**JavaScript Objects Assignment**

**Basic Concepts:**

1. **Object Creation:**
   * Create an object person with properties firstName, lastName, age, and city. Print all properties to the console.
   * Add a new property occupation to the person object and then print the updated object.
2. **Object Methods:**
   * Add a method fullName() to the person object that returns the full name (concatenation of firstName and lastName).
   * Add another method ageIn10Years() that calculates and returns the person's age in 10 years.
3. **Modifying Object Properties:**
   * Modify the age of the person object and print the modified object.
   * Delete the city property from the person object and verify that it’s deleted by logging the object.
4. **Accessing Object Properties:**
   * Write a function that takes an object and a key as arguments and returns the value of the key from the object.
   * Use both dot notation and bracket notation to access the properties of the person object.

**Intermediate Concepts:**

1. **Object Iteration:**
   * Write a function that loops over all the properties of an object and logs each key and value.
   * Create an array of objects students where each object contains properties name, age, and grade. Write a function that iterates over the array and prints the name and grade of each student.
2. **Nested Objects:**
   * Create an object book with properties title, author, and details. details should be a nested object containing publisher and year. Write a function to display all the properties of the book object, including nested properties.
3. **Object Destructuring:**
   * Destructure the firstName, lastName, and age properties from the person object.
   * Given the object:

js

Copy code

const car = { brand: 'Tesla', model: 'Model 3', price: 35000 };

Use destructuring to extract the brand and price properties, then log them to the console.

**Advanced Concepts:**

1. **Constructor Functions:**
   * Create a constructor function Car that takes brand, model, and price as arguments and initializes them as properties of the object. Create three different car objects using the constructor function.
   * Add a method getDetails() to the Car constructor function that returns a string with the car’s details.
2. **Object.assign() and Spread Operator:**
   * Create two objects: person1 and person2, with some common and some unique properties.
   * Use Object.assign() to merge the properties of both objects into a new object and log the result.
   * Perform the same task using the spread operator { ...person1, ...person2 }.
3. **Prototypes:**
   * Add a method greet() to the person object using the prototype and make it accessible to all objects created using the Person constructor function.
   * Create another constructor function Employee that inherits from Person, and add a method getJobTitle() to Employee that returns the employee's job title.

**Challenging Concepts:**

1. **Object.freeze() and Object.seal():**
   * Create an object settings with properties theme, volume, and brightness. Use Object.freeze() to freeze the object and then try modifying one of its properties.
   * Use Object.seal() on the settings object and attempt to add and delete properties. Observe and explain the differences between Object.freeze() and Object.seal().
2. **Inheritance and Prototypes:**
   * Create a constructor function Animal with properties name and species. Add a method speak() to the prototype of Animal that logs a message like, "The [species] goes 'woof'".
   * Create a constructor function Dog that inherits from Animal. Add a method bark() to Dog that logs "The dog barks!".
   * Instantiate an object of Dog and call both bark() and speak() methods.
3. **Shallow Copy vs. Deep Copy:**
   * Explain the difference between shallow copy and deep copy of objects with an example.
   * Write code to demonstrate how modifying the original object affects a shallow copy but not a deep copy. Use Object.assign() for shallow copy and JSON methods (JSON.parse(JSON.stringify())) for deep copy.

**Bonus Challenges:**

1. **Dynamic Properties:**
   * Write a function addProperty(obj, key, value) that dynamically adds a new property to an object.
2. **Object Keys, Values, and Entries:**
   * Given an object inventory with keys as item names and values as quantities:

js

Copy code

const inventory = { apple: 5, banana: 10, orange: 8 };

Write functions to:

* + - List all the keys.
    - List all the values.
    - Convert the object into an array of key-value pairs using Object.entries() and then iterate over it.