**1. Basic Types and Syntax**

* **Q1:** Create a TypeScript program that defines variables of the following types: number, string, boolean, null, undefined, and any. Print the values to the console.
* **Q2:** Define a variable age of type number and a variable isActive of type boolean. Write a function getUserInfo that accepts these two variables and returns a formatted string with the age and status.

### **2. Interfaces and Classes**

* **Q3:** Define an interface Person that includes the properties name (string) and age (number). Create a class Employee that implements this interface. Add an additional property employeeId and a method getEmployeeDetails() that returns the employee's details as a string.
* **Q4:** Create a class Car with properties like make, model, and year. Implement an interface Vehicle with a method startEngine(). Ensure that the Car class implements this interface.

### **3. Advanced Types (Generics, Enums, Union Types)**

* **Q5:** Create a generic function identity<T> that accepts a parameter of type T and returns it. Call this function with different types (e.g., string, number).
* **Q6:** Define an enum Color with values Red, Green, and Blue. Create a function getColorName that takes an enum value and returns its string representation.
* **Q7:** Create a function printDetails that takes a parameter of a union type string | number. Print the type and value of the parameter.

### **4. Functions and Arrow Functions**

* **Q8:** Write a normal function multiplyNumbers that accepts two numbers and returns their product. Convert this function into an arrow function.
* **Q9:** Create an arrow function sum that takes an array of numbers and returns the sum of all numbers.

### **5. Modules and Namespaces**

* **Q10:** Create two files: mathOperations.ts and main.ts. In mathOperations.ts, export a function add(a: number, b: number): number. In main.ts, import and use this function to calculate the sum of two numbers.
* **Q11:** Define a namespace Geometry with a function calculateArea(radius: number): number. Use this namespace in another file to calculate the area of a circle.

### **6. Type Declaration Files (.d.ts)**

* **Q12:** Create a Type Declaration File example.d.ts that declares the SomeLibrary module. Define the types for a function initializeApp(config: AppConfig): void and an interface AppConfig that includes properties like apiUrl (string) and retryCount (number).
* **Q13:** Write a TypeScript function that uses an external JavaScript library (e.g., lodash or jQuery). Create a .d.ts declaration file for that library to define basic type information for the library's methods.

### **7. Decorators**

* **Q14:** Implement a simple class decorator that logs a message whenever a class is instantiated. Apply the decorator to a class User.
* **Q15:** Create a property decorator that checks if a property is a valid email. If it’s not a valid email, throw an error when the class is instantiated.

### **8. Error Handling**

* **Q16:** Create a function parseJson that accepts a string and attempts to parse it as JSON. If parsing fails, return null and log an error message.
* **Q17:** Write a function divideNumbers that accepts two numbers. If the denominator is 0, throw an error with a message "Cannot divide by zero". Otherwise, return the result.