Exam Preparation for CS 144 – Web Applications

John Cho Luis Ángel Zijun Xue Ivy Song

March 13, 2015

1 Basic Standards

1. Provide a list of (at least) 4 **XML** entities.
2. What's the difference between **XML**, **HTML**, and **XHTML**?
3. What are the 3 important changes that **HTML 5** introduced to **HTML 4.01**?
4. Consider the following **CSS** definitions:

```
* {
   margin: 0px;
   padding: 0px;
}

p {
   margin: 10px;
   border: 1px solid #0000ff;
   color: yellow;
}

.p {
   margin: 20px;
   background-color: rgb( 255, 0, 0 );
}

#p {
   margin: 5px;
   color: white;
}
```

Also, consider this segment of **HTML**:

```
<div>Text1</div>
<p>Par1</p>
<p class="p">Par2</p>
<p id="p">Par3</p>
<p class="p" style="background-color:black;">Par4</p>
```

(a) What is the margin of **Text1**?
(b) What is the margin of **Par1**?
(c) What is the margin of **Par2**?
(d) What is the text color of **Par3**?

(e) What is the background color of **Par4**?

In general, make sure you know how to identify the different levels of **CSS** when they are applied to **HTML** elements (i.e. *class*, *identifier*, *top-level element style*, and *local-level tag style*).

5. As for web forms, when do we use **GET** and **POST** in the **method** attribute?
6. If I want to know all the proxies that my **HTTP** request went through, which of the following methods should I use: **GET**, **POST**, **HEAD**, or **TRACE**?
7. Consider the following **HTTP** request:

```
GET index.html HTTP/1.1
host: www.cs.ucla.edu
\r\n
\r\n
```

(a) What's the content of the **HTTP** request header?

(b) Why do we need to include **host** in our request, if we are already contacting the server?

(c) Is it really necessary the second **r****t**? What's its function in this request?

2 XML, DTD, and XML Schema

1. Why do you need *namespaces* in **XML**?
2. Consider the following:

```
<book xmlns="http://manybooks.com"
   xmlns:a="http://otherbooks.com"
   ISBN="123">
   <title>It</title>
   <a:author>Stephen King</author>
</book>
```

(a) What's the default namespace?

(b) What's the namespace of **<title>**?

(c) What's the namespace of **<book>**?

(d) What's the namespace of **ISBN**?

3. Does the order of the children of a given element in a **DTD** definition matter? What about the attribute listings?
4. Recall how to build a **DTD** tree.
5. If we defined **<!ATTLIST author aid ID #REQUIRED>** and **<!ATTLIST book bid ID #REQUIRED>**, what is the constraint for the attributes **aid** and **bid**? Can we have a book and an author with the same identifier?
6. What are the advantages of **XML Schema** over **DTD**?
7. Check out the **XPath** lab in our course website.
8. Know how to transform **DTD** and **XML Schema** into a relational database.

3 Encoding

1. What is **MIME** and where do we use it?
2. If I want to generate a request that contains mixed content objects in the body, which **MIME** type shoud I use?
3. What's the difference between *code point* and *encoding*?
4. What is the main difference between **ASCII** and **ISO-8859-1**?
5. What is the main difference between **UTF-8** and **UTF-16**?
6. Explain: what is **UNICODE**?
7. What are the **UTF-8** encodings for the code points U+3401, U+0277, and U+00C6?

4 Database Normalization

1. What is a *functional dependency*?
2. What is the difference between a *key* and a *superkey*? Give an example.
3. What is a *trivial* functional dependency? Give an example.
4. Given the following functional dependencies:

```
AB -> C
 C -> D
 D -> A
```

   What are the *closures* of **AB**, **C**, and **D**?
5. What is the *Boyce-Codd* Normal Form? Make sure you know the algorithm to identify and normalize a relation that is not in *BCNF*.

5 Information Retrieval System

1. What are the four layers of a Web Server? Give examples of instances or applications running at each layer.
2. What is an *Information Retrieval System*?
3. Explain the fundamentals of the *Boolean Model*. Why is it called *Boolean*?
4. An *Inverted Index* is a system made of two data structures, which are they? Provide an example of an Inverted Index with its two main components.
5. The size of the *Posting Lists* is negligible in comparison to the *Lexicon*... or is it the other way around?
6. Explain with your own words what *Precision* and *Recall* are. How can we achieve the maximum Precision and Recall?
7. What is the *Vector – Space Model*, and how is it different to the Boolean Model?
8. What is **TFIDF**? Why do we use it in the Vector – Space Model?
9. Given a query q, how do you rank the documents d_i under the Vector – Space Model?
10. How can you improve the efficiency of the Vector – Space Model by using an Inverted Index?
11. What is the *Link – Based Model*, and how is it different to the Vector – Space Model?
12. Provide the mathematical expression for the *Page – Rank Algorithm*.

6 Spatial Indexing

1. Why do we need *Spatial Indexing*?
2. Explain with your own words how a *Grid File* works.
3. What is (are) the disadvantage(s) of the Grid File?
4. Explain with your own words: what is a *Quadtree*?
5. What is (are) the disadvantage(s) of the Quadtree?
6. What is an R – *Tree*, and how does it work?

