📌 PHẦN I – HTML

1. **HTML là gì? Khác nhau giữa HTML, XHTML và HTML5?**
   - **HTML (HyperText Markup Language)**: Ngôn ngữ đánh dấu để xây dựng cấu trúc website.
   - **XHTML**: Giống HTML nhưng tuân thủ chặt chẽ cú pháp XML (ví dụ: thẻ phải đóng, thuộc tính viết thường).
   - **HTML5**: Phiên bản mới nhất, hỗ trợ thẻ semantic (`<header>`, `<footer>`, `<section>`), multimedia (`<video>`, `<audio>`), và API mới (Canvas, Local Storage).
2. **Doctype trong HTML dùng để làm gì?**
   - Doctype khai báo phiên bản HTML để trình duyệt render đúng cách.
   - Ví dụ: `<!DOCTYPE html>` cho HTML5.
3. **Thẻ inline và block khác nhau thế nào?**
   - **Block**: Chiếm trọn 1 dòng, có thể chứa thẻ block/inline khác (`<div>`, `<p>`, `<section>`).
   - **Inline**: Chỉ chiếm nội dung vừa đủ, không xuống dòng (`<span>`, `<a>`, `<strong>`).
4. **Semantic HTML là gì?**
   - Sử dụng thẻ HTML có ý nghĩa mô tả nội dung thay vì chỉ trình bày.
   - Ví dụ: `<header>`, `<main>`, `<nav>`, `<footer>`, `<article>`.
   - **Lợi ích**: Tốt cho SEO, code dễ hiểu, hỗ trợ accessibility.
5. **Thuộc tính alt của thẻ `<img>` dùng để làm gì?**
   - Hiển thị text thay thế khi ảnh không load được.
   - Tốt cho SEO (Google Image search).
   - Hỗ trợ người khiếm thị (screen reader).
6. **Sự khác nhau giữa `<div>` và `<span>`?**
   - `<div>`: Thẻ block, dùng để nhóm phần tử lớn.
   - `<span>`: Thẻ inline, dùng để nhóm văn bản nhỏ.
7. **Thẻ `<meta>` là gì?**
   - Cung cấp metadata cho trang web (không hiển thị trực tiếp).
   - Ví dụ:
     - `<meta charset="UTF-8">` (bảng mã ký tự).
     - `<meta name="viewport" content="width=device-width, initial-scale=1.0">` (responsive).
     - `<meta name="description" content="...">` (SEO).

8. **Thẻ `<script>`, `<link>`, `<style>` nên đặt ở đâu?**
    - `<link>` và `<style>`: Trong `<head>`.
    - `<script>`:
        - Trước `</body>` để không chặn render.
        - Hoặc trong `<head>` với `defer`/`async`.
9. **Khác nhau giữa id và class?**
    - `id`: Duy nhất trong trang, dùng để định danh phần tử.
    - `class`: Có thể dùng nhiều lần, để nhóm nhiều phần tử.
10. **Khác biệt giữa `<section>`, `<article>`, `<aside>`, `<nav>`?**
    - `<section>`: Nhóm nội dung theo chủ đề.
    - `<article>`: Nội dung độc lập (tin tức, bài viết).
    - `<aside>`: Nội dung phụ (quảng cáo, sidebar).
    - `<nav>`: Menu điều hướng.
11. **Local Storage, Session Storage và Cookies khác nhau thế nào?**
    - **LocalStorage**: Lưu trữ lâu dài, không hết hạn.
    - **SessionStorage**: Lưu trong session, mất khi đóng tab/trình duyệt.
    - **Cookies**: Dung lượng nhỏ (~4KB), gửi kèm request đến server, thường dùng cho xác thực.
12. **HTML5 có những API nào phổ biến?**
    - Canvas API (vẽ đồ họa).
    - Geolocation API (lấy vị trí người dùng).
    - Web Storage API (LocalStorage, SessionStorage).
    - WebSocket API (giao tiếp real-time).
    - Drag & Drop API.
    - History API.
13. **Khác nhau giữa `<em>`, `<i>` và `<strong>`, `<b>`?**
    - `<em>`: Nhấn mạnh ngữ nghĩa (thường in nghiêng).
    - `<i>`: Chỉ là trình bày (in nghiêng).
    - `<strong>`: Nhấn mạnh ngữ nghĩa quan trọng (thường in đậm).
    - `<b>`: Chỉ là trình bày (in đậm).
14. **Có những loại input trong `<input>` nào?**
    - `text`, `password`, `email`, `number`, `url`, `search`, `tel`.
    - `checkbox`, `radio`.
    - `date`, `time`, `datetime-local`, `month`, `week`.
    - `file`, `color`, `range`.
    - `submit`, `reset`, `button`.

15. **Thẻ `<form>` có những thuộc tính quan trọng nào?**
    - `action`: URL xử lý dữ liệu.
    - `method`: `GET` (query string) hoặc `POST` (body).
    - `enctype`: Kiểu mã hóa (`application/x-www-form-urlencoded`, `multipart/form-data`, `text/plain`).
16. **Lazy loading trong `<img loading="lazy">`?**
    - Chỉ load ảnh khi gần hiển thị trong viewport → tối ưu tốc độ.
17. **Progressive Enhancement và Graceful Degradation?**
    - **Progressive Enhancement**: Bắt đầu từ tính năng cơ bản → thêm nâng cao.
    - **Graceful Degradation**: Xây dựng bản full → giảm bớt để tương thích trình duyệt cũ.
18. **Khác nhau giữa `<iframe>` và `<embed>`?**
    - `<iframe>`: Nhúng tài liệu HTML khác.
    - `<embed>`: Nhúng nội dung ngoài (video, PDF, Flash).
19. **CORS là gì?**
    - **Cross-Origin Resource Sharing**: Cơ chế cho phép/chặn tài nguyên từ domain khác.
20. **Khi nào nên dùng `<button>` thay vì `<a>`?**
    - `<a>`: Dùng để điều hướng (link).
    - `<button>`: Dùng để thực hiện hành động (submit form, gọi JS).

