**Source Exercises**

**1. Build a Person Class**

* Create a class called Person with the following properties:
  + firstName (string)
  + lastName (string)
  + age (number)
* Add methods to:
  + getFullName() – returns the full name in the format: <firstName> <lastName>.
  + celebrateBirthday() – increases the age by 1.
* Create two instances of the Person class and call their methods. Print the results to the console.

**2. Create an Employee Class Using Inheritance**

* Create a subclass Employee that extends the Person class. It should add:
  + jobTitle (string)
  + salary (number)
* Add a new method getEmployeeInfo() – returns a string in the format: "Name: <fullName>, Job Title: <jobTitle>, Salary: <salary>".
* Create an instance of Employee and demonstrate how to use methods inherited from Person along with the new methods in Employee.

**3. Static Methods in a StringUtilities Class**

* Create a StringUtilities class that contains static methods to manipulate strings:
  + toUpperCase(str) – converts a string to uppercase.
  + toLowerCase(str) – converts a string to lowercase.
  + capitalizeFirstLetter(str) – capitalizes the first letter of the string.
  + reverseString(str) – reverses the order of characters in the string.
* Show how to call each method without creating an instance of the class, passing in a string to test each static method.

**4. Encapsulation with Private Fields**

* Modify the Person class so that the age property is private (use the # syntax for private fields).
* Provide a getter for age and a setter that ensures the age cannot be set to a value less than 0.
* Write code to test the encapsulation by attempting to directly access and modify the age and observing the behavior through the getter and setter.

**5. Polymorphism with a ContractEmployee Class**

* Create another subclass of Employee called ContractEmployee with the additional property:
  + contractDuration (number, representing months)
* Override the getEmployeeInfo() method in ContractEmployee to include the contract duration in the output.
* Create instances of both Employee and ContractEmployee and demonstrate polymorphism by calling their getEmployeeInfo() methods.