



KAMALDNP

kamaldnp.com • IT Training, Varanasi

PDF created by Raj Kamal using AI

Practice Question Paper (Theory Part) — Web Technologies

BTEUP Diploma in Computer Science & Engineering • 3rd Semester • Session 2026-27
Session 2026-27

Practice Questions / Model Paper — NOT a BTEUP board paper

Time: 2.30 Hours (suggested practice time) Practice Score: 50 (self-check only)

General Instructions:

- All questions are compulsory. Each question has three parts (a), (b) and (c) — **answer any two** of the three parts in every question.
- Each part carries 5 marks; each question carries 10 marks.
- Draw neat, labelled diagrams wherever they are asked or wherever they support your answer.
- Write your answers in your own words, in the same order as the questions.

About this paper: In the verified BTEUP 2026-27 scheme this subject is a PRACTICUM (100 marks — lecture and practical work together); it has no separate board theory paper. This practice paper is KAMALDNP Academy's own question set for the theory (lecture) part of the practicum, written from the official syllabus topics, so you can test your concepts for the practicum assessment and viva. The marks shown (10 per question, 50 in all) are a practice weighting for self-checking only — they are NOT board marks. This is NOT a BTEUP board paper, and no question in it is promised to appear anywhere.

Q1. (From: Unit 1 — Web Development Introduction) — Answer any two of the following:

(a) Define the following in brief: internet, WWW, web browser and search engine.

(5)

- (b) Explain the client-server model with a diagram. Write the parts of the URL `https://kamaldnp.com/courses/python.html` and state the full form of URL. (5)
- (c) Differentiate between static, dynamic and responsive websites. Also state what View Page Source and Inspect Element are used for. (5)

Q2. (From: Unit 2 — HTML Basics) — Answer any two of the following:

- (a) Explain the basic structure of an HTML document. In your answer, define a tag and an attribute, name the two types of tags, and state the rule of nesting. (5)
- (b) Write the complete HTML code to display a table with a caption 'My Marks', a header row Subject and Marks, and these data rows: Web Technologies 88, Database 92, Operating Systems 85. Use a border of 1 for the table. (5)
- (c) Write the HTML code for a registration form having fields for Name (text), Email, Gender (radio buttons for Male and Female), Hobbies (checkboxes) and a Submit button. (5)

Q3. (From: Unit 3 — Cascading Style Sheets) — Answer any two of the following:

- (a) What is CSS and what are its benefits? Write the syntax of a CSS rule and differentiate between inline, internal and external CSS. (5)
- (b) Explain the following CSS selectors with one example each: ID selector, class selector, grouping selector, universal selector and pseudo-class selector. (5)
- (c) Explain the CSS box model with a labelled diagram. Then write the CSS and HTML code for a box of width 300px with a light yellow background, 15px padding, a 2px solid orange border and 20px margin, containing centred text. (5)

Q4. (From: Unit 4 — JavaScript) — Answer any two of the following:

- (a) What is JavaScript? Explain variables, the main data types and the types of operators in JavaScript in brief. State the two ways of adding JavaScript to a web document. (5)
- (b) Write JavaScript code, using a for loop and an if statement, to print all even numbers from 1 to 20. Also write the output of your code. (5)
- (c) Explain JavaScript objects and the Document Object Model in brief. How is a JavaScript function called on an HTML event? Give a small code example using a button click. (5)

Q5. (From: Unit 5 — PHP) — Answer any two of the following:

- (a) What is server-side scripting? Explain how a PHP script receives the values submitted from an HTML form. (5)
- (b) Explain arrays and functions in PHP with small examples. Your answer should cover an indexed array, an associative array, and a function that takes two parameters and returns their sum. (5)
- (c) Write the PHP code (essential steps, in order, using MySQLi functions) to connect to a MySQL server, create and select a database CollegeDB, create a table Student (RollNo, Name, Marks), insert one row into it and close the connection. Also state what phpMyAdmin is used for. (5)

Model Answer Outline — Web Technologies

The key points a full-marks answer contains, question by question.

Practice Questions / Model Paper — NOT a BTEUP board paper

Q1(a)

- Internet: the worldwide network of networks that connects computers using standard protocols; it carries many services, of which the web is only one.
- WWW (World Wide Web): the service of linked web pages and other resources, identified by URLs and viewed through a browser, running over the internet.
- Browser: the client software (for example Chrome or Firefox) that requests a page, receives HTML/CSS/JavaScript and displays it for the user.
- Search engine: a service that indexes web pages and returns a ranked list of links for the keywords a user types, for example when a student searches for a syllabus PDF.
- Difference line: the internet is the network, the WWW is a service on it, the browser is the tool to view that service, and a search engine helps find pages in it.

