS=(n, i_1, i_2) \) and \( t_1<t_A, t_1\leq t_2 \)
Reset: \( t=t_1 \).
Reset: \( C_1=C_1+1 \)
Collect output data \( D(i_1)=t \).
If \( n=1 \):
  Reset: \( SS=(0) \).
  Reset: \( t_1=\infty \).
If \( n=2 \):
  Reset: \( SS=(1, 0, i_2) \).
  Reset: \( t_1=\infty \).
If \( n>2 \):
Let $m = \max(i_1, i_2)$
Reset: $SS = (n-1, m+1, i_2)$.
Generate $Y_1$ and reset $t_1 = t + Y_1$.

**Case 3:** $SS = (n, i_1, i_2)$ and $t_2 < t_3$, $t_2 < t_1$
Reset: $t = t_2$.
Reset: $C_2 = C_2 + 1$
Collect output data $D(i_2) = t$.
If $n = 1$:
  Reset: $SS = (0)$.
  Reset: $t_2 = \infty$.
If $n = 2$:
  Reset: $SS = (1, i_1, 0)$.
  Reset: $t_2 = \infty$.
If $n > 2$:
  Let $m = \max(i_1, i_2)$
  Reset: $SS = (n-1, i_1, m+1)$.
  Generate $Y_2$ and reset $t_2 = t + Y_2$. 