


Answer to the question no. 4

Purchased informations of a company

Purchased Equipment			
Item No.	Item Image	Description	Price
		Shipping	Expense
1		IBM	\$400
		Shipping	\$40
<u>Total Cost</u>			\$440

Answer to the question no. 6

Personal Details

Name:

Gender : ☐ Male ☐ Female

Email:

Date of Birth: 

City:

Address :

Write your address here...

Submit

Answer to the question no. 8

Favourite players in **Word Cricket**

- David Warner
- Ben Stokes
- Shakib Al Hasan

Answer to the question no. 13(i)

Name:	Bill Gates
Telephone:	555555555
	4444444

Answer to the question no. 14

Table 1: Personal Information of Five Friends

SL	Name	Education	Occupation	Income
1	Boden	Primary	Farmer	\$5800
2	Alen	Habilitation & Full Prof		\$10000
3	Aono	Phd	Professor	\$12000
Total Income				\$45800

Answer to the question no. 20

Gender	Average	
	Height(feet)	Weight(Kg)
Male	5.5	67
Female	5.2	55

An ordered HTML List:

-
11. Item 1

12. Item 2

13. Item 3

14. Item 4

15. Item 5

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
  /> -->
    <title>Final Questions Solution</title>

  </head>
<body>

  <h1 style="text-align: center">Answer to the question no. 4</h1>

  <table border="1" align="center" cellspacing="0" cellpadding="5">
    <caption><i>Purchased informations of a company </i> </caption>
    <tr><th colspan="4">Purchased Equipment</th><tr>
      <tr>
        <th rowspan="2">Item No.</th>
        <th rowspan="2">Item Image</th>
        <th>Description</th>
        <th>Price</th>
      </tr>
      <tr>
        <th>Shipping</th>
        <th>Expense</th>
      </tr>
      <tr>
        <td rowspan="2" align="center">1</td>

```

```

        <td rowspan="2" align="center"></td>
        <td>IBM</td>
        <td>$400</td>
    </tr>
    <tr>
        <td>Shipping</td>
        <td>$40</td>
    </tr>
    <tr>
        <td colspan="3" align="center"><u><b><i>Total Cost</i></b></u> </td>
        <td><b>$440</b></td>
    </tr>
</table>

```

<h1 style="text-align: center">Answer to the question no. 6 </h1>

```

<form>
<fieldset style="width:100px" >
    <legend>Personal Details</legend>
    Name: <input name="firstName" placeholder="Your name" /> <br> <br>
    Gender :
    <input type="radio" name="gender" value="male" /> Male
    <input type="radio" name="gender" value="female" />Female <br> <br>
    Email:
    <input type="email" name="email" /><br> <br>
    Date of Birth:
    <input type="date" name="birthDate"><br> <br>

```

City:

```
<select name="City">
  <option >Thakurgaon</option>
  <option>Rangpur</option>
  <option>Dhaka</option>
  <option>Chittagong</option>
```

```
</select><br> <br>
```

Address :


```
<textarea cols="30" rows="2" >Write your address here...
</textarea><br>
```

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

```
</fieldset>
```

```
</form>
```

```
<h1 style="text-align: center">Answer to the question no. 8 </h1>
```

```
<p>Favourite players in <u><b>Word Cricket</b></u></p>
```

```
<ul>
```

```
<ul type = "square">
```

```
<li>David Warner</li>
```

```
<li>Ben Stokes</li>
```

```
<li>Shakib Al Hasan</li>
```

```
</ul>
```

```
</ul>
```

```
<h1 style="text-align: center">Answer to the question no. 13(i) </h1>
```

```
<table border = "1" cellpadding="10" cellspacing="10">
```

```
<tr>
```

```

    <th>Name:</th>
    <td>Bill Gates</td>
</tr>
<tr>
    <th rowspan="2">Telephone:</th>
    <td>5555555555</td>
</tr>
<tr>
    <td>44444444</td>
</tr>
</table>

```

<h1 style="text-align: center">Answer to the question no. 14 </h1>

<p>Table 1: Personal Information of Five Friends</p>

```

<table border="1" cellspacing = "0" cellpadding="5">
  <tr>
    <th>SL</th>
    <th>Name</th>
    <th>Education</th>
    <th>Occupation</th>
    <th>Income</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Boden</td>
    <td>Primary</td>
    <td>Farmer</td>
    <td>$5800</td>
  </tr>

```

2	Alen	Habilitation & Full Prof		\$10000
3	Aono	Phd	Professor	\$12000
Total Income				\$45800

Answer to the question no. 20

Gender	Average	
	Height(feet)	Weight(Kg)

```
</tr>
<tr>
  <td>Male</td>
  <td>5.5</td>
  <td>67</td>
</tr>
<tr>
  <td>Female</td>
  <td>5.2</td>
  <td>55</td>
</tr>
</table>
```

```
<h1 style="text-align: center;"> <u>An ordered HTML List:</u></h1>
<ol style = "text-align: center;
list-style-position: inside;" type="1" start="11">
  <li >Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
  <li>Item 4</li>
  <li>Item 5</li>
</ol>
</body>
</html>
```

1. What is an Event in JavaScript? Explain two events with a proper example.

JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page.

When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc.

Events are a part of the Document Object Model (DOM) Level 3 .

Example from sheet

2. Write a code in JavaScript using while loop that print the following output:

1 2 5 10 17.....82

```
<script>
var count;
  document.write("Starting Loop" + "<br />");
  var count=0;
  while(count<10)
  {
    var print;
    print = (count*count)+1;
    document.write(" " + print);
    count++;
  }
  document.write("<br>"+ "Loop stopped!");
</script>
```

3. Define the scope of JavaScript variables. Explain it types with JavaScript program.

This means that the scope of a variable is the block of code in the entire program where the variable is declared, used, and can be modified.

JavaScript variables have only two scopes.

- Global Variables – A global variable has global scope which means it can be defined anywhere in JavaScript code.
- Local Variables – A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

<html>

<body onload = checkscope();>

<script type = "text/javascript">


```

var myVar1 = "global"; // Declare a global variable

function checkscope( ) {

var myVar2 = "local"; // Declare a local variable

document.write(myVar1);

document.write(myVar1);


}

</script>

</body>

</html>

```

4. Write a program using JavaScript Function which shows the message “I am the of CSE department in HSTU” after clicking a button.

```

<script language="javascript" type="text/javascript">
    function message(){
        alert("I am the student of CSE department in HSTU");
    }
</script>
<p>Click the following button to call the function</p>
<form>
<input type="button" onclick="message()" value="Click Here!">
</form>

```

5. How can JavaScript be implemented in a web page? Explain the attributes of <script> tag.

Src- It specifies the URL of an external script file.

Type- It specifies the media type of the script.

Language - This attribute specifies what scripting language you are using.

async - Specifies that the script is executed asynchronously.

Charset- Defines the character encoding that the script uses.

Defer- Defines, that the script must be executed after the loading of the page.

6. How do you differentiate between dynamic webpage and static webpage?

| Static Web Page | Dynamic Web Page |
|---|---|
| The content and layout of a web page is fixed | The content and layout may change during run time |
| Static Web pages never use databases | Databases is used to generate dynamic content through queries |
| Static web pages directly run on the browser and do not require any server side application program | Dynamic web pages runs on the server side application programs and displays the results |
| Static Web pages are easy to develop | Dynamic web page development requires programming skills |

| Components | HTML codes | HTML codes and programs |
|---------------|-------------------------|---|
| Programming | Non-programming | Programming is needed |
| Interactivity | Low | High |
| Input data | No input data is needed | Input data are usually collected by forms |

7. What do you mean by NULL and Undefined in JavaScript? Write a code snippet which shows the use of conditional operator.

Null in JavaScript means an empty value and is also a primitive type in JavaScript. The variable which has been assigned as null contains no value.

Undefined, on the other hand, means the **variable has been declared**, but its value **has not been assigned**.

The conditional (ternary) operator is the only JavaScript operator that takes three operands: a condition followed by a question mark (?), then an expression to execute if the condition is truthy followed by a colon (:), and finally the expression to execute if the condition is falsy. This operator is frequently used as an alternative to an if...else statement.

(Condition)?True statement: False statement

8. Is JavaScript a case-sensitive language? What are the variable naming conventions in JavaScript?

JavaScript is a case-sensitive language. This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

So the identifiers **Time** and **TIME** will convey different meanings in JavaScript.

Variable naming conventions in JavaScript:

- We should not use any of the JavaScript reserved keywords as a variable name.. For example, **break** or **boolean** variable names are not valid.
- JavaScript variable names should not start with a numeral (0-9).
- They must begin with a letter or an underscore character. For example, 123test is an invalid variable name but _123test is a valid one.
- JavaScript variable names are case-sensitive. For example, **TIME** and **time** are two different variables.
- Variable and function names written as camelCase
- Global variables written in UPPERCASE (We don't, but it's quite common)
- Constants (like PI) written in UPPERCASE

9. What are the ways of linking JavaScript with web pages? Explain with example.

Javascript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

There are typically three ways to add JavaScript to a web page:

1. Embedding the JavaScript code between a pair of <script> and </script> tag.
2. Creating an external JavaScript file with the **.js extension** and then load it within the page through the **src attribute of the <script> tag**.
3. Placing the JavaScript code directly inside an HTML tag using the **special tag attributes** such as onclick, onmouseover, onkeypress, onload, etc.

1.

```
<body>
<script> var greet = "Hello World!";
document.write(greet); // Prints: Hello World!
</script>
</body>
```

2.

```
Int.js
document.write("this is external js file");
```

```
<body>

    <script src="js/hello.js"></script>

</body>

3.

