Tribhuvan University
Institute of Science and Technology
2078

Bachelor Level / Fifth Semester / Science
Computer Science and Information Technology (CSC318)
Web Technology
Full Marks: 60 + 20 + 20 Pass Marks: 24 + 8 + 8 Time: 3 Hours

Candidates are required to give their answers in their own words as far as practicable.

The figures in the margin indicate full marks.

Section A
Attempt Any Two Questions

1. How do you define an Array in PHP? Write a PHP script to create a multidimensional array named Product that will contain pcode, pname, and price. Initialize the array with at least three instances. Also, write an HTML script to display the initialized values of the array in an HTML table.

PHP Script

This script creates a multidimensional array named `Product` and initializes it with three instances. It then generates an HTML table to display the array values.

php

```
<?php

// Define and initialize the multidimensional array

$Product = array(

    array("pcode" => "P001", "pname" => "Product 1", "price" =>
29.99),

    array("pcode" => "P002", "pname" => "Product 2", "price" =>
49.99),

    array("pcode" => "P003", "pname" => "Product 3", "price" =>
19.99)

);
```

```
// Start the HTML document

echo "<!DOCTYPE html>

<html lang='en'>

<head>

    <meta charset='UTF-8'>

    <meta name='viewport' content='width=device-width,
initial-scale=1.0'>

    <title>Product Table</title>

    <style>

        table {

            width: 50%;

            border-collapse: collapse;

            margin: 25px 0;

            font-size: 18px;

            text-align: left;

        }

        th, td {

            padding: 12px;

            border-bottom: 1px solid #ddd;

        }

        th {

            background-color: #f2f2f2;

        }

    </style>

</head>
```

```
<body>
    <h1>Product List</h1>
    <table>
        <tr>
            <th>Product Code</th>
            <th>Product Name</th>
            <th>Price</th>
        </tr>";

// Loop through the array and generate table rows
foreach ($Product as $item) {
    echo "<tr>
            <td>{$item['pcode']}</td>
            <td>{$item['pname']}</td>
            <td>\${$item['price']}</td>
          </tr>";
}

// Close the table and HTML document
echo "    </table>
</body>
</html>";
?>
```

```
<?php
```

```
// we can define the array either using one of the
following methods
// 1. array constructor()
$arr = array("dilli", 12.22, True);


// 2. square brackets
$fruits = ["apple", "banana", "citrus"];



// 3. associative array
$associative_array = array(
   "name" => "dilli",
   "age" => 25
);


// 4. associative array with using square brackets
$associative_arr_brackets = [
   "name" => "dillihang",
   "age" => 25
];


// createing multi dimensional array
// 1. using array keyword
$arr = array(
   array("sports", "football"),
   array("music", "salsa")
);

// 2. using square method
$multiarr = [
   ["GOT", "2014"],
   ["HELLO", "WORLD"]
]
?>
```

Explanation:

- **PHP Array**: The `$Product` array is a multidimensional associative array where each sub-array represents a product with its code, name, and price.
- **HTML Table**: The PHP script generates an HTML table where each row corresponds to a product entry.
- **Table Styling**: Basic CSS is used to style the table for better readability.

This script will display the product information in a neatly formatted HTML table.

2. Create an HTML signup form with fields Name, Email, Password, and Age. Validate the form using JavaScript. Write functions for validating each of the elements. All fields should not be empty. The email address should be a valid

email, the password should be of length at least 6 and should start with the alphabet and end with a digit. The age should be between 8 and 60.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Signup Form</title>
    <style>
        body {
            font-family: Arial, sans-serif;
        }
        form {
            max-width: 300px;
            margin: auto;
            padding: 20px;
            border: 1px solid #ddd;
            border-radius: 8px;
        }
        .error {
            color: red;
            font-size: 0.875em;
        }
        .hidden {
            display: none;
        }
    </style>
</head>
<body>
    <h1>Signup Form</h1>
    <form id="signupForm">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name">
        <div id="nameError" class="error hidden">Name is required.</div>