7 Web Services

1. What is a *Web Service*?
2. Make sure you know how to build a **SOAP** request and response messages.
3. Which are the 5 elements of a **WSDL** definition, and how are they related? Make sure you can read and understand a **WSDL** definition.
4. How do **SOAP** and **WSDL** couple together?
5. What is **REST**?
6. Compare **SOAP** & **WSDL** against **REST**. What are the advantages and disadvantages of one over the other?
7. Recall that in *Distributed Transactions* the *2 – Phase Commitment Protocol* is one way to ensure atomicity and synchronization. Draw the algorithm's flow diagram, and add a **time-out** at the *Waiting* state of the *Coordinator*.
8. What is the *Asynchronous Transaction* method, and how is it different to the 2 – Phase Commitment Protocol?

8 MVC

1. Explain what *MVC* is and give example applications or software operating on each of the three components.
2. Review all basic **JavaScript** concepts. How do you create a function? How do you create a "class" and generate instances out of it?
3. When do we use **var** within **JavaScript**, and when not?
4. How do you include **JavaScript** in a webpage?
5. What is **AJAX**? Why is it advantageous for **AJAX** being "asynchronous"?
6. Indicate what the following **JavaScript** segment does.

```
var request = new XMLHttpRequest();
request.send( null );
request.open( \GET", "index.html" );
request.onreadystatechange = function(){
   alert("It's here!");
}
```

7. What is **JSON**?
8. Assume that we have retrieved the string `'{"Place":1,"Person":{"Name":["Alexandra","Anahis"]}}'` from a server response. How do we extract `"Anahis"` from the response in **JavaScript**.
9. Explain the *Same Origin Policy*.
10. What is a web **Cookie**? Provide an example.
11. What is **Cookie** *Poisoning* and *Theft*?
12. How do **Cookies** couple with **Sessions**? Why do we use **Sessions**? Where does a **Session** reside: in the client or the server?
13. Draw a diagram that illustrates how **Sessions** and **Cookies** are used in the *multi – website* loging-in Google's system.

9 Security

1. Briefly, define the following security issues:
   (a) Distributed Denial of Service.
   (b) Defacement.
   (c) SQL/Command Injection.
   (d) Spoofing/Phishing.
   (e) Man in the Middle.
   (f) Pharming/DNS Poisoning.
   (g) Buffer Overflow.
2. What are the **4 Guarantees** that we expect from web transactions?
3. Explain the *Shannon's Perfect Secrecy* principle.
4. What is a cypher?
5. How does *One – Time Padding* (OTP) work?
6. What is the disadvantage of OTP?
7. Explain how *Symmetric Encryption* works.
8. What are the disadvantages of Symmetric Encryption?
9. How many keys do we need to keep when there is communication between n entities using Symmetric Encryption? Why is all of this necessary?
10. What are the three properties of *Asymmetric Encryption*?
11. Explain the **RSA** algorithm. What is the *RSA Problem*? What is the *Large – Number Factorizatoin Problem*?
12. How do we ensure *Confidentiality* in the Asymmetric Encryption? How does **SSL** work?
13. How do we ensure *Integrity* of a message by using Asymmetric Encryption? What is the *Signature Scheme*? Which key is used to sign a message: the public key or the private key?
14. How do we ensure *Authentication* under Asymmetric Encryption? What is a *Challenge Message*?
15. How many keys do we need in order to keep confidentiality between n entities using Asymmetric Encryption?
16. What is a *Certificate Authority*, and how does it participate in the Asymmetric Encryption scheme?

10 Common Vulnerabilities

1. What is *Buffer Overflow*? How can you protect against it?
2. What is *Client – State Manipulation*? How can you protect against it?
3. What is *SQL Injection*? How can you protect against it?
4. What is *Command Injection*? How can you protect against it?
5. What is *Cross Site Scripting* (XSS)? How can you protect against it?
6. What is *Cross Site Request Forgery* (XSRF)? How can you protect against it?

11 Scalability

1. Recall how we can approximate our server capacity.
2. What is *Scale Up*?
3. What is *Scale Out*?
4. Given the four layers in a Web Server, which ones can easily Scale Out, and which ones cannot?
5. What is a *Load Balancer*? Briefly, explain how *DNS Round Robin* works.
6. How do we scale the Storage layer when we have just *Read Only* transactions?
7. How do we scale the Storage layer when we have just *Local Read/Write* transactions?
8. How do we scale the Storage layer when we have *Global Read/Write* transactions?
9. What is the *Storage Area Network* (SAN)?
10. What is a *Distributed File System*?

12 NoSQL

1. Explain the *Byzantine General Problem*, and how it relates to maintaining synchronization and consistency in a distributed system.
2. What are the following types of consistency?
   (a) RYOW.
   (b) Session Consistency.
   (c) Monotonic Read.
3. How does the *Key – Value Store* NoSQL API work? What are its disadvantages?
4. How does the *Column – Oriented Store* NoSQL API work? What are its disadvantages?
5. How does the *Document – Oriented Store* NoSQL API work?
6. What is *Consistent Hashing*? What is a potential problem with it and how do we address it?

13 Caching

1. What is *Browser Caching*?
2. What is *Network Caching*?
3. What is *Memcached*?