<button onclick="alert('Hello World!')">Click Me</button>
```

1. What is PHP? Why PHP is a must for CSE students to build career in the Web Development domain?

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web-based software applications.

PHP is a MUST for students and working professionals to become a great Software Engineer especially when they are working in Web Development Domain.

Some of the key advantages of learning PHP:

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

2. Explain different ways to differentiate PHP code from other elements in a web page.

Escaping to PHP

The PHP parsing engine needs a way to differentiate PHP code from other elements in the page. The mechanism for doing so is known as 'escaping to PHP'. There are four ways to do this –

i. Canonical PHP tags

The most universally effective PHP tag style is –

```
<?php ?>
```

ii. Short-open (SGML-style) tags

Short or short-open tags look like this –

<?...?>

We must do one of two things to enable PHP to recognize the tags –

- Choose the **--enable-short-tags** configuration option when building PHP.
- Set the short_open_tag setting in php.ini file to on.

iii. ASP-style tags

ASP-style tags mimic the tags used by **Active Server Pages** to delineate code blocks.

ASP-style

tags look like this –

<%...%>

We need to set the configuration option in your php.ini file.

iv. HTML script tags

HTML script tags look like this –

<script language = "PHP">...</script>

3. Mention the types of loops used in PHP. Explain the syntax for ‘foreach’ loop with example?

PHP supports following four loop types.

- for – loops through a block of code a specified number of times.
- while – loops through a block of code if and as long as a specified condition is true.
- do...while – loops through a block of code once, and then repeats the loop as long as a special condition is true.
- foreach – loops through a block of code for each element in an array.

The syntax for indexed arrays is as given in the following code block:

foreach (iterable as \$value)

statement

```
<?php
$flexible = array("Hire", "top", "freelance", "developers");

foreach ($flexible as $value) {
    echo "$value <br>";
}
?>
```

Output:

```
Hire
Top
Freelance
developers
```

The syntax for associative arrays:

foreach (iterable as \$key => \$value)

statement

```
<?php
$freelancer = array(
    "name" => "Eric",
    "email" => "Eric@gmail.com",
    "age" => 22,
    "gender" => "male"
);

// Loop through employee array
foreach($freelancer as $key => $value) {
    echo $key . ": " . $value . "<br>";
}
?>
```

```
name: Eric
email: Eric@gmail.com
age: 22
gender: male
```

4. What is the meaning of break and continue statements in PHP? Give PHP code snippet of these statements as examples.

Continue and break keywords used to control the loops execution.

The PHP break keyword is used to **terminate the execution of a loop prematurely**.

The break statement is

- Situated inside the statement block.
- Gives us full control
- Whenever we want to exit from the loop we can come out.

An example of break statement:

```
<html>
<body>
<?php
$i = 0;
while( $i < 10)
{
    $i++;
    if( $i == 3 )break;
    echo( " Value is $i <br>");
}
echo ("Loop stopped at i = $i" );
?>
</body>
</html>
```

Output:

Value is 0

Value is 1

Value is 2

Loop stopped at i=3

The PHP continue keyword is **used to halt the current iteration of a loop** but it *does not terminate the loop*.

The continue statement is

- Situated inside the statement block containing the code that the loop executes
- *Preceded by a conditional test.*
- For the pass encountering continue statement, rest of the loop code is skipped and next pass starts.

An example of continue statement:

```
<html>
<body>
```

```

<?php
$array = array( 1, 2, 3, 4, 5);
foreach( $array as $value ) {
    if( $value == 3 )
        continue;
    echo "Value is $value <br />";
}
?>
</body>
</html>

```

Output:

Value is 1
 Value is 2
 Value is 4
 Value is 5

5. How can a text be printed using PHP? Write a program in PHP to check whether a number is prime or not?

There are two basic ways to print a text or get output: **echo and print**.

Echo statement:

The echo statement can be used with or without parentheses: echo or echo().

For example:

echo("I Love You");

or echo "I love you"; gives us same result.

Print statement is same as echo.


```

<?php
function check_prime($num)
{
    if ($num == 1)
        return 0;
    for ($i = 2; $i <= $num/2; $i++)
    {
        if ($num % $i == 0)
            return 0;
    }
    return 1;
}
$num = 47;
$flag_val = check_prime($num);
if ($flag_val == 1)
    echo "It is a prime number";
else
    echo "It is a non-prime number"
?>

```

6. Mention the sorting functions for arrays in PHP. Explain the functions used for adding and removing array elements in PHP.

Sorting Functions for Arrays In PHP

1. sort() – sorts arrays in ascending order
2. rsort() – sorts arrays in descending order
3. asort() – sorts associative arrays in ascending order, according to the value
4. ksort() – sorts associative arrays in ascending order, according to the key
5. arsort() – sorts associative arrays in descending order, according to the value
6. krsort() – sorts associative arrays in descending order, according to the key

Rest of ans from sheet.

7. Mention the types of loops used in PHP. Write a PHP code snippet to apply for each loop.

Same as question 3.

8. Explain the php function of stripslashes() and str_replace().

The strpos() is in-built function of PHP. It is used to find the position of the first occurrence of a string inside another string or substring in a string.

```
<?php
    $str1="Hello php";
    $str2=" Hello php javatpoint!";
    echo "First string is: ".$str1;
    echo "<br>";
    echo "First string is: ".$str2;
    echo "<br>";
    echo "By using 'strpos()' function:".strpos("$str1,$str2","php");
    //Find the position of the first occurrence of "php" inside the string
?>
```

Output:
First string is: Hello php
First string is: Hello php javatpoint!
By using 'strpos()' function:6

The str_replace() function replaces some characters with some other characters in a string.

For example:

Replace the characters "world" in the string "Hello world!" with "Peter":

```
<?php
echo str_replace("world","Peter","Hello world!");
?>
```

Output:

Hello Peter!

9. What is PHP? Describe the importance of using PHP

Same as question 1

HTML

1. What is HTML? Describe basic structure of HTML with a proper example.

HTML (Hypertext Markup Language) is the standard markup language used for creating web pages and structuring their content. It defines the structure and presentation of the content within a web page. HTML uses a set of tags to mark up and organize the elements and text within a document.

The basic structure of an HTML document consists of the following elements:

```
...  
<!DOCTYPE html>  
<html>  
<head>  
  <title>Title of the Page</title>  
</head>  
<body>  
  <!-- Content of the web page goes here -->  
</body>  
</html>  
...
```

Here's a breakdown of the basic structure:

1. `<!DOCTYPE html>`: This declaration at the beginning of the document defines the version of HTML being used. In this case, it indicates that the document is written in HTML5.
2. `<html>`: The `<html>` element serves as the root element of the HTML document. It encapsulates the entire web page content.
3. `<head>`: The `<head>` element contains meta-information about the HTML document. It includes elements such as the document title, meta tags, style sheets, and scripts. The content within the `<head>` section is not displayed on the web page.
4. `<title>`: The `<title>` element is placed within the `<head>` section and specifies the title of the web page. The text within the `<title>` tag is displayed as the title of the browser window or tab.
5. `<body>`: The `<body>` element contains the visible content of the web page. This is where you place headings, paragraphs, images, links, forms, and other HTML elements that are rendered and displayed on the web page.

The actual content of the web page, such as text, images, and other elements, is placed within the ``<body>`` tags.

Here's an example of a simple HTML document:

```
<!DOCTYPE html>
<html>
<head>
  <title>My First Web Page</title>
</head>
<body>
  <h1>Welcome to My First Web Page</h1>
  <p>This is a paragraph of text.</p>
  
  <a href="https://www.example.com">Visit Example.com</a>
</body>
</html>
```

In this example, the HTML document has a title, a heading (`<h1>`), a paragraph (`<p>`), an image (``), and a link (`<a>`). When opened in a web browser, the document will display the content accordingly.

This is a basic structure of an HTML document. You can add more elements, styles, scripts, and other features to create rich and interactive web pages.

2. Shortly explain list in HTML. Describe the types of HTML list with examples.

In HTML, a list is a way to present information in a structured and organized manner. Lists help to group related items together and provide a clear visual representation of the content. There are two main types of lists in HTML: ordered lists and unordered lists.

1. Ordered List (``):

An ordered list is used when the sequence or order of items is important. The list items are automatically numbered in sequential order. The numbering can be customized using CSS or the `start` attribute. Here's an example:

```
<ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

...

Output:

1. First item
2. Second item
3. Third item

2. Unordered List (``):

An unordered list is used when the order of items is not important. The list items are displayed with bullet points by default. Here's an example:

...

```
<ul>
  <li>Red</li>
  <li>Green</li>
  <li>Blue</li>
</ul>
```

...

Output:

- Red
- Green
- Blue

In both ordered and unordered lists, the list items are represented by the `- ` (list item) tag. Each list item is enclosed within the opening and closing `- ` tags.

By default, the browser provides a default style for lists, but you can customize the appearance of lists using CSS. This includes changing the bullet style, numbering format, indentation, and more.

Lists are commonly used for navigation menus, table of contents, steps or instructions, features or bullet points, and other structured content. They help improve the readability and organization of information on web pages.

3.What is Void element? Explain a void element of HTML with at least three attributes.

A void element in HTML is an element that doesn't have a closing tag. Void elements are self-closing, meaning they don't contain any content or nested elements. They are used to insert specific elements or entities into a document without requiring a closing tag.

Void elements in HTML are declared using a self-closing syntax or an empty tag syntax. Here's an example of a void element with three attributes:

```
...  
  
...
```

In this example:

- ```` is the void element used to insert an image into an HTML document.
- ``src="image.jpg"`` is the attribute that specifies the source URL or file path of the image.
- ``alt="Example Image"`` is the attribute that provides alternative text for the image, which is displayed if the image cannot be loaded or for accessibility purposes.
- ``width="300"`` and ``height="200"`` are attributes that define the width and height dimensions of the image, respectively.


Other examples of void elements include ``
``, ``<hr>``, ``<input>``, and ``<meta>`. Void elements are typically used for elements that don't require additional content, such as line breaks, horizontal rules, input fields, and meta information.

It's important to note that in HTML5, the closing slash (`/`) is optional for void elements. The self-closing syntax ``<element />`` is allowed but not required. The example provided above uses the self-closing syntax for consistency and compatibility with older HTML versions.

4. Write HTML code snippets that display the following table:

6

Purchased informations of a company

Purchased Equipment			
Item No.	Item Image	Description	Price
		Shipping	Expense
1		IBM	\$400
		Shipping	\$40
<i>Total Cost</i>			\$440

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>

<body>
  <caption>Purchased information of a company</caption>
  <table border="1" style="border-collapse: collapse;">
    <tr>
      <th colspan="4">Purchased Equipment</th>
    </tr>
    <tr>
      <th rowspan="2">Item No.</th>
      <th rowspan="2">Item Image</th>
      <th>Description</th>
      <th>Price</th>
    </tr>
    <tr>
      <th>Shipping</th>
      <th>Expense</th>
    </tr>
    <tr>
      <td rowspan="2">1</td>
      <td rowspan="2"></td>
      <td>IBM</td>
      <td>$400</td>
    </tr>
    <tr>
      <td>Shipping</td>
      <td>$40</td>
    </tr>
    <tr>
      <th colspan="3"><u>Total Cost</u></th>
      <th>$440</th>
    </tr>
  </table>
</body>

</html><< endl;
}

```

5. What is marquee in HTML? Describe the process how an image can be used to create a hyperlink.

The `<marquee>` element in HTML is used to create a scrolling or moving text or content within a web page. It allows you to animate text horizontally or vertically, creating an attention-grabbing effect. However, it is important to note that the `<marquee>` element is deprecated in HTML5 and should be avoided for modern web development. Instead, CSS animations and transitions are recommended for achieving similar effects.

