UCLA
Computer Science Department
Fall 2011

Student Name and ID: _____

CS144 Final: Closed Book, 2 hours

(IMPORTANT PLEASE READ **):**

- There are 4 + 1 problems on the exam to be completed in 2 hours. *You should look through the entire exam before getting started, in order to plan your strategy.*
- You may use two sheets of double-sided notes during exam. You are also allowed to use a calculator. Attach extra pages as needed. Write your name and ID on the extra pages.
- *Simplicity and clarity of your solutions will count.* You may get as few as 0 points for a problem if your solution is far more complicated than necessary, or if we cannot understand your solution.
- If you need to make any assumption to solve a question, please *write down your assumptions*. You may also want to write down how you arrived at your answer step by step to get partial credit.

Problem	**Score**	
1	20	
2	10	
3	20	
4	15	
extra	4	
Total	69	

Problem 1: 20 points

Answer YES or NO for each of the following questions. Be careful with your answers since you get 2 points for every correct answer, -1 point for every incorrect answer.

1. Consider two web pages P_1 and P_2. P_1 contains only the word w_1 three times. P_2 contains the word w_1 three times and the word w_2 once. Under the tf.idf vector representation and the cosine similarity metric, P_1 will be ranked the same as P_2 for the query w_1.
   ANSWER:
   `NO`

2. Consider a text consisting of n characters.

   (a) The above text of n characters maps to a sequence of n unicode code points.
   ANSWER:
   `YES`

   (b) Assuming the text has ASCII characters only, its UTF-16 encoding is longer than UTF-8 encoding (in terms of the number of bytes).
   ANSWER:
   `YES`

   (c) Assuming the text consists of Chinese characters only (whose code points are between U+4E00 and U+9FBC), its UTF-16 encoding is longer than UTF-8 encoding (again, when measured by the number of bytes).
   ANSWER:
   `NO`

3. Suppose an attacker obtains the private key for a central Certificate Authority.

   (a) The attacker can eavesdrop on SSL connections to a legitimate banking web site.
   ANSWER:
   `NO`

   (b) The attacker can impersonate the bank.
   ANSWER:
   `YES`

4. Suppose that the user A is sending a large video file (of size 700MB) to the user B using SSL.

   (a) The video file is encrypted by the RSA algorithm using B's secret key.
   ANSWER:
   NO

   (b) Using SSL for confidentiality is computationally more expensive than using a symmetric cipher (say, AES) by several orders of magnitude.
   ANSWER:
   NO

5. When combining encryption and compression, it is a better idea to encrypt and then compress than to compress and then encrypt.
   ANSWER:
   NO

6. A perfect hashing scheme is often used for NoSQL datastores in order to minimize data reorganization when a new physical node is added.
   ANSWER:
   NO

Problem 2: 10 points

Using Firefox, you are trying to access the Web page, `http://cs144.edu/P1.html`, whose content is shown below:

```
<html>
<head>
  <style type="text/css">@import "P1.css";</style>
  <script type="text/javascript">
    function InitPage() {
       var request = new XMLHttpRequest();
       request.open("GET", "N2.html");
       request.onreadystatechange = function() {
          if (request.readyState == 4) {
             document.getElementById("link").innerHTML = "Next Page";
          }
       }
       request.send(null);
    }
  </script>
  <title>P1</title>
</head>
<body background="Background-Image.gif" onLoad="InitPage();">
  <p id="i1" class="Image"><img src="Img.gif"/><br/>Caption 1</p>
  <p id="i2" class="Image"><img src="Img.gif"/><br/>Caption 2</p>
  <p><a href="N2.html"><span id="link">Link</span></a></p>
</body>
</html>
```

1. How many HTTP requests does the browser *have to* issue to the server in order to completely render the page? Assume that the browser never visited the Web site before. Explain your answer briefly in no more than 50 words. You will get zero point without any explanation.