        <label for="email">Email:</label>
        <input type="email" id="email" name="email">
        <div id="emailError" class="error hidden">Invalid email address.</div>

        <label for="password">Password:</label>
        <input type="password" id="password" name="password">
        <div id="passwordError" class="error hidden">Password must be at least 6 characters, start with a letter, and end with a digit.</div>
```

```
             <label for="age">Age:</label>

             <input type="number" id="age" name="age">

             <div id="ageError" class="error hidden">Age must
     be between 8 and 60.</div>

             <button type="submit">Submit</button>

         </form>



         <script>


document.getElementById('signupForm').addEventListener('submit
', function(event) {

let isValid = true;

// Name Validationconst name =
document.getElementById('name').value;

if (name === '') {
document.getElementById('nameError').classList.remove('hidden'
);

isValid = false;

} else {


     document.getElementById('nameError').classList.add('hidde
     n');

                 }



                 // Email Validation

                 const email =
     document.getElementById('email').value;
```

```
            const emailPattern =
/^[^\s@]+@[^\s@]+\.[^\s@]+$/;
            if (!emailPattern.test(email)) {

document.getElementById('emailError').classList.remove('h
idden');
                isValid = false;
            } else {

document.getElementById('emailError').classList.add('hidd
en');
            }


            // Password Validation
            const password =
document.getElementById('password').value;
            const passwordPattern =
/^[a-zA-Z].{4,}[0-9]$/;
            if (password.length < 6 ||
!passwordPattern.test(password)) {

document.getElementById('passwordError').classList.remove
('hidden');
                isValid = false;
            } else {

document.getElementById('passwordError').classList.add('h
idden');
```

```
            }

            // Age Validation
            const age =
document.getElementById('age').value;
            if (age < 8 || age > 60) {

document.getElementById('ageError').classList.remove('hid
den');
                isValid = false;
            } else {

document.getElementById('ageError').classList.add('hidden
');
            }

            // Prevent form submission if invalid
            if (!isValid) {
                event.preventDefault();
            }
        });
    </script>
</body>
</html>
```

Explanation:

- **HTML Form**: Contains fields for Name, Email, Password, and Age, with corresponding error messages.
- **JavaScript Validation**:
  - **Name**: Checks if the field is not empty.
  - **Email**: Validates the email format using a regular expression.
  - **Password**: Ensures the password is at least 6 characters long, starts with a letter, and ends with a digit.
  - **Age**: Ensures the age is between 8 and 60.
- **Error Handling**: Displays error messages if validation fails and prevents form submission.

This setup provides straightforward validation and feedback, ensuring users fill out the form correctly before submission.

How is external CSS inserted in an HTML page? Create an HTML page with two paragraph tags with IDs p1 and p2, and one div tag with ID dv1. Write an external CSS file as per the description: the margin of the paragraphs should be 100px from the top and 80px from the left. Both paragraphs should have the display set to block. The dv1 should have padding left and right set to 30px and 25px respectively, its background color should be blue and the font color should be red with the font type Verdana. Set the display of dv1 to none.

To use external CSS in an HTML page, you link the CSS file using the `<link>` element in the `<head>` section of your HTML document. Below is an example HTML page and an external CSS file that meets your requirements.

HTML Page (`index.html`)

html

Copy code

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>External CSS Example</title>
    <!-- Link to the external CSS file -->
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <p id="p1">This is the first paragraph.</p>
    <p id="p2">This is the second paragraph.</p>
    <div id="dv1">This is a div element.</div>
</body>
</html>
```

External CSS File (`styles.css`)

css

Copy code

```
/* Style for paragraphs */
#p1, #p2 {
    margin-top: 100px;
    margin-left: 80px;
    display: block;
}

/* Style for the div */
```

```
#dv1 {
    padding-left: 30px;
    padding-right: 25px;
    background-color: blue;
    color: red;
    font-family: Verdana, sans-serif;
    display: none;
}
```

Explanation:

- **HTML File**: The `<link>` tag in the `<head>` section links to the external CSS file named `styles.css`.
- **CSS File**:
    - **Paragraphs (`#p1`, `#p2`)**: The margin is set to `100px` from the top and `80px` from the left, and the display property is set to `block`.
    - **Div (`#dv1`)**: The padding for the left and right is set to `30px` and `25px`, respectively. The background color is `blue`, the font color is `red`, the font family is `Verdana`, and the display property is set to `none`.

