"Don't give up, believe in yourself and you can achieve your goals."

HTML:

✅ **Free Courses & Roadmap** 🤯👇🏻

Imagine HTML as the skeleton of a webpage. It provides the basic structure: headings, paragraphs, images, links, etc. Think of it like the frame of a house. You wouldn't live in a house with just a frame, right? That's where CSS comes in.

So, here's the roadmap:

1. Grasp the Basics:

- ❖ Start with understanding elements (like headers, paragraphs, lists), attributes (like formatting), and structure (using tags).
- ❖ Practice creating simple HTML pages to solidify these concepts.

2. Explore Text Formatting:

- ❖ Master text styling with headings, emphasis, and alignment.
- ❖ Learn to organize content using lists and tables.

3. Dive into Links and Images:

- ❖ Integrate hyperlinks to connect pages and navigate content.
- ❖ Incorporate images to enhance visual appeal.

4. Embrace Forms:

- ❖ Create interactive forms to gather user input (like contact forms or surveys).

5. Venture into CSS:

- ❖ Begin styling your pages with colors, fonts, and layouts using CSS.

6. Practice Consistently:

- ❖ Build multiple web pages to strengthen your skills and explore different use cases.
- ❖ Use online resources and tutorials to guide and support your journey.
- ❖ Remember:

HTML is the foundation of web development, so patience and practice are key.
Embrace the learning process and enjoy the satisfaction of creating your web pages!

Here are some free resources to get you started:

🔗 https://youtu.be/6P6yillxZY4?si=WAmHKIAeTZvo_3wA

🔗 https://www.javatpoint.com/html-tutorial

🔗 https://www.w3schools.com/html/default.asp

CSS:

✅ **Free Courses & Roadmap** 🤯👇🏻

CSS is the skin and styling of the webpage. It adds colors, fonts, layouts, and animations, making the webpage visually appealing. Think of it like paint, furniture, and decorations that bring the house to life.

Style with CSS: Master colors, fonts, layouts, and basic animations. Make your webpage look good!

So, here's the roadmap:

➢ **Master the Basics:**

1. Syntax: Selectors, declarations, properties, values.
2. Box Model: Padding, margin, border, background.
3. Typography: Fonts, font size, line height, color.
4. Layout: Display, float, positioning (relative, absolute).

➢ **Intermediate Level:**

1. Pseudo-classes & elements: Hover, focus, active, etc.
2. Media Queries: Responsive design for different screens.
3. Flexbox: Flexible one-dimensional layout.
4. Grid Layout: Two-dimensional grid system for complex layouts.

➢ **Advanced Techniques:**

1. Animations: Transitions, transforms, keyframes.
2. CSS Variables: Store and reuse values for consistency.
3. Preprocessors (Sass, LESS): Extend CSS with features like nesting and mixins.
4. Frameworks & Libraries (Bootstrap, Tailwind): Rapidly build UIs with pre-defined styles.

➢ **Keep Learning:**

1. Follow trends & updates: CSS continues to evolve.
2. Practice & build projects: Solidify your skills through real-world application.
3. Be a part of the community: Learn from others and share your knowledge.

Here are some free resources to get you started:

🔗https://youtu.be/4CcWOiTiDZE?si=I8m_J7-nDX20N2kf

🔗https://www.javatpoint.com/css-tutorial

🔗https://www.freecodecamp.org/news/learn-css-in-this-free-6-hour-video-course/

Javascript:

✅ **Free Courses & Roadmap** 🤯👇🏻

Now, how do you tell the house what to do? That's where JavaScript comes in. It's the brain of the webpage, adding interactivity and dynamic elements. Imagine light switches, thermostats, and speakers that make the house functional.

JavaScript for Interactivity: Add some spark with dynamic elements and user interaction. This is where things get exciting!
Remember, each step builds on the previous one. Think of it as building a beautiful and functional house, brick by brick!

So, here's the roadmap:

➡️Basics: Master syntax, variables, operators, and control flow.
➡️DOM: Learn DOM manipulation, events, and forms.
➡️Functions: Define, invoke, scope, and closures.
➡️Objects: Create, and access properties, methods, and prototypes.
➡️**ES6+: Arrow functions, classes, modules.
➡️Async/Await: Handle asynchronous tasks elegantly.
➡️Fetch API: Make HTTP requests for data.
➡️Popular libraries: Choose & learn 1-2 (React, Vue, Angular).
➡️Practice: Build projects, experiment, and solve problems.
➡️Bonus: Git, testing, best practices.
Remember, consistency > speed. Enjoy the journey!

Here are some free resources to get you started:

🔗 https://www.w3schools.com/js/

🔗 https://www.javatpoint.com/javascript-tutorial

🔗 https://youtu.be/oocF8BRL2N0?si=MhGZpbrBxiYambF9

🔗 https://youtu.be/8dWL3wF_OMw?si=MBYzm8UsAta5qfkz

🔗 https://youtu.be/BTuCzffKh8E?si=v1WHECrQGBScapY1

- **Documented by ASIF MA**
https://www.instagram.com/asifma121/

A small Project:
Creating a Weather App Using JavaScript, HTML, and CSS
🔗 https://youtu.be/KqZGuzrY9D4?si=5aCbLNRVa_eyt7R1

1. Set up the Project:

Create a folder for your project and create three files:
index.html
style.css
Script.js

2. Build the HTML Structure:

In index.html, create the basic structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>Weather App</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="search-container">
        <input type="text" id="city-input" placeholder="Enter a city">
        <button class="search-btn">Search</button>
    </div>
    <div class="current-weather">
        </div>
    <script src="script.js"></script>
</body>
</html>
```

3. Add Some Styles:

In style.css, create basic styling:

```
body {
    font-family: sans-serif;
    text-align: center;
}

.search-container {
    margin-top: 50px;
}

/* Add more styles as needed */
```

4. Fetch Weather Data with JavaScript:

In script.js, use JavaScript to fetch weather data from an API:

```
const API_KEY = "YOUR_API_KEY"; // Replace with your OpenWeatherMap API key
const searchButton = document.querySelector(".search-btn");
const currentWeatherDiv = document.querySelector(".current-weather");

searchButton.addEventListener("click", () => {
    const cityName = document.getElementById("city-input").value;
    const apiUrl =
`https://api.openweathermap.org/data/2.5/weather?q=${cityName}&appid=${API_KEY}`;

    fetch(apiUrl)
        .then(response => response.json())
        .then(data => {
            // Display the weather information
            currentWeatherDiv.innerHTML = `
                <h2>${data.name}, ${data.sys.country}</h2>
                <p>Temperature: ${Math.round(data.main.temp - 273.15)}°C</p>
                <p>Weather: ${data.weather[0].description}</p>
            `;
        })
        .catch(error => {
            console.error("Error fetching weather data:", error);
        });
});
```

5. Customise and Enhance:

Add more features:
Display multiple days of weather forecasts.
Use icons to represent weather conditions.
Allow users to choose between Celsius and Fahrenheit.
Improve styling for a visually appealing app.
Use a weather API that aligns with your desired features.

Remember to replace YOUR_API_KEY with your actual API key from a weather provider like OpenWeatherMap.

I have shown you a path Now decide your Way!!