Regarding your second question, an image can be used to create a hyperlink by wrapping it with an `<a>` (anchor) element. Here's the process:

1. Choose the image you want to use. For example, let's say you have an image named "example-image.jpg".
2. Use the `<a>` element to create a hyperlink. Set the `href` attribute to the URL you want to link to. For example, "https://www.example.com":

```
...  
<a href="https://www.example.com">  
  <!-- Insert the image here -->  
</a>  
...
```

3. Place the `` element inside the `<a>` element to insert the image. Set the `src` attribute to the path or URL of the image file. Additionally, you can use the `alt` attribute to provide alternative text for the image:

```
...  
<a href="https://www.example.com">  
    
</a>  
...
```

4. Save the HTML file and open it in a web browser. Now, when the user clicks on the image, they will be redirected to the specified URL.

By wrapping the `` element with the `<a>` element, you have created an image hyperlink. This is commonly used for creating clickable images, such as logos, banners, or buttons, that direct users to specific web pages or resources when clicked.

6. Design the following form by using HTML:

6

Personal Details

Name:

Gender : ☐ Male ☐ Female

Email:

Date of Birth: 

City: ▼

Address :

```
<form action="">
  <fieldset>
    <legend>Personal Details</legend>
    Name: <input type="text" placeholder="Your name"> <br>
    Gender: <input type="radio" name="male" value="male"> Male
    <input type="radio" name="female" value="female"> Female <br>
    Email: <input type="email"> <br>
    Date of Birth: <input type="date"> <br>
    <label>City:</label>
    <select name="city">
      <option value="kushtia" selected>Kushtia</option>
      <option value="kushtia">dinajpur</option>
      <option value="kushtia">rangpur</option>
    </select> <br>
    Address: <br>
    <textarea name="address" cols="30" rows="8">Write your address
here</textarea>
    <br>
    <input type="submit">

  </fieldset>
</form>
```

7.What is an anchor in HTML? How can you link a page to another page in HTML? Describe with code snippets.

In HTML, an anchor is represented by the <a> element, which stands for "anchor." The <a> element is used to create hyperlinks or links within an HTML document. It allows you to navigate

from one web page to another, link to specific sections within a page, or link to external resources.

To link one page to another in HTML, you need to specify the target URL or file path within the href attribute of the <a> element. Here's an example:

```
<a href="page2.html">Go to Page 2</a>
```

In this example, when a user clicks on the link labeled "Go to Page 2," they will be directed to the "page2.html" file in the same directory.

If the target page is located in a different directory, you need to provide the appropriate file path. For instance:

```
<a href="subfolder/page3.html">Go to Page 3</a>
```

In this case, when the link is clicked, the user will be directed to the "page3.html" file located within the "subfolder" directory relative to the current page.

8. Write down the code segment to display the following output using HTML: Favourite Players in <u>World Football</u> 1. Luka Modric 2. Leo Messi 3. Luka Modric	4
--	---

```
<p>Favourite players in <b><u>World Football</u></b></p>  
<ol>  
<li>Luka Modric</li>  
<li>Leo Messi</li>  
<li>Luka Modric</li>  
</ol>
```

9. How can you give the option of downloading a file on your web page? Explain.	5
--	---

To give users the option to download a file from your web page, you can create a download link using the <a> (anchor) element with the download attribute. The download attribute allows you to specify the filename that the user will see when they download the file. Here's how you can do it:

Place the file you want to offer for download in a publicly accessible location on your server.

Create a link using the `<a>` element and set the href attribute to the URL or file path of the file you want to download.

Add the download attribute to the `<a>` element and specify the desired filename for the downloaded file. This filename can include an extension that matches the actual file type.

Here's an example:

```
<a href="/path/to/file.pdf" download="document.pdf">Download PDF</a>
```

In this example, when the user clicks on the "Download PDF" link, the file located at "/path/to/file.pdf" will be downloaded with the filename "document.pdf".

10. Mention five attributes of HTML images with sample codes.	5
---	---

Here's an example of an image tag with multiple attributes and their descriptions:

...

```

```

...

- **src**: Specifies the URL or file path of the image to be displayed (`image.jpg` in this example).
- **alt**: Provides alternative text that describes the image, which is useful for screen readers or if the image cannot be displayed.
- **width**: Specifies the width of the image in pixels or as a percentage of the parent container (set to `300px` in this example).
- **height**: Specifies the height of the image in pixels or as a percentage of the parent container (set to `200px` in this example).
- **title**: Provides additional information or a tooltip that is displayed when the user hovers over the image.

Remember to replace `image.jpg` with the actual URL or file path of your image, and modify the attribute values according to your desired dimensions and descriptions.

11. What is Tag in HTML? How can you differentiate between tag and attribute?	1+4
---	-----

In HTML, a tag is a markup element used to define the structure and content of a web page. It consists of angle brackets (`<` and `>`) surrounding the tag name. Tags are used to create elements such as headings, paragraphs, links, images, lists, and more.

On the other hand, attributes provide additional information about an HTML element. They are used within the opening tag of an element and provide specific instructions or settings for that element. Attributes are specified using the attribute name followed by an equals sign (=) and the attribute value, enclosed in quotation marks.

Here's an example to illustrate the difference between a tag and an attribute:

```
...  
<a href="https://www.example.com">Click here</a>  
...
```

In this example:

- ``<a>`` is the tag, specifically the anchor tag. It defines a hyperlink and is used to create a link to another web page.
- ``href`` is the attribute of the ``<a>`` tag. It specifies the URL ("`https://www.example.com`") that the link should navigate to.
- "Click here" is the content enclosed between the opening and closing tags. It represents the clickable text that will be displayed on the web page.

The ``<a>`` tag creates the link, while the ``href`` attribute provides the destination URL for that link. Together, they form a clickable hyperlink that displays "Click here" on the page and directs the user to "`https://www.example.com`" when clicked.

So, the tag defines the element, in this case, an anchor for a hyperlink, while the attribute provides additional information, such as the URL, to control the behavior or appearance of that element.

12. How can you take data from users using Checkbox and Radio button Control of HTML form? Explain

1. Checkbox Control:

To allow users to select multiple options, use the ``<input type="checkbox">`` element. Each checkbox should have a unique ``name`` attribute to identify the selected options. Here's a short example:

```
...  
<form>  
  <input type="checkbox" name="option1" value="Option 1"> Option 1<br>  
  <input type="checkbox" name="option2" value="Option 2"> Option 2<br>  
  <input type="checkbox" name="option3" value="Option 3"> Option 3<br>  
  <input type="submit" value="Submit">  
</form>  
...
```

In this example, users can select one or more options by checking the checkboxes. When the form is submitted, the selected options are sent as an array to the server with the respective `name` attribute values.

2. Radio Button Control:

To allow users to select a single option, use the `☐

```
...  
<form>  
  <input type="radio" name="option" value="Option 1"> Option 1<br>  
  <input type="radio" name="option" value="Option 2"> Option 2<br>  
  <input type="radio" name="option" value="Option 3"> Option 3<br>  
  <input type="submit" value="Submit">  
</form>  
...
```

In this example, users can select only one option from the radio button group. When the form is submitted, the value of the selected radio button is sent to the server with the `name` attribute value.

By using checkboxes and radio buttons in an HTML form, you provide users with the ability to make selections based on their preferences. The selected options are then sent to the server for further processing or storage.

The value attribute in HTML is used to specify the value associated with an input element. In the context of checkboxes and radio buttons, the value attribute defines the value that gets submitted to the server when the form is submitted.

13. i)

Table-1 This is a demo table

Name	Bill Gates
Telephone	423421
	31523412

```

<table border="2" cellspacing="10" cellpadding="10">
  <thead width="400px">
    Table-1 This is a demo table
  </thead>
  <tbody>
    <tr>
      <th width="200px">Name</th>
      <td width="200px">Bill Gates</td>
    </tr>
    <tr>
      <th rowspan="2" width="200px">Telephone</th>
      <td width="200px">423421</td>
    </tr>
    <tr>
      <td width="200px">31523412</td>
    </tr>
  </tbody>
</table>

```

Ques

14. Write HTML code snippets that display the following table. [See Carefully]

S	Name	Education	Occupation	Income
1	Boden	Primary	Farmer	€800
2	Alen	Habilitation & Full Prof		€10000
3	Aong	PhD	Professor	€12000
Total Income				€13800


```
<table border="1" style="border-collapse: collapse;" cellpadding="10">
  <tbody>
    <tr>
      <th>S</th>
      <th>Name</th>
      <th>Education</th>
      <th>Occupation</th>
      <th>Income</th>
    </tr>
    <tr>
      <td>1</td>
      <td>Boden</td>
      <td>Primary</td>
      <td>Farmer</td>
      <td>E800</td>
    </tr>
    <tr>
      <td>2</td>
      <td>Alen</td>
      <td colspan="2">Habilitation and full prof</td>
      <td>E10000</td>
    </tr>
    <tr>
      <td>3</td>
      <td>Aono</td>
      <td>PhD</td>
      <td>Professor</td>
      <td>E1200</td>
    </tr>
    <tr>
      <th colspan="4" align="right">Total Income</th>
      <td>E13800</td>
    </tr>
  </tbody>
</table>
```

20. Write HTML code to design the table shown below

Gender	Average	
	Height(feet)	Weight(kg)
Male	5.5	55
Female	5.2	52

```
<table border="1" style="border-collapse: collapse;" cellpadding="10">
  <tbody>
    <tr>
      <th rowspan="2">Gender</th>
      <th colspan="2">Average</th>
    </tr>
    <tr>
      <td>height( feet)</td>
      <td> Weight(kg)</td>
    </tr>
    <tr>
      <td>Male</td>
      <td>5.5</td>
      <td>55</td>
    </tr>
    <tr>
      <td>Female</td>
      <td>5.2</td>
      <td>52</td>
    </tr>
  </tbody>
</table>
```