📌 PHẦN II – CSS

1. **CSS là gì? Inline, Internal và External CSS khác nhau thế nào?**
    - **CSS (Cascading Style Sheets)**: Ngôn ngữ định dạng giao diện website.
    - **Inline**: Viết trực tiếp trong thẻ (`style="..."`).
    - **Internal**: Viết trong `<style>` ở `<head>`.
    - **External**: File `.css` liên kết bằng `<link>`. (Tốt nhất cho maintain).
2. **Box Model trong CSS gồm gì?**
    - Content → Padding → Border → Margin.
3. **Position trong CSS có mấy loại?**
    - `static` (mặc định).
    - `relative` (tương đối với vị trí ban đầu).
    - `absolute` (tương đối với phần tử cha có position).
    - `fixed` (cố định theo viewport).

- `sticky` (kết hợp relative + fixed).

4. **Khác nhau giữa relative, absolute, fixed và sticky?**
   - `relative`: Dịch chuyển so với chính nó.
   - `absolute`: Dịch chuyển so với phần tử cha gần nhất có position.
   - `fixed`: Luôn cố định theo màn hình.
   - `sticky`: Dính khi scroll tới ngưỡng.
5. **Flexbox là gì?**
   - Hệ thống layout 1 chiều (row hoặc column).
   - **Thuộc tính quan trọng**:
     - **Container**: `display: flex`, `justify-content`, `align-items`, `flex-wrap`.
     - **Item**: `flex`, `align-self`, `order`.
6. **CSS Grid là gì? So sánh Flexbox và Grid?**
   - **Grid**: Layout 2 chiều (hàng và cột).
   - **Flexbox**: Layout 1 chiều.
   - **So sánh**: Flexbox linh hoạt cho UI nhỏ; Grid mạnh khi làm bố cục tổng thể.
7. **Z-index hoạt động thế nào?**
   - Quyết định thứ tự chồng lớp.
   - Chỉ hoạt động trên phần tử có `position` (khác `static`).
8. **Inline, Block, Inline-block khác nhau?**
   - `inline`: Không xuống dòng, chỉ chiếm nội dung.
   - `block`: Chiếm full chiều ngang, xuống dòng.
   - `inline-block`: Vừa inline vừa cho phép set `width`/`height`.
9. **Pseudo-class và Pseudo-element?**
   - **Pseudo-class**: Trạng thái (`:hover`, `:focus`, `:nth-child()`).
   - **Pseudo-element**: Tạo nội dung ảo (`::before`, `::after`).
10. **Media Queries là gì?**
    - Cú pháp CSS cho responsive, áp dụng theo kích thước màn hình.
    - Ví dụ:

```
@media (max-width: 768px) {
  body {
    font-size: 14px;
  }
}
```

11. **Responsive design là gì?**
    - Website hiển thị tốt trên mọi thiết bị → dùng media queries, flex, grid, `%` thay `px`.
12. **Mobile-first vs Desktop-first?**
    - **Mobile-first**: Viết CSS cho mobile trước → mở rộng bằng `min-width`.
    - **Desktop-first**: Viết CSS cho desktop trước → thu nhỏ bằng `max-width`.
13. **Rem, Em, Px, %, Vw, Vh khác nhau?**
    - `px`: Giá trị cố định.
    - `em`: Phụ thuộc `font-size` phần tử cha.
    - `rem`: Phụ thuộc `font-size` root (html).
    - `%`: Theo phần tử cha.
    - `vw`, `vh`: Theo viewport (chiều rộng, chiều cao).
14. **CSS Specificity là gì?**
    - Thứ tự ưu tiên: `!important` > inline style > `id` > `class`/`pseudo-class` > `element`.
15. **Transition vs Animation?**
    - **Transition**: Chuyển động khi có thay đổi trạng thái.
    - **Animation**: Có keyframe, chạy liên tục.
16. **Keyframes trong CSS Animation?**
    - Xác định trạng thái tại từng mốc % thời gian.
    - Ví dụ:

```
@keyframes move {
  0% {
    left: 0;
  }
  100% {
    left: 100px;
  }
}
```

17. **CSS Variables?**
    - Khai báo biến:

```
:root {
  --main-color: red;
}
div {
```

```
    color: var(--main-color);
}
```

- **Ưu điểm**: Dễ tái sử dụng, thay đổi linh hoạt.

18. **Critical CSS là gì?**

- CSS cần thiết để render giao diện trên màn hình đầu tiên.
- Tách critical CSS để tăng tốc độ load.

19. **CSS Preprocessor (SASS, LESS)?**

- Ngôn ngữ mở rộng cho CSS (có biến, vòng lặp, mixin).
- **Ưu**: Dễ maintain, tái sử dụng.
- **Nhược**: Cần compile sang CSS.

20. **CSS BEM là gì?**

- **Block, Element, Modifier** – cách đặt tên class rõ ràng:
  - **Block**: `.card`
  - **Element**: `.card__title`
  - **Modifier**: `.card--active`