**NAMA : OKI JAYA GUNA**

**NIM : 22650161**

**Menjelaskan sintax java script**

1. **let list\_tugas = [""];**

* Membuat sebuah array bernama list\_tugas dengan elemen awal berupa string kosong (""). Array ini digunakan untuk menyimpan daftar tugas.

1. **const output\_element = document.querySelector("#output");**

* Menggunakan document.querySelector() untuk memilih elemen HTML dengan id="output" dan menyimpannya ke dalam variabel output\_element. Elemen ini akan digunakan untuk menampilkan daftar tugas.

1. **function renderTasks() { output\_element.innerHTML = "";**

* Mendefinisikan fungsi renderTasks() yang bertujuan untuk memperbarui elemen HTML dengan daftar tugas dari list\_tugas.
* output\_element.innerHTML = ""; membersihkan konten elemen output\_element agar tidak terjadi duplikasi saat daftar diperbarui.

1. **list\_tugas.forEach((value, index) => {**

* Melakukan iterasi pada array list\_tugas menggunakan forEach. Setiap elemen dari array diakses melalui parameter value (nilai tugas) dan index (indeks array).

1. **const p\_elm = document.createElement("p");**

* Membuat elemen paragraf (<p>) secara dinamis menggunakan document.createElement() dan menyimpannya di variabel p\_elm.

1. **p\_elm.textContent = value;**

* Menetapkan teks dari elemen paragraf (p\_elm) dengan nilai dari elemen array (value).

1. **p\_elm.addEventListener("click", () => { list\_tugas.splice(index, 1); renderTasks(); });**

* Menambahkan event listener ke elemen p\_elm. Ketika elemen diklik, akan menghapus elemen dari array list\_tugas pada indeks tertentu menggunakan splice.
* Memanggil kembali fungsi renderTasks() untuk memperbarui daftar tugas yang ditampilkan.

1. **output\_element.appendChild(p\_elm);**

* Menambahkan elemen paragraf (p\_elm) yang telah diatur ke dalam elemen HTML output\_element.

1. **renderTasks();**

* Memanggil fungsi renderTasks() untuk pertama kali agar daftar tugas (yang saat ini kosong) ditampilkan.

1. **const formulir = document.querySelector("form");**

* Memilih elemen HTML <form> dan menyimpannya ke dalam variabel formulir.

1. **formulir.addEventListener("submit", (e) => { e.preventDefault();**

* Menambahkan event listener ke form untuk menangani event submit.
* e.preventDefault() mencegah perilaku default form (seperti reload halaman) agar data dapat diproses langsung oleh JavaScript.

1. **const formData = new FormData(e.target); const obj = Object.fromEntries(formData);**

* new FormData(e.target) mengambil data dari form sebagai objek FormData.
* Object.fromEntries(formData) mengubah FormData menjadi objek JavaScript biasa, sehingga dapat diakses dengan properti key-value.

1. **if (obj.tugas) { list\_tugas.push(obj.tugas); renderTasks(); }**

* Memeriksa apakah properti tugas dari objek form memiliki nilai.
* Jika ada nilai, tugas tersebut ditambahkan ke array list\_tugas menggunakan push.
* Memanggil kembali renderTasks() untuk menampilkan daftar yang diperbarui.

1. **e.target.reset();**

* Mengosongkan form setelah data berhasil diproses menggunakan e.target.reset().

# Menjelaskan sintax html

1. <!DOCTYPE html>

<html lang="en">

* + - <!DOCTYPE html>: Mendeklarasikan dokumen sebagai HTML5.
    - <html lang="en">: Membuka elemen root <html> dengan atribut lang="en", yang menandakan bahwa dokumen menggunakan bahasa Inggris.

1. <head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initialscale=1.0" />

<title>Document</title>

</head>

* + - <head>: Bagian ini berisi informasi metadata tentang dokumen HTML.
    - <meta charset="UTF-8" />: Menentukan encoding karakter dokumen sebagai UTF-8, yang mendukung berbagai karakter termasuk huruf dan simbol non-Inggris.
    - <meta name="viewport" content="width=device-width, initialscale=1.0" />
    - <title>Document</title>: Menetapkan judul dokumen yang akan muncul di tab browser.

1. <body>

<form action="">

* <body>: Membuka bagian utama dokumen HTML yang akan ditampilkan di browser
* <form action="">: Membuka elemen form HTML. Atribut action dibiarkan kosong karena pengelolaan data dilakukan menggunakan JavaScript, bukan mengirimkan data ke server.

1. <input type="text" name="tugas" placeholder="masukan tugas" />

<button type="submit">tambah</button>

* 1. <input type="text" name="tugas" placeholder="masukan tugas" />: Membuat input teks untuk pengguna memasukkan tugas.
     + - type="text": Menentukan input sebagai teks.
       - name="tugas": Memberikan nama pada input, yang digunakan untuk mengambil data di JavaScript.
       - placeholder="masukan tugas": Menampilkan teks petunjuk sementara dalam kotak input.
  2. <button type="submit">tambah</button>:
     + - Membuat tombol submit untuk mengirimkan data form.
       - type="submit": Menentukan tombol sebagai submit, sehingga form akan memicu event submit.

1. <div id="output"></div>

* Elemen div dengan id="output" sebagai wadah untuk menampilkan daftar tugas yang akan dibuat dinamis oleh JavaScript.

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

* Menyisipkan file JavaScript eksternal aplikasi.js, yang berisi logika utama untuk mengelola tugas.

1. </body>

</html>

* + - * </body>: Menutup elemen body.
      * </html>: Menutup elemen root HTML.