This setup separates the styling from the HTML content, making it easier to manage and maintain your website's design.

Section B
Attempt Any Eight Questions

What are the services provided under Web 2.0?

Web 2.0 Services

Web 2.0 ushered in a paradigm shift from passive consumption of content to active participation and collaboration. It introduced a plethora of services that revolutionized how we interact online.

Here are some key Web 2.0 services:

Social Networking

- **Platforms:** Facebook, Twitter, Instagram, LinkedIn, TikTok
- **Features:** User profiles, friend connections, messaging, content sharing, groups, and communities.

Content Sharing

- **Video sharing:** YouTube, Vimeo
- **Image sharing:** Flickr, Instagram, Pinterest
- **Audio sharing:** SoundCloud, Spotify
- **Document sharing:** Google Drive, Dropbox

Web Applications

- **Online productivity:** Google Docs, Sheets, Slides
- **Project management:** Trello, Asana
- **Communication:** Slack, Zoom

User-Generated Content

- **Blogs:** WordPress, Blogger
- **Wikis:** Wikipedia, MediaWiki
- **Social bookmarking:** Delicious, Pinterest
- **Online forums and communities:** Reddit, Quora

Other Notable Services

- **Mashups:** Combining data from multiple sources to create new applications (e.g., Google Maps with real-time traffic data).
- **Web APIs:** Allowing developers to access data and functionality from other websites (e.g., Twitter API).
- **Folksonomies:** User-generated tagging systems (e.g., tagging photos on Flickr).

Key characteristics of Web 2.0 services:

- **User-centric:** Focus on user participation and interaction.
- **Dynamic content:** Content is constantly updated and evolving.

- **Collaboration:** Encourages sharing and collaboration among users.
- **Rich media:** Incorporates images, videos, and audio.

These services have transformed the way we communicate, consume information, and interact with each other online.

Create an HTML page with its body containing a div. The div should contain an image within it. Create a link on the image to redirect to the url http://www.tuiost.edu.np. Set the title of the page to “iost”.

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="description" content="iost website">  <title>iost</title>
</head>
<body>
 <div>
   <a href="http://www.tuiost.edu.np">
     <img src="your_image.jpg" alt="iost website"> </a>
 </div>
</body>
</html>
```

How jQuery ID selector can be used to select specific element? Write an example with jQuery that will hide a paragraph by clicking a button.

```html
<!DOCTYPE html>
<html>
<head>
  <title>jQuery ID Selector</title>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
</head>
<body>
  <button id="hideButton">Hide Paragraph</button>
  <p id="myParagraph">This is a paragraph.</p>

  <script>
    $(document).ready(function() {
      $("#hideButton").click(function() {
        $("#myParagraph").hide();
      });
    });
  </script>
</body>
</html>
```

```javascript
// Hide the element
$("#myDiv").hide();
// Show the element
```

```
$("#myDiv").show();

// Toggle the element's visibility

$("#myDiv").toggle();
```

Create an HTML page containing ordered and unordered lists. Set the value of the ordered list type to "A". The list should start at "D".

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width,
initial-scale=1.0">

    <title>Lists Example</title>

    <style>

        body {

            font-family: Arial, sans-serif;

        }

        ul, ol {

            margin: 20px;

        }

    </style>

</head>

<body>

    <h1>Ordered and Unordered Lists</h1>
    <h2>Ordered List (Type A, Start at D)</h2>
    <ol type="A" start="4">
        <li>Item D</li>
        <li>Item E</li>
        <li>Item F</li>
        <li>Item G</li>
    </ol>

    <h2>Unordered List</h2>
    <ul>
        <li>Item 1</li>
        <li>Item 2</li>
        <li>Item 3</li>
        <li>Item 4</li>
    </ul>
</body>
</html>
```

Discuss different JSON data types.

