HTML_5_<No-1>_Hands_On_Gunta Divya:

Assessment Goal: Ensure learners understand responsiveness and screen adaptability.

Hands-on Tasks:

1. Add viewport meta tag to the HTML page
2. Use media queries to:
    - Change background color on mobile screen
    - Adjust font size for smaller screens
3. Convert navigation into vertical layout on mobile
4. Test the page using browser responsive mode

Expected Outcome:

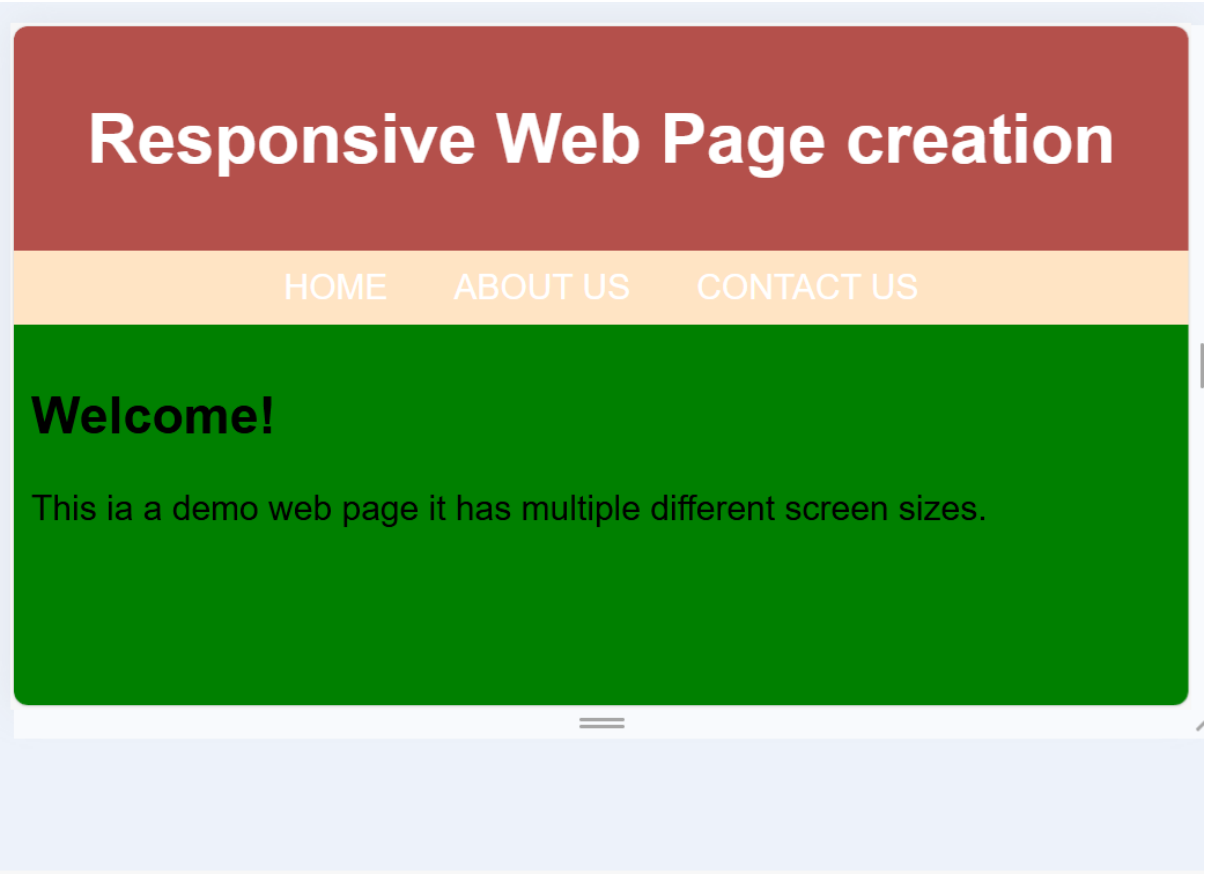
A webpage that looks different and readable on mobile and desktop screens