25.What is table? Write the HTML code to design the table shown below

Purchased Equipment			
Item No	Item Image	Description	Price
		Shipping	Expense
1		IBM	\$400
		Shipping	\$40
Total Cost			\$440

```
<table border="1" style="border-collapse: collapse;" cellpadding="10">
  <tbody>
    <tr>
      <th colspan="4">Purchased Equipment</th>
    </tr>
    <tr>
      <td rowspan="2">Item no. </td>
      <td rowspan="2">Item Image</td>
      <td>Description</td>
      <td>Price</td>
    </tr>
    <tr>
      <td>Shipping</td>
      <td>Expense</td>
    </tr>
    <tr>
      <td rowspan="2">1</td>
      <td rowspan="2"></td>
      <td>IBM</td>
      <td>$400</td>
    </tr>
    <tr>
      <td>Shipping</td>
      <td>$40</td>
    </tr>
    <tr>
      <th colspan="3" align="right">Total Cost</th>
      <td>$440</td>
    </tr>
  </tbody>
</table>
```

CSS

1How do you link external CSS file with HTML. Briefly discuss with proper example.	5
--	---

To link an external CSS file to an HTML document, you can use the ``<link>`` element within the ``<head>`` section of your HTML file. Here's how you can do it:

1. Create a CSS file:

First, create a separate CSS file with a .css extension. For example, let's create a file named "styles.css" and define some styles within it.

```
...  
/* styles.css */  
body {  
    background-color: lightblue;  
}  
  
h1 {  
    color: red;  
}  
...
```

2. Link the CSS file to your HTML file:

Open your HTML file and place the following code within the ``<head>`` section. Use the ``href` attribute to specify the path or URL of your CSS file.`

```
...  
<!DOCTYPE html>  
<html>  
<head>  
    <link rel="stylesheet" type="text/css" href="styles.css">  
</head>  
<body>  
    <h1>Hello, CSS!</h1>  
</body>  
</html>  
...
```

In this example, the CSS file "styles.css" is linked to the HTML document using the ``<link>`` element. The ``rel` attribute is set to "stylesheet" to indicate that it is a style sheet. The `type` attribute specifies the MIME type of the linked file, which is "text/css" for CSS files. Lastly, the `href` attribute provides the path or URL to the CSS file.`

The CSS styles defined in "styles.css" will now be applied to the HTML elements within your HTML document. In this case, the background color of the body element will be light blue, and the color of the h1 element will be red.

By linking an external CSS file to your HTML document, you can separate the styling concerns from the HTML structure, making your code more organized and maintainable. It also allows for the reuse of styles across multiple web pages.

2. Apply the following CSS attributes to design HTML elements: (You can design whatever HTML element/s you want):

- i. **overflow/background-color**
- ii. **padding/text-transform**

Soln:

Here's an example of applying the mentioned CSS attributes to design HTML elements:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    /* Apply CSS attributes */
    .box {
      overflow: hidden;
      background-color: lightblue;
      padding: 20px;
      text-transform: uppercase;
    }
  </style>
</head>
<body>
  <div class="box">
    <h1>This is a Heading</h1>
    <p>This is a paragraph with some text.</p>
  </div>
</body>
</html>
```

In this example, we have created a `

` element with the class "box" and applied the following CSS attributes:

- `overflow: hidden;`: This attribute specifies that if the content of the `

` exceeds its dimensions, it should be clipped and hidden. This is useful when you want to hide any overflowing content.

- `background-color: lightblue;`: This attribute sets the background color of the `<div>` to light blue.
- `padding: 20px;`: This attribute adds 20 pixels of padding to all sides of the `<div>`. Padding creates space between the content and the border of the element.
- `text-transform: uppercase;`: This attribute transforms the text inside the `<div>` to uppercase, making it all capital letters.

By applying these CSS attributes to the `<div>` element, the background color of the box will be light blue, and it will have 20 pixels of padding on all sides. The text inside the box, including the heading and paragraph, will be transformed to uppercase.

Feel free to adjust the CSS attributes and HTML structure as needed to match your desired design.

3.What is the difference between a class and an ID selector? Construct CSS style rule syntax with a proper example?

In CSS, both class selectors and ID selectors are used to target and apply styles to specific elements, but they have some key differences:

1. Class Selector:

- Starts with a dot (`.`) followed by the class name.
- Can be applied to multiple elements on the same page.
- Can be reused across different elements.
- Allows for grouping and styling of related elements.
- Supports multiple class names for a single element, separated by spaces.
- Example: `.my-class { color: blue; }`

2. ID Selector:

- Starts with a hash (`#`) followed by the ID name.
- Should be unique within the entire HTML document.
- Should only be used for a single element on the page.
- Typically used for targeting specific elements with unique styles or behavior.
- Provides high specificity in CSS, overriding other selectors with lower specificity.
- Example: `#my-id { font-size: 20px; }`

CSS style rule syntax examples:

1. Using a class selector:

...

```
.my-class {
```

```
color: blue;
font-size: 16px;
}
```

In this example, the `.my-class` selector targets all elements with the class "my-class" and applies a blue color and font size of 16 pixels to them.

2. Using an ID selector:

```
#my-id {
background-color: yellow;
border: 1px solid black;
}
```

In this example, the `#my-id` selector targets the element with the ID "my-id" and sets a yellow background color and a 1-pixel black solid border for that specific element.

It's important to note that class selectors are generally more flexible and reusable, while ID selectors are typically used for targeting unique elements with specific styles or functionality. It's best practice to avoid using the same ID for multiple elements on the same page to ensure valid HTML markup.

4. Describe three types of selectors in CSS with proper examples.	5
--	----------

Certainly! Here are three types of selectors commonly used in CSS, along with examples:

1. Element Selectors:

- Target HTML elements based on their tag names.
- Apply styles to all instances of the specified element.
- Example: `p { color: blue; }`
- In this example, the `<p>` elements will have a blue color.

2. Class Selectors:

- Target elements based on their class attribute.
- Apply styles to elements that have a specific class.
- Example: `.highlight { background-color: yellow; }`
- In this example, any element with the class "highlight" will have a yellow background color.

3. ID Selectors:

- Target elements based on their unique ID attribute.
- Apply styles to a specific element with a given ID.

- Example: `#logo { width: 200px; }`
- In this example, the element with the ID "logo" will have a width of 200 pixels.

These are just a few examples of selector types. CSS provides various other selectors, such as attribute selectors, pseudo-classes, and pseudo-elements, which offer more specific targeting options. Using a combination of selectors allows for fine-grained control over styling elements on a web page.

5. Describe the ways by which CSS can be integrated into a HTML page.	5
--	----------

CSS can be integrated into an HTML page using various methods. Here are the most common ways to include CSS styles in an HTML document:

1. Inline CSS:

- Inline styles are directly applied to individual HTML elements using the `style` attribute.
- Example: `<h1 style="color: blue;">Hello, CSS!</h1>`
- In this example, the color of the `<h1>` element is set to blue using inline CSS.

2. Internal CSS:

- Internal styles are defined within the `<style>` element in the `<head>` section of the HTML document.
- Example:

```
<head>
  <style>
    h1 {
      color: blue;
    }
  </style>
</head>
<body>
  <h1>Hello, CSS!</h1>
</body>
```

- In this example, the `<h1>` element is styled to have a blue color using internal CSS.

3. External CSS:

- External stylesheets are separate CSS files linked to the HTML document using the `<link>` element.

- Example:

```
...  
<head>  
  <link rel="stylesheet" type="text/css" href="styles.css">  
</head>  
<body>  
  <h1>Hello, CSS!</h1>  
</body>  
...
```

- In this example, the CSS styles are defined in an external file named "styles.css" and linked to the HTML document using the ``<link>`` element.

Using external CSS files is the preferred method for larger projects as it promotes separation of concerns and allows for better organization and reusability of styles. Inline and internal CSS are useful for quick styling or when specific styles need to be applied to individual elements.

It's important to note that the order of CSS integration matters. Inline styles take precedence over internal styles, and both inline and internal styles can be overridden by styles in an external CSS file.

6. Write down the meaning of the following CSS statements: i. <code>h1, .logo, #val {font-weight: italic; font-family:courier}</code> ii. <code>*p{color: red} {margin: 5px 10px 7px}</code>	5
---	---

Here's the meaning of the provided CSS statements:

i. ``h1, .logo, #val {font-weight: italic; font-family: courier;}``

- This statement targets three different selectors: ``h1``, elements with the class `` .logo``, and the element with the ID ``#val``.

- It applies the following styles to the selected elements:

- ``font-weight: italic;``: Sets the font weight to italic.

- ``font-family: courier;``: Sets the font family to Courier.

ii. ``*p {color: red; margin: 5px 10px 7px;}``

- This statement targets all paragraph elements (``<p>``) within any parent element using the universal selector ``*`` in combination with the ``p`` element selector.

- It applies the following styles to the selected paragraph elements:

- ``color: red;``: Sets the text color to red.

- ``margin: 5px 10px 7px;``: Sets the margins to 5 pixels (top), 10 pixels (right and left), and 7 pixels (bottom) for the selected paragraphs.

It's important to note that the closing curly braces (``) are required to properly close the CSS rules and separate different styles. The semicolon (;) is used to separate multiple property-value pairs within a single CSS rule.

7. What are the meanings of the following CSS statements: i. <code>h1, .link, #top {font-weight: italic; font-family: courier}</code> ii. <code>img {margin: 7px 15px 8px;}</code> iii. <code>p > .err { font-family: arial; font-size: 10px; fontweight:bold;}</code>	5
---	---

Sure! Here's the meaning of the provided CSS statements:

i. `h1, .link, #top {font-weight: italic; font-family: courier}`

- This statement targets three different selectors: `h1`, elements with the class `.link`, and the element with the ID `#top`.
- It applies the following styles to the selected elements:
 - `font-weight: italic;`: Sets the font weight to italic.
 - `font-family: courier;`: Sets the font family to Courier.

ii. `img {margin: 7px 15px 8px;}`

- This statement targets all `` elements.
- It applies the following styles to the selected images:
 - `margin: 7px 15px 8px;`: Sets the margins to 7 pixels (top), 15 pixels (right and left), and 8 pixels (bottom) for the selected images.

iii. `p > .err { font-family: arial; font-size: 10px; font-weight: bold; }`

- This statement targets elements with the class `.err` that are direct children (`>`) of `<p>` elements.
- It applies the following styles to the selected elements:
 - `font-family: arial;`: Sets the font family to Arial.
 - `font-size: 10px;`: Sets the font size to 10 pixels.
 - `font-weight: bold;`: Sets the font weight to bold.

In summary, these CSS statements define styles for various selectors. The first statement targets `h1`, elements with the class `.link`, and the element with the ID `#top`. The second statement targets all `` elements. The third statement targets elements with the class `.err` that are direct children of `<p>` elements.

8.How do you link an external CSS file with an HTML? Describe with a proper example.	
--	--

To link an external CSS file with an HTML document, you can use the ``<link>`` element in the ``<head>`` section of your HTML file. Here's an example:

1. Create your CSS file:

Create a separate CSS file with the desired styles. Let's name it "styles.css" for this example.

2. Place the CSS file in the appropriate location:

Make sure your CSS file is located in the same directory as your HTML file or provide the correct file path if it's in a different location.

3. Link the CSS file to your HTML document:

In the ``<head>`` section of your HTML file, use the ``<link>`` element with the ``rel``, ``type``, and ``href`` attributes to specify the CSS file you want to link.

Example:



```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
  <!-- Your HTML content goes here -->
</body>
</html>
```

In this example, we've used the ``<link>`` element to link the "styles.css" file to the HTML document. The ``rel`` attribute specifies the relationship of the linked file, which is "stylesheet" in this case. The ``type`` attribute specifies the MIME type of the linked file, which is "text/css" for CSS files. The ``href`` attribute specifies the path or URL of the CSS file.

Make sure to replace "styles.css" with the actual file name and path if it's in a different location. By linking the CSS file to your HTML document in this way, the styles defined in the CSS file will be applied to the corresponding HTML elements.

9. Apply the following CSS attributes to design HTML elements: (You can design whatever HTML element/s you want)

- i. overflow
- ii. opacity
- iii. padding

Here's an example of applying the CSS attributes `overflow`, `opacity`, and `padding` to design HTML elements:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .box {
      width: 200px;
      height: 200px;
      background-color: #f1f1f1;
      border: 1px solid #ccc;
      overflow: auto;
      opacity: 0.8;
      padding: 20px;
    }
  </style>
</head>
<body>
  <div class="box">
    <h2>Overflow Example</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <p>Sed ut perspiciatis unde omnis iste natus error sit voluptatem
accusantium doloremque laudantium.</p>
    <p>Eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae
vitae dicta sunt explicabo.</p>
  </div>
</body>
</html>
```

In this example, we have created a `

` element with a class of "box" to showcase the CSS attributes. Here's what each attribute does:

- i. `overflow: auto;`
 - Specifies how content that exceeds the size of the element should be handled.
 - The `auto` value allows the browser to add scrollbars when necessary if the content exceeds the element's dimensions.

ii. ``opacity: 0.8;``

- Sets the opacity or transparency level of the element.
- The value ranges from 0.0 (completely transparent) to 1.0 (fully opaque).
- In this example, the opacity is set to 0.8, making the element slightly transparent.

iii. ``padding: 20px;``

- Specifies the padding or space between the content of the element and its border.
- The value can be in pixels, percentages, or other length units.
- In this example, the padding is set to 20 pixels, creating space around the content within the element.

Feel free to customize the example and apply these attributes to other HTML elements as per your requirements.

11. Write down the meaning of the following CSS statements:	
--	--

- | | |
|--|--|
| <ul style="list-style-type: none">i. <code>h1, .logo, #val {font-weight: italic; font-family: courier;}</code>ii. <code>img[alt~=banner] {margin: 5px 10px 7px;}</code> | |
|--|--|

Sure! Here's the meaning of the provided CSS statements:

i. ``h1, .logo, #val {font-weight: italic; font-family: courier;}``

- This statement targets three different selectors: ``h1``, elements with the class ``logo``, and the element with the ID ``#val``.
- It applies the following styles to the selected elements:
 - ``font-weight: italic;``: Sets the font weight to italic.
 - ``font-family: courier;``: Sets the font family to Courier.

ii. ``img[alt~=banner] {margin: 5px 10px 7px;}``

- This statement targets ```` elements that have an ``alt`` attribute with a value containing the word "banner".
- It applies the following style to the selected images:
 - ``margin: 5px 10px 7px;``: Sets the margins to 5 pixels (top), 10 pixels (right and left), and 7 pixels (bottom) for the selected images.

In summary, these CSS statements define styles for specific selectors. The first statement targets ``h1``, elements with the class ``logo``, and the element with the ID ``#val`` to apply italic font weight and the Courier font family. The second statement targets ```` elements with an ``alt`` attribute containing the word "banner" and sets the specified margins for those images.

12. What is CSS? Explain the basic format to write the CSS document with proper example.	
--	--

CSS (Cascading Style Sheets) is a stylesheet language used to describe the presentation and formatting of HTML or XML documents. It separates the design and style aspects from the content, allowing developers to control the visual appearance of web pages.

The basic format of a CSS document includes selectors and declarations. Selectors target specific elements, classes, or IDs, while declarations define the styles to be applied to those targets. Each declaration consists of a property and a value.

CSS rules can be placed inside `<style>` tags within the `<head>` section of an HTML document or in an external CSS file linked to the HTML file. Styles can be applied to various elements using element selectors, class selectors, ID selectors, and attribute selectors.

By using CSS, you can customize the colors, fonts, layout, spacing, and other visual aspects of HTML elements, providing a consistent and appealing design for your web pages.

Here's a short example of a CSS document:

```
...
h1 {
  color: blue;
  font-size: 24px;
}

.logo {
  width: 200px;
  height: 150px;
  background-image: url("logo.png");
}

#top {
  background-color: #f1f1f1;
  border: 1px solid #ccc;
}
...
```

In this example, we have three CSS rules:

1. The `h1` selector targets all `<h1>` elements and applies a blue color and a font size of 24 pixels.
2. The `.logo` selector targets elements with the class "logo" and applies a width of 200 pixels, height of 150 pixels, and sets a background image.

3. The `#top` selector targets the element with the ID "top" and applies a background color of #f1f1f1 and a 1-pixel solid border.

These rules can be placed within the `<style>` tags in the `<head>` section of an HTML document or in an external CSS file linked to the HTML file. By applying these styles, you can control the appearance of HTML elements, such as heading tags, elements with a specific class, or elements with a specific ID, as shown in the example.

13. Write syntaxes for the ways CSS code can be added to an HTML file.	
---	--

Same as 5

14. Why should you use the CSS in web programming.	
---	--

CSS (Cascading Style Sheets) is widely used in web programming for several important reasons:

1. **Separation of Concerns:** CSS allows for a clear separation of presentation and content. By separating the visual styling from the HTML structure, it promotes cleaner and more maintainable code. This separation makes it easier to update the visual design of a website without modifying the underlying HTML structure.
2. **Consistent Styling:** CSS enables consistent styling across multiple web pages. By defining styles in a central CSS file, you can easily apply those styles to different elements throughout your website. This consistency ensures a cohesive and professional look and feel across all pages.
3. **Easy Maintenance:** With CSS, you can make global changes to the design of your website with minimal effort. Instead of modifying each HTML element individually, you can update the CSS styles once, and the changes will be applied throughout the website. This saves time and effort when making design updates or fixing styling issues.
4. **Responsive Design:** CSS plays a crucial role in creating responsive web designs. It allows you to apply different styles based on the device's screen size, enabling your website to adapt and look good on various devices, such as desktops, tablets, and mobile phones. CSS media queries, along with other techniques, make it possible to create responsive layouts and optimize the user experience.
5. **Efficiency and Performance:** CSS helps improve the performance of web pages by reducing the file size and load time. By applying styles through CSS classes and IDs, you can avoid

repetitive inline styles, resulting in smaller HTML files. Additionally, CSS provides various optimization techniques like combining and minifying stylesheets, reducing HTTP requests, and implementing caching strategies to enhance overall website performance.

6. Browser Compatibility: CSS is widely supported across different web browsers, ensuring consistent rendering of styles. It helps create cross-browser compatible websites by providing a standardized way to define and apply styles.

In summary, using CSS in web programming offers benefits such as separation of concerns, consistent styling, easy maintenance, responsive design, improved performance, and browser compatibility. It enhances the efficiency and flexibility of web development, enabling developers to create visually appealing and accessible websites.

Javascript

1.What is an Event in JavaScript? Explain two events with a proper example.

In JavaScript, an event refers to an action or occurrence that happens within the browser. These events can be triggered by the user, the browser, or the web page itself. JavaScript allows you to listen for these events and execute specific code or functions in response to them.

Here are two commonly used events in JavaScript with examples:

1. Click Event:

The click event is triggered when the user clicks on an HTML element. It is widely used for various interactive functionalities. Here's an example:

```
...  
<button id="myButton">Click Me</button>  
  
<script>  
  // Adding a click event listener to the button  
  document.getElementById("myButton").addEventListener("click", function() {  
    alert("Button clicked!");  
  });  
</script>  
...
```

In this example, when the user clicks the "Click Me" button, the click event is triggered, and the associated event listener function is executed. In this case, an alert box will appear with the message "Button clicked!"

2. Keydown Event:

The keydown event is triggered when a key on the keyboard is pressed down. It can be used to capture user input or implement keyboard shortcuts. Here's an example:

```
...  
<input type="text" id="myInput">  
  
<script>  
  // Adding a keydown event listener to the input field  
  document.getElementById("myInput").addEventListener("keydown", function(event) {  
    console.log("Key pressed:", event.key);  
  });  
</script>  
...
```

In this example, when the user presses a key while the focus is on the input field, the keydown event is triggered, and the associated event listener function is executed. The event object is passed to the function, which can be used to access information about the key that was pressed. In this case, the pressed key will be logged to the console.

These are just two examples of events in JavaScript, but there are many more events available that can be used to create interactive and dynamic web pages. Events allow you to respond to user actions or other interactions within the browser, enabling you to add interactivity and enhance the user experience of your web applications.