JSON (JavaScript Object Notation) is a lightweight data-interchange format that is easy for humans to read and write and easy for machines to parse and generate. It supports six primary data types:

Primitive Data Types

These are the basic building blocks of JSON data.

- **String:** A sequence of characters enclosed in double quotes (e.g., "Hello, World!", "true", "123").
- **Number:** A numeric value, including integers and floating-point numbers (e.g., 42, 3.14, -10).
- **Boolean:** A logical value, either true or false.
- **Null:** Represents a null value.

Complex Data Types

These data types allow for structured data representation.

- **Object:** An unordered collection of key-value pairs. Keys must be strings, and values can be any JSON data type. Objects are enclosed in curly braces `{}`

Valid Data Types

In JSON, values must be one of the following data types:

- a string
- a number
- an object (JSON object)
- an array
- a boolean
- *null*

`{"name":"John"}- JSON STRINGS`

`{"age":30} - JSON Objects` Values in JSON can be objects.

```
{

"employee":{"name":"John", "age":30, "city":"New York"}

}

{

"employees":["John", "Anna", "Peter"]
}
```

```
{"sale":true}
```

```
{"middlename":null}
```

Values in JSON can be arrays.

Values in JSON can be true/false

Values in JSON can be null.

Objects as values in JSON must follow the JSON syntax.

What is DTD? Create an XML file and write its equivalent DTD.

A Document Type Definition (DTD) is a set of rules that define the structure and the legal elements and attributes of an XML document. DTDs are used to enforce the structure of XML documents, specifying the types of elements, their order, and their relationships.

DTDs can be internal or external, where internal DTDs are defined within the XML

document, and external DTDs are stored in separate files.

Here's an example of a simple XML file representing information about books and its equivalent DTD:

XML File (books.xml):

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE library SYSTEM "books.dtd">

<library>

<book>

<title>Introduction to XML</title>

<author>John Doe</author>

<publication_year>2020</publication_year>

<price>29.99</price>
```

```
</book>
<book>
<title>Data Science Basics</title>
<author>Jane Smith</author>
<publication_year>2021</publication_year>
<price>39.99</price>
</book>
</library>
```

DTD File (books.dtd):

```
<!ELEMENT library (book+)>
<!ELEMENT book (title, author, publication_year, price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT publication_year (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

In this example:

- `<!ELEMENT library (book+)>`: Defines that the `<library>` element must contain one

or more `<book>` elements.

- `<!ELEMENT book (title, author, publication_year, price)>`: Specifies that the `<book>`

element must contain the elements `<title>`, `<author>`, `<publication_year>`, and

`<price>` in that order.

- `<!ELEMENT title (#PCDATA)>`, `<!ELEMENT author (#PCDATA)>`, `<!ELEMENT

publication_year (#PCDATA)>`, `<!ELEMENT price (#PCDATA)>`: Define that these

elements contain parsed character data (#PCDATA).

The `DOCTYPE` declaration in the XML file references the external DTD file (`books.dtd`), specifying the rules for the structure of the XML document. The DTD enforces the structure of the XML file, ensuring that it adheres to the specified rules and preventing the use of undefined elements or invalid structures.

How can you define a function in PHP? Create a function that will take two integers as arguments and will return the product of them.

In PHP, you define a function using the `function` keyword followed by the function name, parentheses, and curly braces. Here's how you can define a function that takes two integers as arguments and returns their product:

php

```php
<?php

// Define the function

function multiply($a, $b) {

    return $a * $b;

}


// Example usage of the function

$result = multiply(4, 5);

echo "The product of 4 and 5 is: " . $result;

?>
```

Explanation:

- `function multiply($a, $b)`: Defines a function named `multiply` with two parameters `$a` and `$b`.

