Answers to Sample Midterm Questions

1. Claims processing. (This is an example of both a “fork-join” queueing system and a workflow system.)

(a) GSMP specification:

(i) **State space**: \( S = \{ (n_1,n_2,m) \in \{0,1,2,...\}^3 : n_1 \leq m \text{ and } n_2 \leq m \} \)

(ii) **Event set mapping**: For \( s = (n_1,n_2,m) \in S \)
\[ e_i \in E(s) \text{ if and only if } n_i > 0 \]
\[ e_i \in E(s) \text{ if and only if } n_i > 0 \]
\[ n_i \leq m - n_i > 0 \text{ and } m - n_i > 0 \]
\[ \text{[equivalently: } e_i \in E(s) \iff \min(m-n_i,m-n_i) > 0; e_i \in E(s) \iff m - \max(n_i,n_i) > 0] \]

(iii) **Speeds**: All speeds \( r(s,e) \) are equal to 1.

(iv) **State-transition probabilities**:
\[ p(s',s,e_i) = 1 \text{ when } s = (n_1,n_2,m) \text{ and } s' = (n_1+1,n_2+1,m+1) \]
\[ p(s',s,e_i) = 1 \text{ when } s = (n_1,n_2,m) \text{ with } n_i > 0 \text{ and } s' = (n_1-1,n_2-1,m) \]
\[ p(s',s,e_i) = 1 \text{ when } s = (n_1,n_2,m) \text{ with } n_2 > 0 \text{ and } s' = (n_1,n_2-1,m) \]
\[ p(s',s,e_i) = 1 \text{ when } s = (n_1,n_2,m) \text{ with } \min(m-n_i,m-n_i) > 0 \text{ and } s' = (n_1,n_2,m-1) \]

(b) The table is as follows (new clock readings are in **boldface**.)

<table>
<thead>
<tr>
<th>( n )</th>
<th>( \zeta_n )</th>
<th>( X(\zeta_n) )</th>
<th>( C_{n,1} )</th>
<th>( C_{n,2} )</th>
<th>( C_{n,3} )</th>
<th>( C_{n,4} )</th>
<th>( e^* )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
<td>(1,1,1)</td>
<td><strong>4</strong></td>
<td>2</td>
<td>3</td>
<td>--</td>
<td>( e_2 )</td>
</tr>
<tr>
<td>1</td>
<td>2.0</td>
<td>(0,1,1)</td>
<td>2</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>( e_3 )</td>
</tr>
<tr>
<td>2</td>
<td>3.0</td>
<td>(0,0,1)</td>
<td>1</td>
<td>--</td>
<td>--</td>
<td><strong>4</strong></td>
<td>( e_1 )</td>
</tr>
<tr>
<td>3</td>
<td>4.0</td>
<td>(1,1,2)</td>
<td><strong>5</strong></td>
<td><strong>6</strong></td>
<td><strong>4</strong></td>
<td>3</td>
<td>( e_4 )</td>
</tr>
<tr>
<td>4</td>
<td>7.0</td>
<td>(1,1,1)</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>--</td>
<td>( e_3 )</td>
</tr>
</tbody>
</table>

(c) The computations are as follows:
\[ \hat{\mu} = \frac{1}{5}(1.0+3.0+7.0+5.0+4.0) = 4.0 \]
\[ s^2 = \frac{1}{4}((1.0-4.0)^2 + (3.0-4.0)^2 + (7.0-4.0)^2 + (5.0-4.0)^2 + (4.0-4.0)^2) = 5.0 \]
\[ z_p = 1.645 \quad \text{[0.95 quantile for } N(0,1)\text{]} \]
\[ \varepsilon = 0.1 \]
\[ n = \frac{z^2_p s^2}{\varepsilon \hat{\mu}^2} = \frac{(1.645)^2(5)}{(0.1)^2(4)^2} \approx 85, \]

so we need at least 85 runs.
2. Random number generation.

(a) The inverse of the cdf is given by \( F^{-1}_X(u) = (u / \beta)^{1/\alpha} \) for \( 0 \leq u \leq 1 \). Using inversion, generate \( U \) and return \( X = (u / \beta)^{1/\alpha} \).

(b) The pdf is \( f_X(x) = dF_X(x) / dx = \alpha \beta x^{\alpha-1} \) so the expected value of the distribution is
\[
\int_0^{\beta^{-1/\alpha}} x f_X(x) \, dx = \int_0^{\beta^{-1/\alpha}} \alpha \beta x^{\alpha} \, dx = \frac{\alpha}{\alpha + 1} \beta^{-1/\alpha}.
\] For \( \alpha = 2 \) and \( \beta = 4 \), the expected value equals 1/3. The given value is less than half of this amount, so the generator is untrustworthy.