CODE:-

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Mobile page screen</title>

<style>
  body{
    margin : 0;
    font-family: Arial, Helvetica, sans-serif;
    background-color: green;
    font-size: 20px;
  }
  header{
    background-color: rgb(180, 80, 75);
    color: white;
    padding: 15px;
    text-align: center;
  }
  nav ul{
    display: flex;
    list-style: none;
    margin: 0;
    padding: 0;
    justify-content: center;
    gap: 18px;
    background-color:bisque;
  }
  nav ul li{
    padding: 10px;
  }
```

```
    nav ul li a {
        color: white;
        text-decoration: none;
    }
    main{
       padding:10px;

    }
    @media(max-width: 600px){
          body{
            background-color:blue;
            font-size: 16px;
          }
          nav ul{
            display: block;
            text-align: center;
          }
          nav ul li{
            margin: 10px;
          }
    }

</style>
</head>
<body>
    <header>
        <h1>Responsive Web Page creation</h1>
    </header>

    <nav>
        <ul>
        <li><a href = #>HOME</a></li>
        <li><a href = #>ABOUT US</a></li>
        <li><a href = #>CONTACT US</a></li>
        </ul>
    </nav>
    <main>
        <h2>Welcome!</h2>
        <p>This ia a demo web page it has multiple different screen sizes.</p>
    </main>

</body>
</html>
```

OUTPUT:-



Code Explaination:-
This code creates a responsive web page that adapts to different screen sizes. The viewport meta tag allows the layout to scale correctly on mobile devices. Media queries are used to change the background color and reduce the font size on smaller screens, making the content easier to read. The navigation menu changes from a horizontal layout on desktop to a vertical layout on mobile screens, improving usability. As a result, the webpage looks clear and readable on both desktop and mobile devices.

HTML_5_<No-2>_Hands_On_Gunta Divya:

Assessment Goal:-

A school wants a simple JavaScript program to evaluate a student's performance based on marks obtained in a subject.

Requirements

- Accept the student's marks as a variable
- Use if–else statements to assign grades:
  - Marks ≥ 75 → Grade A
  - Marks ≥ 60 → Grade B
  - Marks ≥ 40 → Grade C
  - Marks < 40 → Fail

Display the grade on the web page or console

Technical Constraints

- Use JavaScript variables (let or const)

- Use numeric data types
- Use comparison and logical operators
- No functions or arrays allowed
- Output using console.log() or document.write()

Learning Outcome

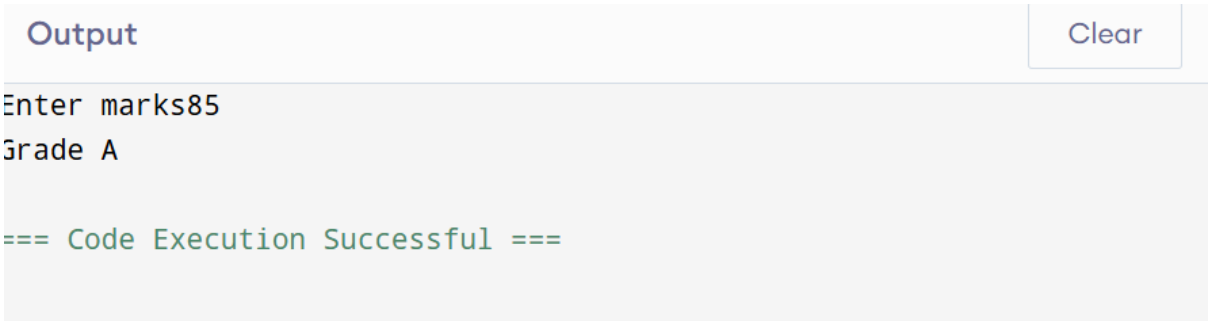
You should be able to:

- Declare and use variables
- Apply comparison operators
- Implement conditional logic using if–else
- Understand decision-making in JavaScript

CODE:-

