ESERCIZI JAVASCRIPT AVANZATO

Esercizio 1

Scrivi una funzione che calcoli la somma di due numeri. Definisci le variabili con `var`, poi riscrivi la stessa funzione utilizzando `let` e `const`. Osserva le differenze di comportamento.

Domanda:

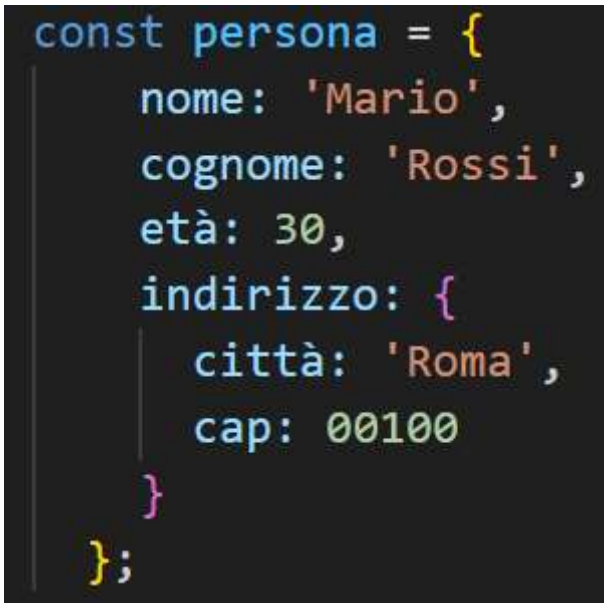
Spiega perché `const` non può essere riassegnato, ma un oggetto definito con `const` può essere modificato.

Esercizio 2

Scrivi un ciclo `for` che conta da 1 a 5 utilizzando `let`. Prova a fare lo stesso usando `var`. Che differenza noti?

Esercizio 3

Dai il seguente oggetto:



```
const persona = {
    nome: 'Mario',
    cognome: 'Rossi',
    età: 30,
    indirizzo: {
      città: 'Roma',
      cap: 00100
    }
  };
```

- Destruttura il nome e la città.
- Cambia il valore della città senza modificare l'oggetto originale

Esercizio 4

Dato il seguente array:

```
const numeri = [1, 2, 3, 4, 5];
```

- Usa la destrutturazione per estrarre il primo e il secondo elemento in due variabili.
- Crea una funzione che accetti un array come parametro e destrutturi i primi tre elementi.

Esercizio 5

Copia il seguente array utilizzando lo spread operator e aggiungi un nuovo numero alla fine:

```
const numeriOriginali = [10, 20, 30];
```

Bonus:

Fai lo stesso con un oggetto e aggiungi una nuova proprietà.

Esercizio 6

Scrivi una funzione che accetti un numero e ritorni:

- `"pari"` se il numero è pari,
- `"dispari"` se il numero è dispari.

Fai questo usando un **ternary operator**.

Esercizio 7

Riscrivi questo `if` usando il ternary operator:

```
if (utente && utente.isAdmin) {
    console.log('Accesso consentito');
  } else {
    console.log('Accesso negato');
  }
```

Esercizio 8

Dato l’oggetto:

Usa **optional chaining** per accedere alla città. Modifica l'oggetto per vedere come l'optional chaining previene errori.

Esercizio 9

Scrivi un codice che intercetti il click su un link e **impedisca il comportamento predefinito**.

Esercizio 10

Scrivi una funzione che intercetti il click su un bottone all’interno di un div e mostri un messaggio.

Ora fai in modo che il click sul div **non propaghi l'evento** al bottone.

Esercizio 11

Crea due file JavaScript:

- `modulo.js` con una funzione `saluta` che esporta in modo predefinito (export default).
- `index.js` che importa e utilizza la funzione.

Esercizio 12

Scrivi una funzione **eseguiDopo** che accetti una **callback** come parametro e la esegua dopo 2 secondi.

Esercizio 13

Dato il seguente codice:

```
console.log('Inizio');
setTimeout(() => console.log('Timeout'), 2000);
console.log('Fine');
```

- Qual è l'ordine di esecuzione delle istruzioni?
- Spiega come il **browser** gestisce il **setTimeout**.

Esercizio 14

Osserva il seguente codice e prevedi l'output (considera quali istruzioni sono sincrone e vengono eseguite prima e quali invece asincrone e vengono delegate al browser):

```
console.log('Uno');
setTimeout(() => console.log('Due'), 0);
console.log('Tre');
```