- `return $a * $b;`: Computes the product of the two parameters and returns the result.
- `$result = multiply(4, 5);`: Calls the function with `4` and `5` as arguments and stores the result in `$result`.
- `echo "The product of 4 and 5 is: " . $result;`: Outputs the result.

This function will output `The product of 4 and 5 is: 20`.

Describe the CSS Box Model with an example.

CSS Box Model

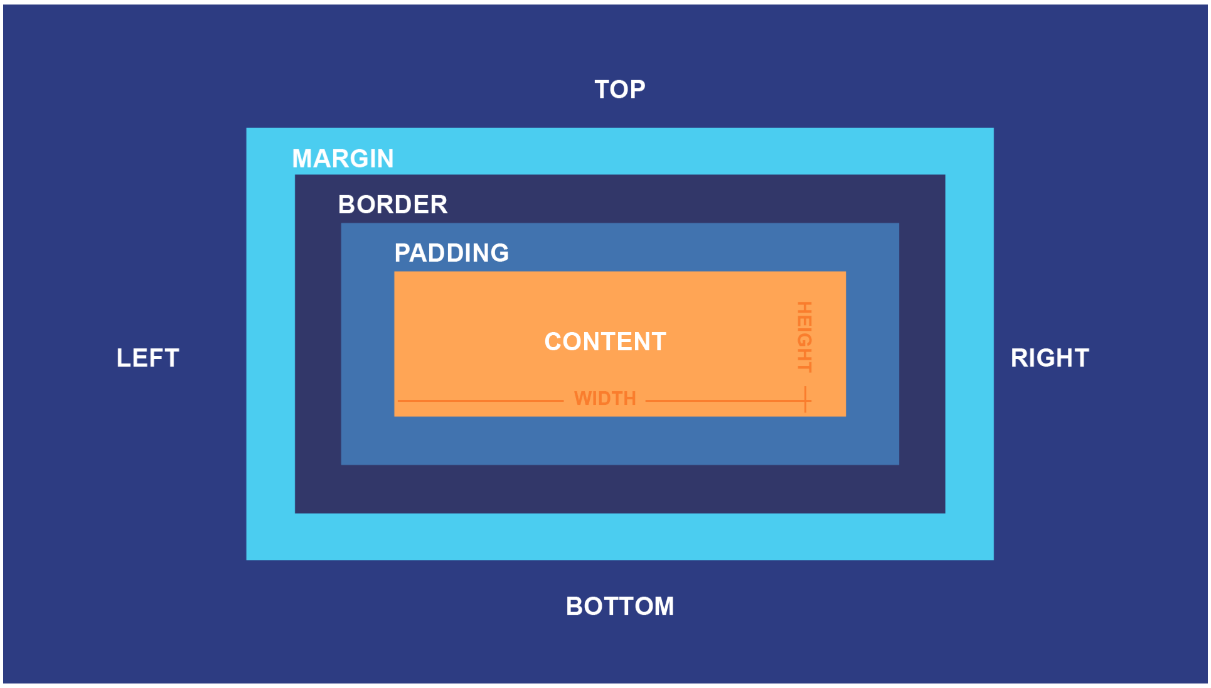
The CSS Box Model is a fundamental concept that governs how elements are structured and positioned on a webpage. Every HTML element can be considered as a box, and the box model describes the properties of this box.

Components of the Box Model:

1. **Content:** This is the actual content of the element, like text, images, or other elements.
2. **Padding:** The space between the content and the border.
3. **Border:** The border surrounds the content and padding.
4. **Margin:** The space outside the border, separating the element from other elements.

```
<!DOCTYPE html>

<html>

<head>

  <style>

      div {

          width: 200px;

          border: 5px solid blue;

          padding: 15px;

          margin: 10px;
```

```
    }
  </style>
</head>
<body>
  <div>
    This is a div element.
  </div>
</body>
</html>
```


TOP
MARGIN
BORDER
PADDING
LEFT
CONTENT
HEIGHT
WIDTH
RIGHT
BOTTOM