Q1(b)

- Client-server model: the client (browser) requests a service and the server (web server) processes the request and sends back a response, usually an HTML page.
- Flow to state or draw: user types a URL, browser sends an HTTP request, server finds or builds the page, response returns, browser displays the page.
- URL full form: Uniform Resource Locator — the address of a resource on the web.
- Parts of `https://kamaldnp.com/courses/python.html` : protocol `https`, server/domain `kamaldnp.com`, path `courses/python.html` naming the folder and file requested.
- The server keeps the website files and runs continuously, while a client connects only when it needs a page.

Q1(c)

- Static website: fixed HTML pages shown as stored; content changes only when the developer edits the files; suitable for a small information site.
- Dynamic website: pages are built by a server program using data (often from a database) at request time, so content changes per user or per data update; example: a login-based student portal.
- Responsive website: layout adjusts automatically to screen size, so the same site works well on mobile, tablet and desktop.
- Developer options: View Page Source shows the HTML received; Developer Tools and Inspect Element let you select an element and see and test its HTML and CSS live in the browser.

- State clearly: inspecting a page changes only your local display, not the actual website on the server.

Q2(a)

- Basic structure skeleton: `<!DOCTYPE html>`, then `<html>` containing `<head>` (with `<title>`) and `<body>` holding the visible content, each part closed in reverse order.
- Tags are instructions written in angle brackets; most tags work in pairs (opening and closing), while a few such as `<br>`, `<hr>` and `<img>` are single (empty) tags.
- Attributes give extra information inside the opening tag, written as name and value in quotes, for example `<img src='photo.jpg' alt='My photo'>`.
- Rules of nesting: an inner tag must close before its outer tag closes, for example `<p><b>text</b></p>` is correct and crossing tags are wrong.
- Headings h1 to h6, paragraph p, line break br and comment `<!-- ... -->` should be named as basic examples; a comment is not displayed on the page.

Q2(b)

- Table skeleton: `<table border='1'>` with a `<caption>`, a header row using `<th>`, and data rows using `<td>`; tags closed in correct nesting order.
- Expected rows: header row Subject | Marks; then rows Web Technologies | 88, Database | 92, Operating Systems | 85.
- Marks column values are numbers only, so the table is easy to read and check.
- Full code is short: DOCTYPE, html, head with title, body with heading and the table. No CSS is asked, so do not add styling.
- Common errors to avoid: forgetting `</table>`, using `<td>` for the heading row, or leaving a row unclosed with `</tr>` missing.

Q2(c)

- Form skeleton: `<form action="" method='post'> ... </form>` containing labelled input controls and a submit button.
- Controls in order: text input for Name (name='name'), email input for Email (name='email'), radio buttons sharing name='gender' with values Male and Female, checkboxes for hobbies, and `<input type='submit' value='Register'>`.
- The name attribute of each control is essential: it is the key under which the value is sent to the server.
- Radio buttons in one group share one name so only one can be selected; checkboxes allow more than one choice.
- Keep the code complete but plain: labels as text before each control and `<br>` for line breaks are enough for this task.

Q3(a)

- CSS (Cascading Style Sheets) separates the presentation of a page from its HTML structure, so layout and appearance can be changed in one place for many pages.
- Benefits: same style reused across pages, smaller cleaner HTML, easy site-wide changes, and different presentation for different screen sizes.

- CSS syntax: selector { property: value; property: value; } — for example p { color: blue; font-size: 14px; }.
- Inline implementation: style attribute inside one tag, applies to that element only. Internal: a <style> block in the head of one page. External: a separate .css file linked with <link>, applying to all pages that link it.
- Preference line: external CSS is preferred for a real website because one change updates every page.

Q3(b)

- Element/class basics implied by the asked selectors: a class selector (.name) styles all elements with that class and can be reused many times on a page.
- ID selector (#name) styles the one element with that unique id, so it is used once per page for a special element.
- Grouping selector: the same style is given to several selectors separated by commas, for example h1, h2, p { font-family: Arial; }.
- Universal selector (*) applies a style to every element on the page, commonly used to reset margin and padding.
- Pseudo-class selects an element in a special state, for example a:hover changes a link's style when the mouse is over it and a:link styles an unvisited link.

