Question 1

Question 1a

Jehan chooses to set a biometric password for her mobile device, instead of a personal identification number (PIN).

i. State what is meant by a biometric password.

- Biometric Password: A biometric password refers to a method of securing a device using a unique biological characteristic of the user, such as a fingerprint, facial recognition, iris scan, or voice recognition. These characteristics are unique to individuals and difficult to replicate.

ii. Give three (3) reasons why a biometric password is more secure than a PIN.

1. Uniqueness: Biometric traits are unique to each individual, making it extremely difficult for unauthorized users to replicate or guess.
2. Non-transferable: Unlike PINs or passwords, biometric data cannot be easily shared or stolen.
3. Convenience and Speed: Biometric authentication is often quicker and more convenient than entering a PIN, reducing the likelihood of shoulder surfing or other forms of observation-based attacks.

Question 1b

An example of a Uniform Resource Locator (URL) is:

```
<html>
<head><title>404 Not Found</title></head>
<body>
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.24.0 (Ubuntu)</center>
</body>
</html>
```

Identify the three (3) parts that make up this URL.

1. Part 1: Protocol (https://)
    - **Description**: Specifies the protocol used to access the resource on the internet. 'https' indicates a secure version of HTTP.
2. Part 2: Domain Name (www.open.ac.mu)
    - **Description**: Identifies the server where the resource is hosted. This is a human-readable address that is translated to an IP address by the DNS.
3. Part 3: Path (/index.htm)
    - **Description**: Specifies the exact resource or file to be retrieved from the server.

Question 1c

Devices connected to the Internet have IP (Internet Protocol) addresses. Three IPv4 addresses are given. Indicate whether each address is valid or invalid. Justify your answer.

i. Address 1: 3A.21.2H.1

- **Invalid**: An IPv4 address consists of four decimal numbers separated by dots, each ranging from 0 to 255. Here, '3A' and '2H' contain non-decimal characters.

ii. Address 2: 299.53.2.2

- **Invalid**: Each segment of an IPv4 address must be in the range of 0 to 255. The number '299' exceeds this range.

iii. Address 3: 192.2.1.0

- **Valid**: This address consists of four decimal numbers within the valid range (0-255).

Question 1d

The jQuery syntax is used for selecting HTML elements and performing some action on the element(s). Briefly explain the function of the following jQuery syntax:

i. `$("p").hide()`

- **Function**: Hides all `<p>` (paragraph) elements on the page.

ii. `$("p").remove()`

- **Function**: Removes all `<p>` (paragraph) elements from the page.

iii. `$(".date").hide()`

- **Function**: Hides all elements with the class `date`.

iv. `$("#date").remove()`

- **Function**: Removes the element with the ID `date`.

v. `$(this).hide()`

- **Function**: Hides the current element that triggered the event.

Question 1e

Common Gateway Interface (CGI) is a standard way for web servers to interface with executable programs installed on a server that generate web pages dynamically. Elaborate on some uses of Common Gateway Interface (CGI).

1. **Dynamic Content Generation**: CGI scripts can generate dynamic content based on user input or other variables, such as displaying search results or customized user data.
2. **Form Processing**: CGI scripts are commonly used to process form data submitted by users, such as registration forms, feedback forms, or online surveys.
3. **Database Interaction**: CGI can be used to interact with databases to retrieve, update, or delete data based on user queries or actions.
4. **E-commerce Transactions**: CGI scripts can handle secure transactions for online shopping, including processing payments and generating order confirmations.
5. **Server-side Scripting**: CGI enables server-side scripting, allowing for complex server-side logic to be executed, such as generating reports or performing backend calculations.

Question 2

Question 2a

In a Perl script, anything appearing after a `#` on a line is a comment. Explain the need for comments in writing Perl script and outline what needs to be commented for every script a developer creates.