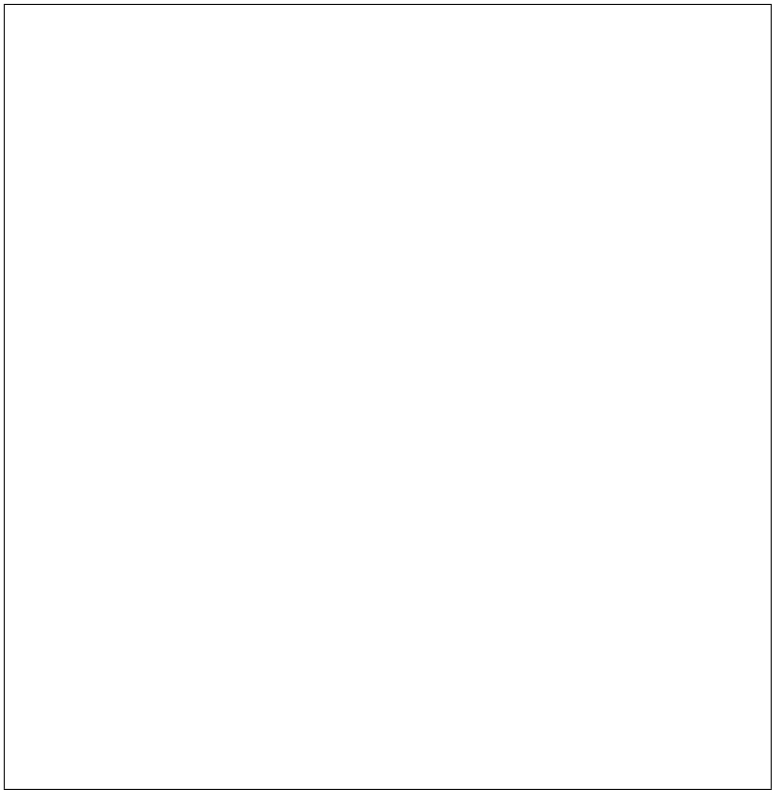
ANSWER:

5.
P1.css, P1.html, Background-Image.gif, Img.gif, and N2.html (AJAX)
Note: partial points will be provided to those who have a correct answer but provide wrong explanation(s).

2. Assume that `Background-Image.gif` is a blank image of solid white color and `Img.gif` is an image of black square whose width/height is about the same as the width of the text "Caption 1". Assume that the following is the content of `P1.css` file.

```
p {
  font-family: "Tahoma";
  color: blue;
  text-align: center;
}
.Image {
  font-family: "Times New Roman";
  color: yellow;
}
p.Image { color: black; }
#i1 {
  font-family: "Courier New";
  text-align: left;
  color: red;
}
```

In the space provided below, roughly draw the content of the page once it is completely rendered by the browser. For each line of the text in the rendered page, indicate the font and the color of the text.



ANSWER:

```
[]  <- black square
Caption 1   <- red, Courier New

                []
             Caption 2   <- black, Times New Roman

             Next Page  <- blue, Tahoma, underlined
```

Note: partial points will be provided based on correctness.

Problem 3: 20 points

You are developing a shopping Web site for a Chinese company. Since the majority of its users are Chinese, you set up your database to use the Chinese GBK character encoding. Under GBK, a byte in the range 0x00-0x7F represents the corresponding ASCII character. A byte in the range 0x80-0xFE indicates the first byte of a two-byte Chinese character, followed by a second byte in the range of 0x00-0xFF. For example, 0x27 and 0x5C represent ' and \, respectively, and 0x815C represents 乘.

Now consider the following Java servlet code fragment used for user authentication:

```
String username = request.getParameter("username");
String password = request.getParameter("password");
ResultSet result = statement.executeQuery(
                     "SELECT * FROM Users WHERE username='" + username +
                     "' AND password='" + password + "';");
if (result.next()) { /* Successful login */ }
else               { /* Login denied */ }
```

1. Assuming that your Java servlet code is accessible at `https://shop.com/login` through the HTTP GET method, write down how a user's login request URL will look like if the user name is `hacker` and her password is `malicious`.

   ANSWER:
   `https://shop.com/login?username=hacker&password=malicious`

2. In no more than 50 words, explain why a URL where `username` is set to `"' OR 1 = 1; --"` will allow an attacker to login the system without knowing a correct username and password pair.

   ANSWER:

   The SQL statment becomes

   ```
   SELECT * FROM Users WHERE username=' ' OR 1 = 1 -- ' AND password...
   ```

   Justification:
   (1) the single quote symbol acts as an end of a SQL condition (username =).
   (2) adding "OR 1=1" make the combination of two conditions always be true.
   (3) the semicolon symbol ";" ends the SQL statement.
   (4) "--" comments out the rest of statements(the password part).

3. To avoid the above attack, you "escape" all user inputs by scanning the input parameters and adding the byte 0x5c (the ASCII character \) whenever you encounter the byte 0x27 (the ASCII character ’). For example, the user input `"a’"` will be escaped to the string `"a\’"`. Explain why this prevents the attack from Part 2 in no more than 50 words.

   ANSWER:

   By escaping the single quote, the WHERE conditions becomes:

   ```
   username=’ \’ OR 1 = 1 --’ AND password...
   ```

   Namely, the username becomes " \’ OR 1 = 1 --" and the WHERE conditions result in FALSE.

4. Unfortunately, your solution in Part 3 may still be vulnerable to a SQL injection attack because your database uses the Chinese GBK encoding. Write down a `username` byte sequence that will allow an attacker to login to the system without knowing the correct username and password pair.

