Question 1.
Consider the mystery code below:

```java
int n = 7;
int m = 4;
int result = 1;
while (m > 0) {
    result = result * n;
    m = m - 1;
}
textbox.append(" + result + "\n");
```

(a) What is appended to the textbox?
(b) How many times is the \( (m > 0) \) condition evaluated?
(c) If the second line was \( \text{int } m = 0 \), then what would be written to the textbox?
(d) If the second line was \( \text{int } m = 0 \), then how many times would the \( (m > 0) \) condition be evaluated?
(e) Give a mathematical formula describing the relationship between the values initially assigned to \( m \) and \( n \) and the final value displayed for result.

Question 2.
A strand of bacterial messenger RNA can be encoded as a long string of letters representing the four possible bases: adenine (A), guanine (G), cytosine (C), and uracil (U). For example, one short RNA strand is:

```
UUACAUGUUAACGGACUAGGACGGAUGACACCAUAGAA
```

Only certain parts of an RNA strand are responsible for protein synthesis — the rest of the strand is unused while building proteins. The subsequences that are used for protein synthesis can be identified by the fact that they always begin with the triple of bases “AUG”, known as the start codon and end with the triple “UAG”, known as the stop codon. (Other stop codons exist, but we will ignore them.) The following is the example from above, rewritten to delineate the important subsequences from the junk:

```
UUAC | AUG | UUAACGGAC | UAG | GACGG | AUG | ACACCA | UAG | AA
```

Real strands of RNA contains many thousands of bases, and the sections encoding proteins are much longer. Suppose we wish to write a segment of Java code that extracts the sequences of bases between adjacent start and stop codons in a form that would make it easy to display the results. We will assume that before our code is executed, other parts of the program will associate the variable `rna` with a `String` composed only of the letters A, G, C, and U, and at the end our code should associate the variable `result` with a `String` composed of just those subsequences of the original String that were preceded by a start codon and followed by a stop codon. Each sequence extracted should be separated from the next sequence by a newline character. For example, if the variable `rna` is initialized as:

```
String rna = "UUACAUGUUAACGGACUAGGACGGAUGACACCAUAGAA";
```

then the desired code should associate the `String`

```
"UUAACGGAC\nACACCA"
```

with the variable `result`. For simplicity, we will assume that a sequence never ends in the middle of a coding region, so every “AUG” will be matched by a “UAG”.

Here is one, correct version of the desired code:

```
int start = rna.indexOf("AUG");
String result = "";
while (start >= 0) {
    int end = rna.indexOf("UAG", start);
    result = result + rna.substring(start + 3, end) + "\n";
    start = rna.indexOf("AUG", end + 3);
}
```
Below, you will find four code fragments that are similar but not identical to the fragment shown above. Each differs from above by a single line of code. The lines containing the changes are the ones identified by arrows. Some of these programs will function exactly like the program shown above. Others will work differently or not at all.

For each program, indicate whether:

- it works the **SAME AS THE ORIGINAL** program,
- a **SYNTAX ERROR** will be detected as soon as you compile the program (explain what line causes the error and why),
- the program will run, but its behavior is **DIFFERENT FROM THE ORIGINAL**, possibly because an error occurs at run time. If there will be an error, identify the line on which it will occur and explain the nature of the error. If no error occurs, explain what result would be when the loop terminates (or why there is no termination!) if the code fragment were run with string `rna` described above.

Start your answer with one of the three **bold** phrases above, so that if your explanation is unclear we can still decipher your intent.

a) ```java
start = rna.indexOf("AUG"); // <-----------------
String result = ""
while (start >= 0) {
    int end = rna.indexOf("UAG", start);
    result = result + rna.substring(start + 3, end) + "\n";
    start = rna.indexOf("AUG", end + 3);
}
```  

b) ```java
int start = rna.indexOf("AUG");
String result = ""
while (start >= 0) {
    int end = rna.indexOf("UAG", start);
    result = result + rna.substring(start, end) + "\n"; // <----------
    start = rna.indexOf("AUG", end + 3);
}
```  

c) ```java
int start = rna.indexOf("AUG");
String result = ""
while (start >= 0) {
    int end = rna.indexOf("UAG", start + 3); // <----------------
    result = result + rna.substring(start + 3, end) + "\n";
    start = rna.indexOf("AUG", end + 3);
}
```  

d) ```java
int start = rna.indexOf("AUG");
String result = ""
while (start >= 0) {
    int end = rna.indexOf("UAG", start);
    result = result + rna.substring(start + 3, end) + "\n";
    start = rna.indexOf("AUG", start); // <----------------
}
```  