Need for Comments:

- Readability: Comments improve the readability of the code by explaining what the code does, making it easier for others (or the original author at a later time) to understand.
- Maintenance: Comments aid in maintaining the code by providing context, which helps in debugging and updating the code.
- Collaboration: In collaborative environments, comments help team members understand each other's code, facilitating smoother teamwork.

What to Comment:

- Purpose of the Script: A brief description at the beginning of the script explaining its purpose and functionality.
- Complex Logic: Detailed explanations of complex or non-obvious logic within the code.
- Parameters and Returns: Description of function parameters, return values, and any side effects.
- Important Variables: Explanation of important variables and their roles within the script.
- Dependencies: Information about any external modules or dependencies used in the script.

Question 2b

A teacher is writing examination papers on a laptop computer. The computer is connected to the internet. The teacher is concerned about the security and privacy of the papers.

i. State the difference between the security of data and the privacy of data.

- **Security of Data**: Refers to the protection of data from unauthorized access, corruption, or theft. It involves implementing measures such as encryption, firewalls, and access controls to safeguard data integrity and availability.
- **Privacy of Data**: Refers to the protection of personal or sensitive information from being disclosed to unauthorized individuals. It involves ensuring that data is collected, stored, and shared in compliance with privacy laws and regulations, and that individuals' rights to control their personal information are respected.

ii. **Identify and describe two (2) threats to the data. Identify one (1) security measure to protect against each threat. Each security measure must be different.**

1. **Threat 1: Phishing Attacks**
    - **Description**: Phishing involves tricking the user into divulging sensitive information, such as login credentials, by pretending to be a legitimate entity.
    - **Security Measure**: Implement email filters and user education programs to recognize and avoid phishing emails.
2. **Threat 2: Malware**
    - **Description**: Malware includes viruses, trojans, and ransomware that can infect the computer and compromise or steal data.
    - **Security Measure**: Install and regularly update antivirus software to detect and remove malicious programs.

Question 2c

The ABSA online portal has the following display when opting for Account Activity for a particular customer. It uses methods of ordered and unordered list.

Write HTML codes to display the above unordered list items and rewrite the HTML codes to make it an ordered list.

Unordered List:

```
<ul>
  <li>Transaction Days</li>
  <li>Current Period</li>
  <li>For Date Ranging From</li>
</ul>
```

Ordered List:

```
<ol>
  <li>Transaction Days</li>
  <li>Current Period</li>
  <li>For Date Ranging From</li>
</ol>
```

Question 2d

Using examples where appropriate, elaborate on how AJAX works when requesting and responding to an event.

AJAX (Asynchronous JavaScript and XML):

- **Request**: AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that parts of a web page can be updated without reloading the entire page.
- **Example**: A user fills out a form and submits it. Instead of the entire page refreshing, an AJAX request is sent to the server to process the form data.
- **Code Example**:

```javascript
// JavaScript code to send an AJAX request
function sendData() {
    var xhr = new XMLHttpRequest();
    xhr.open("POST", "submit_form.php", true);
    xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4 && xhr.status == 200) {
            document.getElementById("response").innerHTML = xhr.responseText;
        }
    };
    var formData = "name=John&age=30";
    xhr.send(formData);
}
```

- **Explanation**: When the `sendData` function is called, it creates a new XMLHttpRequest object, sets up a POST request to `submit_form.php`, and sends form data to the server. When the server responds, the response text is displayed in an HTML element with the ID `response`.
- **Benefits**:
    - **Improved User Experience**: Provides a more dynamic and responsive user experience by updating parts of the web page without a full reload.
    - **Efficiency**: Reduces server load and bandwidth usage by only transferring necessary data.