2. Write a code in JavaScript using while loop that print the following output:
1 2 5 10 17.....82

```
let num = 1, increment = 1, res = '';  
while (num <= 82) {  
  res += (num + ' ');  
  num += increment;  
  increment += 2;  
}  
document.write(res);
```

3. Define the scope of JavaScript variables. Explain its types with JavaScript program.	1+4
---	-----

The scope of a variable in JavaScript refers to the part of the program where the variable is accessible. The scope determines the visibility and lifetime of a variable.

In JavaScript, there are three types of variable scope:

1. Global Scope:

Variables declared outside any function, block, or module have global scope. They can be accessed from anywhere within the JavaScript program, including all functions, blocks, and nested scopes. Here's an example:

```
...  
  
// Global scope variable  
var globalVariable = "I'm a global variable";  
  
function foo() {  
  console.log(globalVariable); // Accessible within functions  
}  
  
foo(); // Outputs: I'm a global variable  
...
```

In this example, the `globalVariable` is declared outside any function and can be accessed from both the global scope and the `foo()` function.

2. Local Scope (Function Scope):

Variables declared inside a function have local scope. They are accessible only within that function and any nested functions. Here's an example:

```
...  
  
function myFunction() {  
  var localVariable = "I'm a local variable";  
  console.log(localVariable); // Accessible within the function  
}  
  
myFunction(); // Outputs: I'm a local variable  
console.log(localVariable); // Throws an error: localVariable is not defined  
...
```

In this example, the `localVariable` is declared inside the `myFunction()` function and can only be accessed within that function.

3. Block Scope (Introduced in ES6):

Variables declared with `let` and `const` keywords have block scope. They are accessible only within the block in which they are defined, such as within an `if` statement, `for` loop, or a block enclosed by curly braces `{}`. Here's an example:

```
...  
  
if (true) {  
  let blockVariable = "I'm a block variable";  
  console.log(blockVariable); // Accessible within the block  
}  
  
console.log(blockVariable); // Throws an error: blockVariable is not defined  
...
```

In this example, the `blockVariable` is declared within the `if` block and can only be accessed within that block.

It's important to note that variables declared without the `var`, `let`, or `const` keywords are automatically assigned global scope, which can lead to unintended consequences. It's generally recommended to use `let` or `const` to declare variables and limit their scope to the smallest necessary scope.

Understanding variable scope in JavaScript is crucial for managing variable access and preventing naming conflicts within your code.

4. Write a program using JavaScript Function which shows the message “I am the student of CSE department in HSTU” after clicking a button.

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link rel="stylesheet" href="">
</head>

<body>
  <input type="button" onclick="foo()" value="click me!">

  <script>
    function foo() {
      alert('I am the student of CSE department of HSTU');
    }
  </script>
</body>

</html>
```

5. How can JavaScript be implemented in a web page? Explain the attributes of <script> tag.

JavaScript can be implemented in a web page by using the ``<script>`` tag. The ``<script>`` tag is used to embed or reference JavaScript code within an HTML document. It can be placed in the ``<head>`` section or the ``<body>`` section of the HTML file.

Here are the commonly used attributes of the ``<script>`` tag:

1. ``src`` attribute:

The ``src`` attribute is used to specify the source file (URL) of an external JavaScript file. It allows you to separate your JavaScript code into a separate file for better organization and reusability. Here's an example:

```

...
<script src="script.js"></script>
...

```

In this example, the JavaScript code from the "script.js" file will be executed.

2. `type` attribute:

The `type` attribute specifies the MIME type of the scripting language being used. For JavaScript, the value should be set to "text/javascript". Although this attribute is not required in modern HTML5, it's still a good practice to include it. Here's an example:

```

..
<script type="text/javascript">
  // JavaScript code here
</script>
...

```

3. `async` attribute:

The `async` attribute is used to specify that the script should be executed asynchronously. It allows the HTML parsing to continue while the script is being fetched. This attribute is useful when the script does not depend on other elements on the page. Here's an example:

```

...
<script src="script.js" async></script>
...

```

It's important to note that the `async` attribute is only applicable when the `

Dynamic Webpage:

- Content is generated on the fly and can change.
- Content is generated using programming languages like JavaScript, PHP, or Python.
- The server processes the request, executes code, and interacts with databases or external sources.
- The generated HTML is sent back to the user's browser for rendering.
- Content can change based on factors such as user input, database queries, or real-time data.
- Examples include e-commerce websites, social media platforms, web applications, or news websites.
- Dynamic webpages can display personalized content, generate reports, handle user input, and provide interactive features.
- More interactive, responsive, and adaptable to user input compared to static webpages.

7.What do you mean by NULL and Undefined in JavaScript? Write a code snippet which shows the use of conditional operator.

In JavaScript, `null` and `undefined` are both special values that indicate the absence of a value. While they are similar in some ways, there are subtle differences between them:

1. `null`:

- It is an assignment value that represents the intentional absence of any object value.
- It is a primitive value and a specific value of the object type.
- It must be assigned explicitly to a variable.
- It is often used to indicate the absence of an object or to reset a variable's value.
- Example:

```
...  
  
let myVariable = null;  
console.log(myVariable); // Outputs: null  
...
```

2. `undefined`:

- It is a variable value that indicates the absence of an assigned value.
- It is a primitive value and a type itself.
- It is automatically assigned to a variable that is declared but not initialized or does not have a value assigned to it.
- It can also be explicitly assigned to a variable.
- Example:

```
...  
  
let myVariable;  
console.log(myVariable); // Outputs: undefined
```

...

As for the conditional (ternary) operator in JavaScript, it provides a concise way to write conditional statements. It takes three operands: a condition, an expression to evaluate if the condition is true, and an expression to evaluate if the condition is false.

Here's an example code snippet that demonstrates the use of the conditional operator:

...

```
let number = 10;  
let result = (number > 5) ? "Greater than 5" : "Less than or equal to 5";  
console.log(result); // Outputs: Greater than 5  
...
```

In this example, the condition `number > 5` is evaluated. If the condition is true, the value "Greater than 5" is assigned to the `result` variable. If the condition is false, the value "Less than or equal to 5" is assigned to the `result` variable. Finally, the `result` variable is logged to the console.

8. Is JavaScript a case-sensitive language? What are the variable naming conventions in JavaScript?

Yes, JavaScript is a case-sensitive language. This means that uppercase and lowercase letters are treated as distinct and separate entities in JavaScript.

When it comes to variable naming conventions in JavaScript, the following guidelines are commonly followed:

1. Use descriptive names: Choose meaningful and descriptive names for variables to make your code more readable and understandable. This helps in maintaining code clarity and makes it easier for others (including yourself) to understand the purpose and usage of the variables.
2. Use camelCase: JavaScript conventionally uses camelCase for variable names. In camelCase, the first word starts with a lowercase letter, and each subsequent word starts with an uppercase letter. This convention helps differentiate words within the variable name and improves readability. For example: `firstName`, `totalAmount`, `isLogged`.
3. Avoid reserved keywords: Avoid using reserved keywords (such as `if`, `for`, `while`, `function`, etc.) as variable names. These keywords have predefined meanings in JavaScript and should not be used for other purposes.
4. Begin with a letter or underscore: Variable names in JavaScript must begin with a letter (a-z, A-Z) or an underscore (`_`). They cannot start with a number or any other special characters.

5. Use meaningful abbreviations: If using abbreviations, make sure they are meaningful and widely understood. Avoid using cryptic or confusing abbreviations that may make the code harder to comprehend.

6. Be consistent: Maintain consistency in your variable naming conventions throughout your codebase. This helps in improving code readability and making it easier to follow.

Here are some examples of correctly formatted variable names:

```
``javascript
let firstName = "John";
let totalAmount = 100.50;
let isLoggedIn = true;
let _privateVariable = "Something";
``
```

By following these naming conventions, your JavaScript code will be more readable, maintainable, and in line with common programming practices.

9.What are the ways of linking JavaScript with web pages? Explain with example.	
---	--

There are multiple ways to link JavaScript with web pages. Here are three common methods:

1. Inline JavaScript:

In this approach, JavaScript code is directly written within the HTML file using the ``<script>`` tag. The code can be placed in the ``<head>`` section or the ``<body>`` section of the HTML file.

Example:

```
<html>
<head>
  <script>
    // Inline JavaScript code
    function greet() {
      console.log("Hello, World!");
    }
  </script>
</head>
<body>
  <button onclick="greet()">Click me</button>
</body>
</html>
```

In the above example, the JavaScript code is embedded within the ``<script>`` tags in the ``<head>`` section. The `greet()` function is called when the button is clicked, as specified by the `onclick` attribute.

2. External JavaScript file:

JavaScript code can be written in a separate file with a `.js` extension and then linked to the HTML file using the ``<script>`` tag with the `src` attribute.

Example:

```
<html>
  <head>
    <script src="script.js"></script>
  </head>
  <body>
    <button onclick="greet()">Click me</button>
  </body>
</html>
```

In this example, the JavaScript code is written in an external file named `script.js`. The ``<script>`` tag with the `src` attribute is used to link the external JavaScript file to the HTML file. The `greet()` function can be defined within the `script.js` file.

3. JavaScript event handlers:

JavaScript code can also be linked to HTML elements through event handlers. Event handlers are attributes of HTML elements that specify JavaScript code to be executed when a particular event occurs.

Example:

```
<html>
  <body>
    <button onclick="greet()">Click me</button>
    <script>
      // JavaScript code linked through event handler
      function greet() {
        console.log("Hello, World!");
      }
    </script>
  </body>
</html>
```

In this example, the JavaScript code is placed directly within the `<script>` tags within the `<body>` section. The `greet()` function is linked to the `onclick` event of the button element. When the button is clicked, the `greet()` function is executed.

10.What is JAVASCRIPT? Write the merits of using Javascript in web programming	
--	--

JavaScript is a high-level, interpreted programming language primarily used for adding interactivity and dynamic behavior to web pages. It is one of the core technologies of web development and runs on the client-side, meaning it executes directly in the user's web browser.

Merits of using JavaScript in web programming:

1. Enhanced interactivity: JavaScript allows developers to create interactive and engaging web experiences. It enables features like form validation, image sliders, interactive maps, and dynamic content updates without requiring page reloads.
2. Client-side processing: JavaScript runs on the client-side, reducing the need for server-side processing and minimizing the load on the web server. This improves performance and responsiveness, resulting in a smoother user experience.
3. Wide browser support: JavaScript is supported by all major web browsers, including Chrome, Firefox, Safari, and Edge. This ensures broad compatibility and consistent behavior across different platforms and devices.
4. Rich ecosystem and libraries: JavaScript has a vast ecosystem with a wide range of libraries, frameworks, and tools available. These resources help developers streamline their workflow, increase productivity, and build complex applications more efficiently.
5. Cross-platform development: JavaScript can be used for both web and mobile app development. With frameworks like React Native and Ionic, developers can build mobile applications using JavaScript, sharing code between web and mobile platforms, saving time and effort.
6. Asynchronous programming: JavaScript supports asynchronous programming, allowing tasks to be executed in the background without blocking other processes. This enables the development of responsive and efficient applications, such as handling AJAX requests or fetching data from APIs.
7. Easy integration with HTML and CSS: JavaScript seamlessly integrates with HTML and CSS, allowing developers to manipulate the Document Object Model (DOM) and modify the structure and styling of web pages dynamically.

8. Continuous evolution: JavaScript is constantly evolving, with regular updates to the language and new features being introduced. This ensures that developers have access to modern tools, techniques, and standards for web development.

9. Community support: JavaScript has a large and active community of developers, offering support, sharing knowledge, and contributing to open-source projects. This vibrant community provides resources, tutorials, and forums for learning and problem-solving.

Overall, JavaScript empowers web developers to create interactive and dynamic web applications with enhanced user experiences. Its versatility, broad support, and extensive ecosystem make it a powerful language for web programming.

PHP

1.What is PHP? Why PHP is a must for CSE students to build career in the Web Development domain?

PHP is a server-side scripting language designed for web development. It is widely used to create dynamic web pages, build web applications, and interact with databases. PHP stands for "PHP: Hypertext Preprocessor" and it is embedded within HTML code.

Reasons why PHP is a must for CSE students to build a career in web development:

1. Popularity and demand: PHP is one of the most popular programming languages for web development. It is used by a large number of websites and has a strong community support. Many well-established companies and organizations rely on PHP for their web applications, making it a highly in-demand skill in the job market.

2. Versatility and flexibility: PHP can be used to develop a wide range of web applications, from simple websites to complex enterprise-level systems. It supports various frameworks, content management systems (CMS), and databases, providing flexibility and scalability in application development.

3. Integration with databases: PHP has excellent integration capabilities with databases like MySQL, PostgreSQL, and Oracle. It allows developers to easily connect to databases, perform database operations, and handle data efficiently. This makes PHP a valuable skill for working with data-driven applications and managing large volumes of information.

4. Extensive documentation and community support: PHP has a vast amount of documentation, tutorials, and online resources available, making it easy for students to learn and develop their

skills. The PHP community is active and supportive, providing forums, discussion groups, and open-source projects to facilitate learning and collaboration.

5. Career opportunities: PHP skills are highly sought after in the web development industry. With a solid understanding of PHP, CSE students can pursue careers as web developers, full-stack developers, PHP programmers, or CMS developers. The demand for PHP developers is expected to continue growing, offering ample career opportunities and growth potential.

6. Easy to learn and use: PHP has a relatively simple and straightforward syntax, making it easy for beginners to grasp the basics and start building web applications quickly. It has similarities to other programming languages like C and JavaScript, which can help CSE students leverage their existing programming knowledge.

7. Cost-effectiveness: PHP is open-source and free to use, which makes it a cost-effective choice for developing web applications. It eliminates the need for expensive software licenses and reduces development costs, making it an attractive option for businesses and startups.

In summary, PHP offers a powerful and versatile platform for web development. Its popularity, integration capabilities, extensive community support, and career opportunities make it a must-have skill for CSE students interested in pursuing a career in the web development domain.

2.Explain different ways to differentiate PHP code from other elements in a web page.	5
---	---

There are several ways to differentiate PHP code from other elements in a web page:

1. PHP Opening and Closing Tags:

PHP code is typically enclosed within PHP opening and closing tags. The most common syntax is ``<?php` for the opening tag and `?>` for the closing tag. This clearly marks the PHP code and distinguishes it from other HTML or text content on the page.`

Example:

...

```
<html>
<head>
  <title>PHP Example</title>
</head>
<body>
  <h1>Welcome to my PHP page!</h1>
```

```

<?php
// PHP code goes here
echo "This is PHP code.";
?>

<p>Regular HTML content</p>
</body>
</html>
...

```

2. PHP Inline Code:

PHP code can be embedded directly within HTML elements using the `<?php ... ?>` tags. This allows for dynamic content generation and manipulation within specific HTML elements.

Example:

```

...
<p>The current year is <?php echo date("Y"); ?>.</p>
...

```

In the above example, the PHP code `date("Y")` is embedded within the HTML `<p>` element. It will be executed by the server and replaced with the current year when the page is rendered.

3. Separate PHP Files:

Another way to differentiate PHP code is by placing it in separate PHP files with a `.php` extension. These files contain PHP code exclusively, and their output is typically integrated into HTML files using PHP include or require statements.

3. Mention the types of loops used in PHP. Explain the syntax for 'foreach' loop with example?

In PHP, there are several types of loops that can be used for repetitive execution of code. The types of loops in PHP are:

1. `for` loop: Executes a block of code a specified number of times.
2. `while` loop: Executes a block of code as long as a specified condition is true.
3. `do-while` loop: Executes a block of code once, and then repeats the execution as long as a specified condition is true.
4. `foreach` loop: Iterates over elements in an array or an object.

Here is the syntax for the `foreach` loop in PHP:

```

...
foreach ($array as $value) {
    // Code to be executed for each value
}

```

```
}  
...
```

The `foreach` loop iterates over each element in the `$array` and assigns the current element's value to the `$value` variable. The code within the loop's block will be executed for each element in the array.

Example:

```
...  
$fruits = array("Apple", "Banana", "Orange");  
  
foreach ($fruits as $fruit) {  
    echo $fruit . "<br>";  
}  
...
```

In this example, the `foreach` loop iterates over each element in the `$fruits` array. On each iteration, the current fruit is assigned to the `$fruit` variable, and the code within the loop echoes the fruit name followed by a line break. The output will be:

```
...  
Apple  
Banana  
Orange  
...
```

The `foreach` loop is particularly useful when working with arrays or objects, as it automatically handles the iteration and provides an easy way to access each element's value without the need for index manipulation.

4.What is the meaning of break and continue statements in PHP? Give PHP code snippet of these statements as examples.

In PHP, the `break` and `continue` statements are used to control the flow of execution within loops.

1. `break` statement:

The `break` statement is used to exit the current loop prematurely. It is commonly used when a certain condition is met, and you want to stop the loop from executing further iterations.

Example:

```
...  
for ($i = 1; $i <= 10; $i++) {  
    if ($i == 5) {
```

```

        break; // exit the loop when $i equals 5
    }
    echo $i . " ";
}
...

```

In this example, the `for` loop will iterate from 1 to 10. However, when the value of `\$i` becomes 5, the `break` statement is encountered, and the loop is terminated. The output will be:

...

1 2 3 4

...

2. `continue` statement:

The `continue` statement is used to skip the remaining code within the current iteration of the loop and move to the next iteration. It is useful when you want to skip certain iterations based on a specific condition.

Example:

...

```

for ($i = 1; $i <= 5; $i++) {
    if ($i == 3) {
        continue; // skip the current iteration when $i equals 3
    }
    echo $i . " ";
}
...

```

In this example, the `for` loop will iterate from 1 to 5. When the value of `\$i` is 3, the `continue` statement is encountered, and the code within the current iteration is skipped. The loop continues to the next iteration. The output will be:

...

1 2 4 5

...

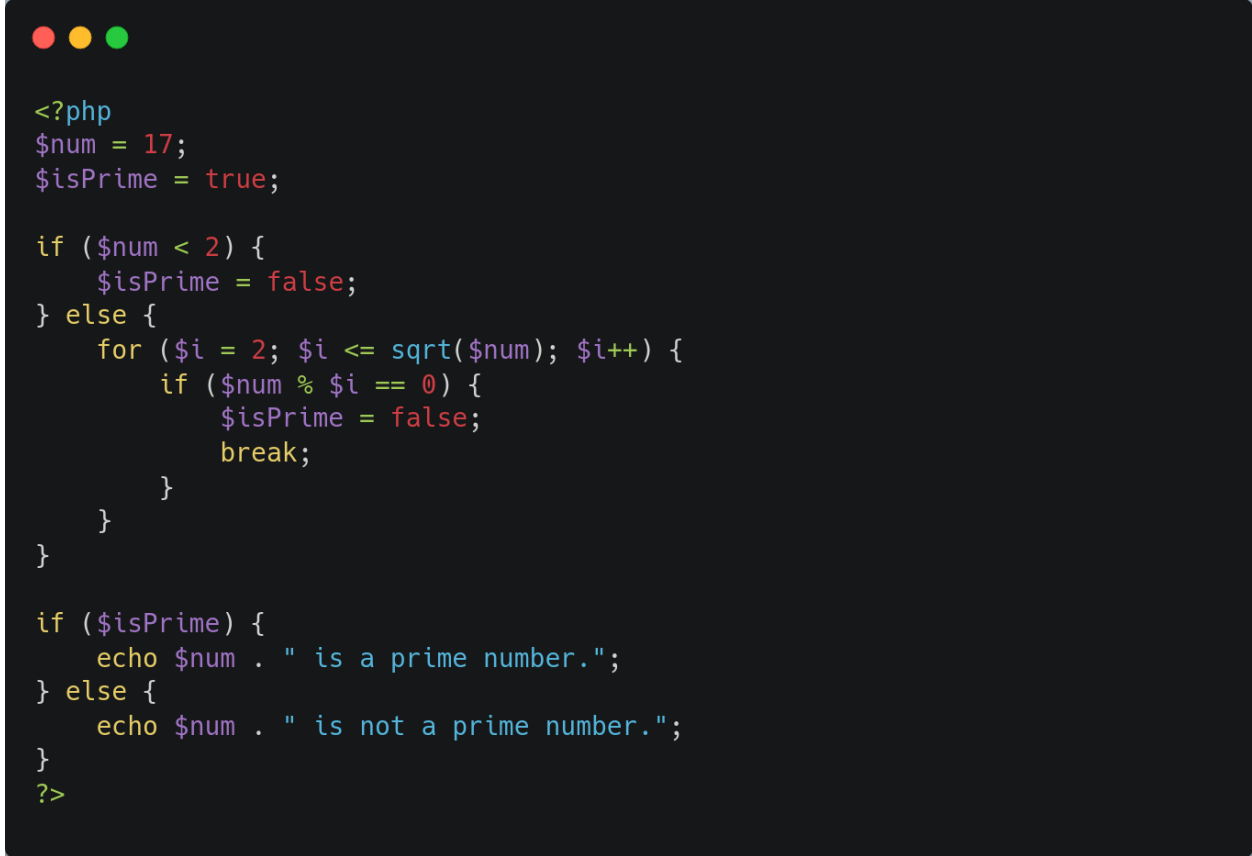
Both `break` and `continue` statements are powerful tools for controlling the flow of execution within loops and providing more flexibility in handling specific conditions during looping.

5. How can a text be printed using PHP? Write a program in PHP to check whether a number is prime or not?

To print text using PHP, you can use the `echo` statement or `print` function.