```
let marks = Number(prompt("Enter marks"));


if(isNaN(marks) || marks < 0 || marks > 100)
{
   console.log("Invalid input");
}
else if (marks >= 75 && marks<100)
{
   console.log("Grade A");
}
else if (marks >= 60 && marks<75)
{
   console.log("Grade B");
}
else if (marks >= 40 && marks<60)
{
   console.log("Grade C");
}
else
{
   console.log("Fail");
}
```

Output:-



Code Explaination:-

This JavaScript code evaluates a student's grade based on the marks entered by the user. The marks variable stores numeric input using Number(prompt()). An if–else ladder is used to check conditions with comparison and logical operators. If the input is invalid (not a number or outside 0–100), an error message is shown. Otherwise, grades A, B, or C are assigned based on the marks, and marks below 40 result in "Fail". The output is displayed using console.log(), demonstrating simple decision-making in JavaScript.

HTML_5_<No-3>_Hands_On_Gunta Divya:

Assessment Goal:-

An online store wants to apply a discount based on the total purchase amount.

Requirements

- Store purchase amount in a variable
- Apply discount rules:
  - Amount ≥ 5000 → 20% discount
  - Amount ≥ 3000 → 10% discount
  - Amount < 3000 → No discount
- Calculate and display:
  - Discount amount
  - Final payable amount

Technical Constraints

- Use arithmetic operators
- Use if–else statements
- Use only primitive data types

No user input (hardcoded values allowed)

Learning Outcome

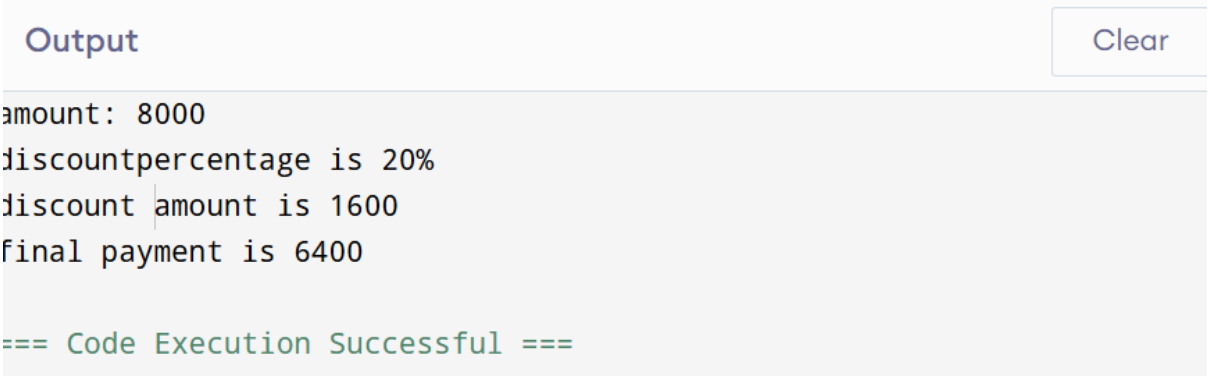
You will be able to:

- Perform calculations using operators
- Work with expressions
- Apply conditional statements
- Build real-world logic using JavaScript basics.

CODE:-