Question 3

Question 3a

A web page includes HTML and JavaScript code.

```html
<!DOCTYPE html>
<html>
<body>
    <h1>Calculate area of a triangle:</h1>
    <form name="Triangle">
        <p>Base: <input type="number" name="B" value=""></p>
        <p>Height: <input type="number" name="H" value=""></p>
        <button onclick="area()">Calculate</button>
    </form>
    <script>
        function area() {
            var base = document.forms["Triangle"]["B"].value;
            var height = document.forms["Triangle"]["H"].value;
            if (base == "" || height == "") {
                alert("Both values must be entered");
            } else {
                var area = 0.5 * height * base;
                alert("The area is: " + area);
            }
        }
    </script>
</body>
</html>
```

i. Give the three (3) identifiers used in the JavaScript code.

1. `base`
2. `height`
3. `area`

ii. State the purpose of the code on line 08.

- **Purpose**: The code on line 08 (`onclick="area()"`) assigns an event handler to the button. When the button is clicked, it triggers the `area` function to calculate the area of the triangle.

iii. The page is loaded and the values 2 and 8 are entered. State the output when the calculate button is clicked.

- **Output**: The alert box will display: `The area is: 8`

iv. State the meaning of the II operator in line 15 of the code.

- **Meaning**: The `||` operator is the logical OR operator. It checks if at least one of the conditions is true. In this case, it checks if either `base` or `height` is an empty string.

v. Data validation has been used in line 15 of the JavaScript code. Identify the type of data validation used in line 15.

- **Type of Data Validation**: Presence check. It ensures that both `base` and `height` values are entered (not empty).

Write the MISSING CODES at line 9, 17 & 24

- **Line 9**: `// This line should be left empty as there is no code needed here.`
- **Line 17**: `var area = 0.5 * height * base;`
- **Line 24**: `</script>`

Question 3b

jQuery selectors select elements to add behavior to those elements. Comment on the above statement.

- **Comment**: jQuery selectors are used to select HTML elements based on their id, class, tag, or attributes. Once selected, jQuery can be used to add behaviors such as event handling (e.g., clicks, hovers), animations, and modifications to the element's content, style, or attributes. This allows developers to easily manipulate the DOM and enhance interactivity on web pages.

Question 3c

A teacher uses a relational database, RESULTS, to store data about her students and their test results.

i. A student with ID JM0701 has obtained a mark of 92 for a Computer Science test paper with ID CS2021. Write an SQL script to add this data to the database.

```sql
INSERT INTO STUDENT_TEST (StudentID, TestID, Mark) VALUES ('JM0701', 'CS2021', 92);
```

ii. Write SQL script to display the Student ID, mark and maximum marks for all tests with the topic of 'Programming'.

```sql
SELECT STUDENT_TEST.StudentID, STUDENT_TEST.Mark, TEST.MaxMarks
FROM STUDENT_TEST
JOIN TEST ON STUDENT_TEST.TestID = TEST.TestID
WHERE TEST.Topic = 'Programming';
```

Question 4

Question 4a

Websites can be created using HTML structure and presentation. State what is meant by HTML structure and presentation. Give an example of each in your answer.

HTML Structure:

- **Definition**: HTML structure refers to the semantic arrangement of the content on a webpage using HTML elements. It defines the organization and hierarchy of the webpage content, ensuring that it is logically and meaningfully structured.
- **Example**: Using `<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<footer>` tags to define different parts of a webpage.

```html
<!DOCTYPE html>
<html>
<head>
    <title>Sample Webpage</title>
</head>
<body>
    <header>
        <h1>Website Title</h1>
    </header>
    <nav>
        <ul>
            <li><a href="#home">Home</a></li>
            <li><a href="#about">About</a></li>
            <li><a href="#contact">Contact</a></li>
        </ul>
    </nav>
    <main>
        <section id="home">
            <h2>Home</h2>
            <p>Welcome to our website!</p>
        </section>
        <section id="about">
            <h2>About Us</h2>
            <p>Information about us.</p>
        </section>
        <section id="contact">
            <h2>Contact</h2>
            <p>Contact details here.</p>
        </section>
    </main>
    <footer>
        <p>&copy; 2024 Sample Webpage</p>
    </footer>
</body>
</html>
```

HTML Presentation:

- **Definition**: HTML presentation refers to the styling and visual formatting of the content on a webpage. It involves using CSS (Cascading Style Sheets) to define the appearance of HTML elements.