   ANSWER:

   Suppose that we had a username "0x81 0x27", after escaping, the byte sequence becomes "0x81 0x5c 0x27". The first two bytes become this Chinese character 乘 and the third byte becomes a single quote. For the same reasons as described in the solution of the first sub-problem, we has a SQL injection now.

   Note: some other byte sequence might be an acceptable answer. For example, the first byte is 0x80-0xFE, and the second byte is 0x27. However, if you did not write down

a correct **byte sequence**, you will get no points. Partial points will be assigned to those who wrote down a correct byte sequence but provide a wrong justification. (If you provide no justification/explanation, it's fine because the question does NOT ask you to explain. But if you somehow provide a wrong one, you will lose some points.)

Problem 4: 15 points

We need to compute the PageRank values of 1 billion Web pages. In constructing the PageRank equation, we assume the PageRank definition that we learned in the class, where the user never performs a random jump (i.e., the damping factor d is 1 for any page p_i).

1. Write down the PageRank equation for the page p_i. In writing down the equation, you may assume $P(p_i)$ is PageRank of the page p_i, $OL(p_i)$ is the set of all pages that p_i has a link to, $IL(p_i)$ is the set of all pages that links to p_i, and $|S|$ represents the size of the set S. If you need further symbols and notation for other information, please state the meaning your symbols after the equation. (5 points)

   ANSWER:

   $P(p_i) = \sum_{p_k \in IL(p_i)} \frac{P(p_k)}{|OL(p_k)|}$

   `Note: if two or more TAs says that they need the test-taker's further explanation to understand your formula, you may get no points. See the third point in the Page 1 (paragraph: IMPORTANT MUST READ).`

2. We decided to compute the PageRanks of the billion pages *iteratively* after initializing all PageRank values to 10^{-9}. In order to run the PageRank computation on a cluster of machines, we also decided to perform the PageRank computation as a sequence of Map/Reduce operations. More precisely, each iteration of PageRank computation should be performed as a pair of Map and Reduce operations, where the input to the Map function is the sequence of `(pageid, old_pagerank)` pairs computed from the previous iteration and the output from the Reduce function should be the sequence of `(pageid, new_pagerank)` pairs after one iteration of PageRank computation is done. You can assume that all `old_pagerank` values are set to 10^{-9} in the initial input.

   Fill in the spaces in the Map and Reduce functions in the following page, so that, together, they perform one iteration of PageRank computation. You may assume that `Links.GetOutLinks(n)` returns an array that contains the IDs of all pages that the page `n` links to and `Links.GetInLinks(n)` returns an array that contains the IDs of all pages that link to the page `n`. You can also get the size of the array `A` with `A.length` and access an element of `A` with `A[i]` where `i` is an integer between `0` and `A.length-1`.

Write your code using Java syntax, but note that the exact syntax of your code is not as important as the logic and the clarity of your code. You may get full points even if your code may have minor syntactic errors, but it is important to make your code clean and succinct. You may lose as low as zero point if we cannot understand your code or if you have unnecessary line(s) or logic(s) in your answer. (10 points)

```
// key: integer value representing the page id
// value: double value representing the old pagerank value of the page
// output: output file. You can write the output (key, value) pair
//         by calling output.collect(key, value). Note that (key, value) pair
//         for the output may or may not be the same as the input key
//         and value.
void Map(int key, double value, MapOutput output)
{









}

// key: an output key from Map operation
// values: the set of all values with the same output key from Map operation
// output: output file. You can write the output (key, value) pair
//         by calling output.collect(key, value).
void Reduce(int key, double[] values, ReduceOutput output)
{









}
```

ANSWER:

(1) MAP part

```
int[] outPR = Links.getOutLinks(key);
if(outPR.length==0) return;

int temp = value/outPR.length;
for (int i=0; i < outPR.length; i++) {
  output.collect(outPR[i], temp);
}
```

(2) REDUCE part

```
double new_pr = 0.0;
for (int i=0, i < values.length; i++) {
  new_pr += values[i];
}
output.collect(key, new_pr);
```

Note: Some (minor) syntax error(s) may be okay at our discrimination. In addition, you may lose some minor point(s) if your implementation is clearly inefficient with regards to time/space complexity. For example, putting `value/outPR.length` in the for loop.

Extra Credit Problem: 4 points

1. Who is the instructor of this class? Provide a brief description of the instructor. (1 point)

2. Write down any topic(s) that you liked in the class. (1 point)

3. Write down any topic(s) that you disliked in the class. (1 point)

4. Write down any topic(s) that you hope the class will cover in the future. (1 point)