**Question 3.**

Consider two computers, A and B that are connected by a wire that transmits at a rate of 10 Mbps (10 million bits/second). Suppose that A sends 1000 bits to B and the distance between A and B is 60Km.

(a) How long is it from when A transmits the first bit until B receives it, assuming that data travels at the speed of light (3 × 10^8 meters/second)? That is, what will be the propagation time for these bits?
(b) How long does it take A to put all 1000 bits onto the wire? That is, what will be the transmission time for these bits?

(c) How long is it from when A starts sending the bits until B has completely received it?

(d) How wide is a single bit being transmitted between these two computers? That is, how far can light travel in the time it takes to send a single bit?

**Question 4.**
Consider the signal represented by the following diagram:

(a) Assuming that signal is intended to transmit data encoded using on-off keying and that each of the intervals between the thin vertical lines represents the transmission of one bit, what sequence of binary digits does this signal represent? (Hint: Yes, the answer should be 21 bits long.)

(b) Now, continue to assume that on-off keying is being used and that the vertical lines represent bit times, suppose that the data has been grouped into 8-bit frames with one start bit preceding each frame. Under these assumptions, what data does the signal represent?

(c) Finally, suppose we drop the assumption about on-off keying and the assumption that the vertical lines represent bit times. Instead, assume that the Manchester encoding scheme is being used. In addition, continue to assume that the data is grouped into 8-bit frames and that each frame is preceded by a start bit (i.e., a one transmitted using Manchester encoding precedes the data). Under these assumptions, what data does the signal represent?

**Question 5.** Suppose that five computers named A, B, C, D, and E are waiting for a transmission to finish on an Ethernet. All transmit at once when the previous packet is finished and collide.

Table 1 on the last page shows a partial history of how these five computers might contend to obtain access to the network after this collision. In constructing this table, we have deliberately ignored variability in collision detection times so that if two computers choose the same random number when deciding to delay their next attempt to transmit, then they will begin their next transmissions at exactly the same time. We have also assumed the amount of time consumed by a collision is exactly equal to the time a computer will pause before transmitting if it randomly decides to wait for one slot.

In the table, we indicate which of computers A through D attempted to transmit in each time slot by showing the range of back-off times (e.g., “0..3”) from which each computer that is involved in a collision must choose its next back-off time and the random back-off value it actually selected (e.g., “:2”). We left the entries in the table for computer E empty.

Fill in the column for computer E in a way that minimizes the number of times E attempts to transmit under the assumption that no station successfully transmits a packet until round 8 when computer C transmits alone. Show which slots E transmits in by indicating the range of back-off values it must choose between and the back-off value it selects after each collision in which it is involved. There are several possible solutions. You need only provide one. If you need an extra copy of the table for scrap work, you can access the PDF for this assignment on the course web page.
<table>
<thead>
<tr>
<th>SLOT</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 0</td>
</tr>
<tr>
<td>1</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 1</td>
</tr>
<tr>
<td>2</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 3</td>
</tr>
<tr>
<td>3</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 2</td>
</tr>
<tr>
<td>4</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 3</td>
</tr>
<tr>
<td>5</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 6</td>
</tr>
<tr>
<td>6</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 2</td>
</tr>
<tr>
<td>7</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 7</td>
</tr>
<tr>
<td>8</td>
<td>backoff range 0..1: delay used: 0</td>
<td>backoff range 0..3: delay used: 0</td>
<td>backoff range 0..1: delay used: 1</td>
<td>backoff range 0..3: delay used: 3</td>
<td>delay used: 7</td>
</tr>
</tbody>
</table>

C transmits alone!