Exam 1—Take Home Component

Assigned: Monday, July 27, 2009
Due: by electronic submission through Collab by 1 pm Weds., July 29, 2009

Value: This is worth 20% of your Exam 1 score. Exam 1 is worth 15% of your final grade.

Instructions:

Part two (one programming question). This part of the examination is pledged. All work must be your own. This take-home component of the exam is open book, open notes, and you may freely examine all class materials and your own homeworks and notes. You may use the Internet to access your own information or other information that is freely available and intended for public access. You may NOT utilize the class notes, class homeworks, or class files authored by any person other than yourself or your homework partner. You may not consult with any person other than the course instructor and teaching assistant.

PLEDGE: By submitting my solution through Collab, I certify that I have fully complied with the above pledge requirements this part of the exam.
Use a client-side scripting language (e.g., JavaScript) to implement a one-time pad encryption service. Generate a web form that allows a user to input exactly eight hexadecimal digits (the plaintext). Allow the user to enter his own secret key (exactly 8 hex digits) and click on a “Compute Ciphertext” button, after which your program error-checks the input and, if correct, computes and displays the resulting ciphertext (exactly 8 hex digits). Your program's display should look similar to this one.

<table>
<thead>
<tr>
<th>ONE-TIME PAD ENCRYPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>By John Q. Public (jqp7fg)</td>
</tr>
<tr>
<td>Enter plaintext (8 hex digits)</td>
</tr>
<tr>
<td>Enter secret key (8 hex digits)</td>
</tr>
<tr>
<td>Ciphertext (8 hex digits)</td>
</tr>
<tr>
<td>Compute ciphertext</td>
</tr>
</tbody>
</table>

Requirements:

- Display the program title, your name, and your email as shown. If your name is not displayed, your grade is zero.
- All operations use exactly 8 hex characters. If the user enters more or fewer than 8 characters, or if the input characters are not legal hex characters, use an alert to tell the user about his error and make him correct it.
- The user will enter his plaintext and his secret key and then click on “Compute ciphertext.”
- When the button is clicked, error-check the user’s input and proceed only if the input is correct. If correct, compute and display the appropriate ciphertext.

Hints and Help:

- You might find it simplest to process the 8-digit hex numbers one character at a time.
- JavaScript has a bitwise XOR operator: i.e. $x \oplus y$ where $x$ and $y$ are numbers.
- Since this is a symmetric method, make sure your get the original plaintext when you put the ciphertext back in as the plaintext textbox.
- We don’t want you spending too much time on the web or in books figuring out about numeric conversions etc., so consider the following JavaScript code:

```javascript
var hexStr = "0xE";
var hexStrAsInt = Number(hexStr); // converts from hex because of the 0x
var intVal = 10;
var intValAsHex = intVal.toString(16); // toString takes a base
var xorInts = hexStrAsInt ^ intVal; // ^ is bitwise XOR
var xorIntsAsHex = xorInts.toString(16);
document.writeln("<BR>hexStr= " + hexStr);
document.writeln("<BR>hexStrAsInt = " + hexStrAsInt);
document.writeln("<BR>intVal= " + intVal);
document.writeln("<BR>intValAsHex = " + intValAsHex);
document.writeln("<BR>xorInts= " + xorInts);
document.writeln("<BR>xorIntsAsHex = " + xorIntsAsHex);
```