```
echo "Hello, World!";  
print "Hello, World!";
```

Here's an example of how to check whether a number is prime or not in PHP:



```
<?php  
$num = 17;  
$isPrime = true;  
  
if ($num < 2) {  
    $isPrime = false;  
} else {  
    for ($i = 2; $i <= sqrt($num); $i++) {  
        if ($num % $i == 0) {  
            $isPrime = false;  
            break;  
        }  
    }  
}  
  
if ($isPrime) {  
    echo $num . " is a prime number.";  
} else {  
    echo $num . " is not a prime number.";  
}  
?>
```

In this example, the variable `\$num` is set to the number we want to check for prime. The variable `\$isPrime` is initially set to `true`. We then check if the number is less than 2, in which case it is not prime. Otherwise, we iterate from 2 up to the square root of the number and check if there is any divisor other than 1 and the number itself. If we find such a divisor, we set `\$isPrime` to `false` and break out of the loop.

Finally, we use the `echo` statement to print the result, indicating whether the number is prime or not.

6.Mention the sorting functions for arrays in PHP. Explain the functions used for adding and removing array elements in PHP.

In PHP, there are several sorting functions available for arrays:

1. `sort()`: Sorts an array in ascending order based on the values.

2. ``rsort()``: Sorts an array in descending order based on the values.
3. ``asort()``: Sorts an array in ascending order based on the values, while maintaining the association between keys and values.
4. ``arsort()``: Sorts an array in descending order based on the values, while maintaining the association between keys and values.
5. ``ksort()``: Sorts an array in ascending order based on the keys.
6. ``krsort()``: Sorts an array in descending order based on the keys.
7. ``usort()``: Sorts an array using a user-defined comparison function.
8. ``uasort()``: Sorts an array using a user-defined comparison function, while maintaining the association between keys and values.
9. ``uksort()``: Sorts an array using a user-defined comparison function based on the keys.

Here's an example of using the ``sort()`` function to sort an array:

```
...  
$numbers = array(5, 2, 8, 1, 3);  
sort($numbers);  
  
print_r($numbers);  
...
```

Output:

```
...  
Array  
(  
    [0] => 1  
    [1] => 2  
    [2] => 3  
    [3] => 5  
    [4] => 8  
)  
...
```

Regarding adding and removing elements in an array, PHP provides several functions:

1. Adding Elements:

- ``array_push()``: Adds one or more elements to the end of an array.
- ``$array[]``: Appends an element to the end of an array.

2. Removing Elements:

- ``array_pop()``: Removes and returns the last element from an array.
- ``array_shift()``: Removes and returns the first element from an array.
- ``unset()``: Removes a specific element from an array by its key.

Here's an example:

```

$fruits = array("Apple", "Banana", "Orange");

// Adding elements
array_push($fruits, "Mango");
$fruits[] = "Grapes";

// Removing elements
$removedElement = array_pop($fruits);
$removedElement2 = array_shift($fruits);
unset($fruits[1]);

print_r($fruits);

/*
Output:
```
Array
(
 [0] => Banana
 [2] => Mango
)
```
*/

```

In this example, we add elements to the `\$fruits` array using `array_push()` and the array append syntax (`[]`). We also remove elements using `array_pop()` and `array_shift()`, and we remove a specific element using `unset()`. The `print_r()` function is used to display the resulting array.

7. Mention the types of loops used in PHP. Write a PHP code snippet to apply for each loop.

Same as question 3

8. Explain the php function of strpos() and str_replace().	5
--	---

Let's explain the `strpos()` and `str_replace()` functions in PHP.

1. `strpos()`: The `strpos()` function is used to find the position of the first occurrence of a substring within a string. It returns the numeric position of the substring if found, or `false` if the substring is not found.

Syntax: ``strpos($string, $substring)``

Example:

```
$string = "Hello, World!";  
$position = strpos($string, "World");  
echo $position; // Output: 7
```

In this example, the ``strpos()`` function is used to find the position of the substring "World" within the string "Hello, World!". It returns the value ``7``, which is the starting position of the substring.

2. ``str_replace()``: The ``str_replace()`` function is used to replace all occurrences of a substring within a string with another substring. It searches for a specified substring and replaces it with a given replacement string.

Syntax: ``str_replace($search, $replace, $string)``

Example:

```
$string = "Hello, World!";  
$newString = str_replace("World", "Universe", $string);  
echo $newString; // Output: Hello, Universe!
```

In this example, the ``str_replace()`` function replaces the substring "World" with "Universe" within the string "Hello, World!". The resulting string is "Hello, Universe!".

The ``strpos()`` function is used when you want to find the position of a specific substring within a string, while the ``str_replace()`` function is used when you want to replace all occurrences of a substring with another substring within a string.

9.What is PHP? Describe the importance of using PHP

5

Same as question 1

MySQL

1. List six major steps that you would take in setting up a database for a particular enterprise.

Here are the six major steps to set up a database for a particular enterprise, explained in a simplified manner:

1. **Requirements Gathering:** Understand the enterprise's data needs and what they want to achieve with the database.
2. **Database Design:** Create a blueprint of the database structure, including tables and relationships between them.
3. **Selecting a Database Management System (DBMS):** Choose a software that will handle storing and retrieving data efficiently.
4. **Database Implementation:** Create the actual database using the chosen DBMS, including tables and other necessary components.
5. **Data Population:** Add initial data to the database, either by importing from existing sources or manually entering it.
6. **Testing and Deployment:** Thoroughly test the database to ensure it works correctly and meets the enterprise's needs. Once tested, deploy it for regular use.

2. What is data abstraction and examples?	5
---	---

Data abstraction is a concept in programming that allows us to represent complex data in a simplified and manageable way. It involves hiding the implementation details of data structures and exposing only the essential attributes and behaviors. By using data abstraction, we can focus on the high-level functionality of the data rather than the low-level implementation details.

In object-oriented programming, data abstraction is achieved through classes and objects. A class defines the abstract representation of the data, while objects are instances of the class that hold the actual data and provide methods to manipulate it.

1. Encapsulation: Encapsulation is the bundling of data and related methods into a single unit (class), hiding the internal details and providing controlled access to the data through public methods.

2. Abstract Class: An abstract class is a class that cannot be instantiated directly but serves as a blueprint for other classes. It can contain both implemented methods and abstract methods that must be implemented by its subclasses.

3. Interfaces: Interfaces define a contract of methods that a class must implement without providing the method implementations themselves. They enable multiple classes to share a common behavior, promoting code abstraction and loose coupling.

#3.

employee (employee name, street, city)
works (employee name, company name, salary)
company (company name, city)
manages (employee name, manager name)

Figure. Employee database.

- a. Find the names and cities of residence of all employees who work for First Bank Corporation.
- b. Find the names, street addresses, and cities of residence of all employees who work for First Bank Corporation and earn more than \$10,000.
- c. Find all employees in the database who do not work for First Bank Corporation.
- d. Find all employees in the database who earn more than each employee of Small Bank Corporation

a.

```
SELECT employee.employee_name, employee.city
FROM employee
JOIN works ON employee.employee_name = works.employee_name
JOIN company ON works.company_name = company.company_name
WHERE company.company_name = 'First Bank Corporation';
```

B.

```
SELECT employee.employee_name, employee.street, employee.city
FROM employee
JOIN works ON employee.employee_name = works.employee_name
JOIN company ON works.company_name = company.company_name
WHERE company.company_name = 'First Bank Corporation' AND works.salary > 10000;
```

C.

```
SELECT employee.employee_name
FROM employee
LEFT JOIN works ON employee.employee_name = works.employee_name
WHERE works.employee_name IS NULL;
```

D.

```
SELECT employee.employee_name
FROM employee
JOIN works ON employee.employee_name = works.employee_name
JOIN company ON works.company_name = company.company_name
WHERE works.salary > (
    SELECT MAX(works.salary)
    FROM works
    JOIN company ON works.company_name = company.company_name
    WHERE company.company_name = 'Small Bank Corporation'
);
```

#4.

branch(branch name, branch city, assets)
customer (id, customer name, customer street, customer city)
loan (loan number, branch name, amount)
borrower (id, customer name, loan number)
account (account number, branch name, balance)
depositor (id, customer name, account number)

Consider the bank database of Figure. where the primary keys are underlined. Construct the following SQL queries for this relational database.

- Find all customers of the bank who have an account but not a loan.
- Find the names of all customers who live on the same street and in the same city as "Smith".
- Find the names of all branches with customers who have an account in the bank and who live in "Harrison".

A.

```
SELECT customer.customer_name
FROM customer
JOIN depositor ON customer.id = depositor.id
LEFT JOIN borrower ON customer.id = borrower.id
WHERE borrower.id IS NULL;
```

B.

```
SELECT c.customer_name
FROM customer c
JOIN customer s ON c.customer_street = s.customer_street AND c.customer_city = s.customer_city
WHERE s.customer_name = 'Smith';
```

C.

```
SELECT DISTINCT b.branch_name
FROM branch b
JOIN account a ON b.branch_name = a.branch_name
JOIN depositor d ON a.account_number = d.account_number
JOIN customer c ON d.id = c.id
WHERE c.customer_city = 'Harrison';
```

5. List six major steps that you would take in setting up a database for a particular enterprise.	5
---	---

Same as question 1

Prepared By
Sabbir Ahmed