Q3(c)

- Box model, inside out: content in the centre, then padding (space inside, around the content), then border, then margin (space outside, between this box and other elements). A labelled diagram earns marks.
- Code skeleton: .box { width: 300px; background-color: lightyellow; padding: 15px; border: 2px solid orange; margin: 20px; } and a matching <div class='box'> ... </div>.
- Each asked property must appear: background colour of the page from a body rule, and width, padding, border and margin in the box rule.
- Text should be centred (text-align: center on the box or heading), matching the task wording.
- Order of writing: CSS rule first, then the HTML element that uses class='box', so the reader can link the two.

Q4(a)

- JavaScript is the client-side scripting language of the web: it runs in the browser and makes pages interactive, for example validating forms and responding to clicks.
- Variables are named storage for values, declared with var, let or const, for example let marks = 85;
- Data types to name: number, string, boolean, undefined and null, with one tiny example each, such as 'Hello' for string and true for boolean.
- Operators to name: arithmetic (+, -, *, /, %), assignment (=, +=), comparison (==, >, <) and logical (&&, ||, !), with the comparison/logical ones mainly used in conditions.

- Adding JavaScript to a document: internal code in a `<script>` block in the page, or an external .js file linked with `<script src='file.js'></script>` — state both ways.

Q4(b)

- Code logic: read or set a limit value, run a for loop with a counter from 1 to 20, test each number with `if (i % 2 == 0)` and print only when the condition is true.
- Essential skeleton:

```
for (let i = 1; i <= 20; i++) { if (i % 2 == 0) { document.write(i + ' '); } }
```
- Correct output to state: 2 4 6 8 10 12 14 16 18 20.
- The % operator gives the remainder, so an even number gives remainder 0 — this line explains why the condition works.
- A while-loop version would need the counter increased inside the loop body; forgetting that increase causes an endless loop.

Q4(c)

- A JavaScript object is a collection of related named values (properties) and actions, for example a student object with name and marks properties.
- DOM (Document Object Model) is the browser's tree-shaped representation of the HTML document; JavaScript uses it to find elements and change their text, value or style.
- Access method to name: `document.getElementById('id')` returns the element with that id, and its innerHTML or value property can then be read or changed.
- An HTML event is an action such as a button click (onclick), a page load or a key press; the event attribute names the JavaScript function to run.
- Code skeleton: `<button onclick='show()'>Show</button>` with function `show()`
`{ document.getElementById('msg').innerHTML = 'Welcome'; }` and a paragraph with `id='msg'`.

Q5(a)

- Server-side scripting means the program runs on the web server, not in the browser; the server executes the code and sends only the resulting HTML to the client.
- Because the code stays on the server, the visitor cannot view the PHP source through View Source, unlike JavaScript.
- PHP form handling: values sent by a form are received in the `$_POST` array (or `$_GET` for URL data) by the control's name, for example `$name = $_POST['name'];`
- Echo/print is then used to display or use the received values, often after checking that the fields are not empty.
- Typical uses to name: login systems, processing registrations, and storing form data in a database.

Q5(b)

- An array stores several values under one name. Indexed array example: `$subjects = array('HTML', 'CSS', 'PHP');` with elements reached as `$subjects[0]`, `$subjects[1]` and so on.
- An associative array uses named keys, for example `$student = array('name' => 'Student', 'marks' => 85);` reached as `$student['marks']`.

- A user-defined function is declared once with the function keyword and called whenever needed: `function add($a, $b) { return $a + $b; }` and the call `echo add(4, 5);` displays 9.
- Parameters carry values into the function and return sends the result back to the caller; the small addition example proves both points.
- `count()` on an array gives its number of elements — a useful built-in to mention with arrays.

Q5(c)

- Connection step: `$conn = mysqli_connect('localhost', username, password);` — write placeholder words for the user name and password, never a real password. Check and stop with an error message if the connection fails.
- Create and select: `mysqli_query($conn, 'CREATE DATABASE IF NOT EXISTS CollegeDB');` followed by `mysqli_select_db($conn, 'CollegeDB');`
- Create table: `CREATE TABLE IF NOT EXISTS Student (RollNo INT PRIMARY KEY, Name VARCHAR(50), Marks INT)` sent through `mysqli_query`.
- Insert one row: `INSERT INTO Student (RollNo, Name, Marks) VALUES (1, 'Student', 88)` sent through `mysqli_query`, then close with `mysqli_close($conn);`
- phpMyAdmin line: phpMyAdmin is the web-based tool for doing these same database jobs (create, view, edit, delete databases and tables) through a graphical interface instead of code, and database errors/bugs are commonly checked there and through the error messages returned by queries.

KAMALDNP • kamaldnp.com • support@kamaldnp.com • +91 8318013442 • Varanasi, UP
Prepared by KAMALDNP Academy from the official BTEUP 2026-27 syllabus. This is a practice/model paper only.