- **Example**: Using CSS to style the `<h1>`, `<p>`, and other HTML elements.

```html
<!DOCTYPE html>
<html>
<head>
    <title>Styled Webpage</title>
    <style>
        body {
            font-family: Arial, sans-serif;
        }
        header {
            background-color: #4CAF50;
            color: white;
            padding: 10px;
            text-align: center;
        }
        nav ul {
            list-style-type: none;
            padding: 0;
        }
        nav ul li {
            display: inline;
            margin-right: 10px;
        }
        main {
            padding: 20px;
        }
        footer {
            background-color: #333;
            color: white;
            text-align: center;
            padding: 10px;
            position: fixed;
            bottom: 0;
            width: 100%;
        }
    </style>
</head>
<body>
    <header>
        <h1>Website Title</h1>
    </header>
    <nav>
        <ul>
            <li><a href="#home">Home</a></li>
            <li><a href="#about">About</a></li>
            <li><a href="#contact">Contact</a></li>
        </ul>
    </nav>
    <main>
        <section id="home">
            <h2>Home</h2>
            <p>Welcome to our website!</p>
        </section>
        <section id="about">
            <h2>About Us</h2>
            <p>Information about us.</p>
        </section>
        <section id="contact">
            <h2>Contact</h2>
            <p>Contact details here.</p>
        </section>
    </main>
    <footer>
        <p>&copy; 2024 Styled Webpage</p>
    </footer>
</body>
</html>
```

Question 4b

The form below represents a web page which allows the user to record visitor's details of queries issued by administrators. On submission of the form, the data is further processed by the script sniffing.php.

Write the HTML5 codes for query.html (as shown in Figure 1) taking into consideration the above information. You may assume that NO table tag has been used in this form.

```html
<!DOCTYPE html>
<html>
<head>
    <title>Sniffing Cable - Visitors Contact List July 2022</title>
</head>
<body>
    <h1>Sniffing Cable</h1>
    <h2>Visitors contact List July 2022</h2>
    <form action="sniffing.php" method="POST">
        <p>
            <label for="detcontact">Contact Date:</label>
            <input type="date" id="detcontact" name="detcontact">
        </p>
        <p>
            <label for="txtCID">Visitor Id:</label>
            <input type="text" id="txtCID" name="txtCID">
        </p>
        <p>
            <label for="txtFname">First Name:</label>
            <input type="text" id="txtFname" name="txtFname">
        </p>
        <p>
            <label for="txtLname">Last Name:</label>
            <input type="text" id="txtLname" name="txtLname">
        </p>
        <p>
            <label for="rdoGender">Gender:</label>
            <input type="radio" id="rdoGender" name="rdoGender" value="M"> Male
            <input type="radio" id="rdoGender" name="rdoGender" value="F"> Female
        </p>
        <p>
            <label for="slIssues">Issues:</label>
            <select id="slIssues" name="slIssues">
                <option value="Accounts">Accounts</option>
                <option value="Site">Site</option>
                <option value="Others">Others</option>
            </select>
            <label for="txtOther">Other Particulars:</label>
            <input type="text" id="txtOther" name="txtOther">
        </p>
        <p>
            <label for="chkOfficer">Technical Officer:</label>
            <input type="checkbox" id="chkOfficer" name="chkOfficer"> Above Visitor's info
        </p>
        <p>
            <button type="submit">Submit Form</button>
            <button type="reset">Reset Form</button>
        </p>
    </form>
</body>
</html>
```

This code provides the HTML structure for the form depicted in the given figure and includes the specified IDs, names, and values for each input element, as well as the form submission method via POST.