```
let amount=8000;
let discountpercentage;
if(amount>=5000)
{
   discountpercentage=20;
}
else if(amount>=3000)
{
   discountpercentage=10;
}
else
{
   discountpercentage=0;
}
let discount_amount=(amount*discountpercentage)/100;
let final_payabaleamount=amount - discount_amount;
console.log("amount:",amount);
console.log("discountpercentage is",discountpercentage + "%");
console.log("discount amount is",discount_amount);
console.log("final payment is",final_payabaleamount);
```

OUTPUT:-



Code Explaination:-

This JavaScript code calculates a discount for an online store based on a fixed purchase amount. The amount variable stores the total purchase value, and an if–else statement determines the applicable discount percentage according to the given rules. Arithmetic operators are then used to calculate the discount amount and subtract it from the original amount to get the final payable amount. All results, including the original amount, discount

percentage, discount amount, and final payment, are displayed using console.log(). This demonstrates the use of variables, calculations, and conditional logic to solve a real-world problem.

HTML_5_<No-4>_Hands_On_Gunta Divya:

Assessment Goal:-

A traffic control system needs a JavaScript program that displays instructions based on traffic signal color.

Requirements

- Store signal color in a variable ("red", "yellow", "green")
- Use a **switch statement** to display:
- Red → Stop
- Yellow → Get Ready
- Green → Go

Handle invalid signal input gracefully

Technical Constraints

- Must use switch–case
- Use string data types
- Use console.log() for output
- No if–else allowed

Learning Outcome

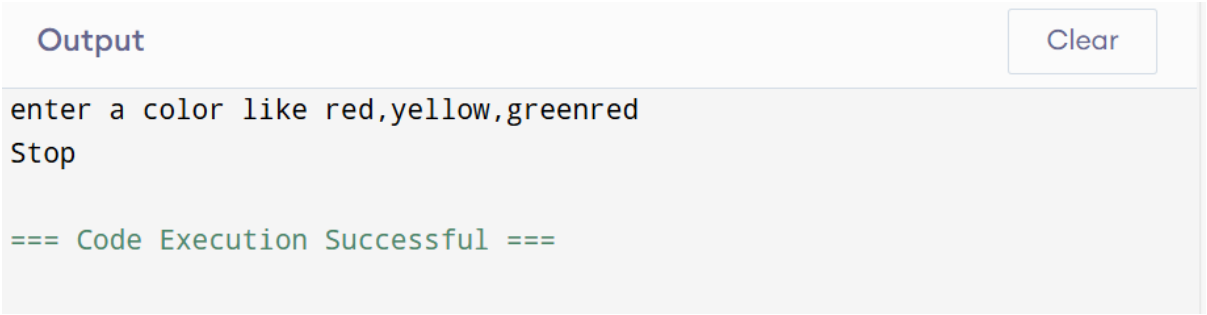
Learners should be able to:

- Use switch statements effectively
- Compare string values
- Handle multiple conditions cleanly
- Understand control flow alternatives

CODE:-

```
let color=prompt("enter a color like red,yellow,green");
switch(color.toUpperCase())
{
 case "RED":
   console.log("Stop");
   break;
 case "YELLOW":
   console.log("Get,Ready");
   break;
 case "GREEN":
   console.log("Go");
   break;
 default:
   console.log("Invalid signal color");
}
```

Output:-



Code Explaination:-

This JavaScript program displays traffic instructions based on the signal color entered by the user. The signal color is stored as a string variable and converted to uppercase to ensure case-insensitive comparison. A switch–case statement is used to match the color with the appropriate instruction: red shows “Stop”, yellow shows “Get Ready”, and green shows “Go”. If the input does not match any valid signal color, the default case handles it gracefully by displaying an error message using console.log().

HTML_5_<No-5>_Hands_On_Gunta Divya:

Assessment Goal:-

A utility program is required to analyze numbers and provide insights such as positivity, parity, and range.

Requirements

- Store a number in a variable
- Use **conditional (ternary) operator** to check:
- Positive or Negative
- Use **if–else** to check:
- Even or Odd
- Use a **loop** to print all numbers from 1 to the given number

Technical Constraints

- Store a number in a variable
- Use conditional (ternary) operator to check:
- Positive or Negative
- Use if–else to check:
- Even or Odd
- Use a loop to print all numbers from 1 to the given number

Learning Outcome

You will be able to:

- Combine multiple control flow techniques
- Use loops for iteration
- Apply conditional operators
- Build multi-step logical programs

CODE:-

```
let number=Number(prompt("enter a number"));
let a=(number>0)?"Positive":"Negative";
```

```
{
   console.log(a);
}
if(number%2==0)
{
   console.log("Even");
}
else
{
   console.log("Odd");
}
let i=1;
while(i<=number)
{
   console.log(i);
   i++
}
```

OUTPUT:-

Code Explaination:-

This JavaScript program analyzes a number entered by the user and provides multiple insights about it. The number is stored in a variable and a conditional (ternary) operator is used to check whether the number is positive or negative. An if–else statement then determines whether the number is even or odd using the modulus operator. Finally, a while loop prints all numbers from 1 up to the given number. This program demonstrates the combined use of conditional operators, decision-making, and loops to build a multi-step logical solution.