### [**https://shorturl.at/70zDo**](https://shorturl.at/70zDo)

### **Exercise 1: Product Click Handler**

**Task:** Implement a product click handler.

**Function Name:** createProductClickHandler

**Guidelines:**

* **Parameters:**
  + productId (number): Unique identifier for the product.
  + productName (string): Name of the product.
* **Return Value:** Returns a function that, when called, logs the product details.

**Example Usage:**

const product1Handler = createProductClickHandler(1, 'Shirt');

const product2Handler = createProductClickHandler(2, 'Shoes');

// Simulating product clicks

product1Handler(); // Product clicked: ID=1, Name=Shirt

product2Handler(); // Product clicked: ID=2, Name=Shoes

### **Exercise 2: Discount Generator**

**Task:** Build a discount generator.

**Function Name:** createDiscountGenerator

**Guidelines:**

* **Parameters:**
  + discount (number): Discount percentage as a decimal (e.g., 0.15 for 15%).
* **Return Value:** Returns a function that accepts a price and applies the discount.

**Example Usage:**

const tenPercentOff = createDiscountGenerator(0.10);

const twentyPercentOff = createDiscountGenerator(0.20);

tenPercentOff(100); // Original Price: $100, Discounted Price: $90

twentyPercentOff(100); // Original Price: $100, Discounted Price: $80

### **Exercise 3: Inventory Management**

**Task: Create an inventory management system.**

**Function Name: createInventory**

**Guidelines:**

* **Private Variable: Use an object to store product quantities.**
  + **Hint:** 
    - **let inventory = {};**
    - **inventory['Laptop'] = 10**
* **Methods to Implement:**
  + **addProduct(name, quantity): Adds a product with a specified quantity.**
    - **Parameters:**
      * **name (string): Name of the product.**
      * **quantity (number): Quantity to add.**
    - **Output: Logs the addition of the product.**
  + **sellProduct(name, quantity): Sells a specified quantity of a product.**
    - **Parameters:**
      * **name (string): Name of the product to sell.**
      * **quantity (number): Quantity to sell.**
    - **Output: Logs whether the sale was successful or if there was not enough stock.**
  + **getInventory(): Returns the current inventory.**
    - **Output: Logs and returns the inventory object.**

**Example Usage:**

**const storeInventory = createInventory();**

**storeInventory.addProduct('Laptop', 10); // Added 10 of Laptop.**

**storeInventory.addProduct('Phone', 5); // Added 5 of Phone.**

**storeInventory.getInventory(); // Current Inventory: { Laptop: 10, Phone: 5 }**

**storeInventory.sellProduct('Laptop', 2); // Sold 2 of Laptop.**

**storeInventory.getInventory(); // Current Inventory: { Laptop: 8, Phone: 5 }**

**storeInventory.sellProduct('Phone', 10); // Not enough stock for Phone.**

### **Exercise 4: Shopping Cart**

**Task:** Create a shopping cart using closures.

**Function Name:** createCart

**Guidelines:**

* **Private Variable:** Use a private array to store cart items.
* **Methods to Implement:**
  + addItem(name, price): Adds an item to the cart.  
    - **Parameters:**
      * name (string): The name of the item.
      * price (number): The price of the item.
    - **Output:** Logs the addition of the item.
  + removeItem(name): Removes an item by name.  
    - **Parameters:**
      * name (string): The name of the item to remove.
    - **Output:** Logs whether the item was removed or not found.
  + getTotal(): Calculates the total price of items in the cart.  
    - **Output:** Logs the total price and returns it.
  + getItems(): Returns the current items in the cart.  
    - **Output:** Returns the cart array.

**Example Usage:**

* const myCart = createCart();
* myCart.addItem('Shirt', 20); // Added: Shirt, Price: $20
* myCart.addItem('Shoes', 50); // Added: Shoes, Price: $50
* myCart.getTotal(); // Total Price: $70
* myCart.removeItem('Shirt'); // Removed: Shirt
* console.log(myCart.getItems()); // [{ name: 'Shoes', price: 50 }]