**List the features of ES6**  
ES6, also known as ECMAScript 2015, introduced several new features to JavaScript that improve the readability, maintainability, and performance of code. Key features include block-scoped variable declarations (let and const), arrow functions, template literals, default parameters, rest and spread operators, destructuring assignments, classes, modules, promises, and new data structures such as Map and Set.

**Explain JavaScript let**  
The let keyword is used to declare variables in JavaScript with block-level scope. Unlike var, which is function-scoped, let confines the variable to the block in which it is defined, such as within a loop or conditional statement. It prevents variable hoisting issues and re-declaration within the same scope, making it safer and more predictable for managing variables.

**Identify the differences between var and let**  
The main differences between var and let are in scope, hoisting, and re-declaration. var is function-scoped, while let is block-scoped. Variables declared with var are hoisted to the top of their scope and initialized with undefined, whereas let variables are hoisted but not initialized, resulting in a "temporal dead zone" until the declaration is evaluated. Additionally, var allows re-declaration within the same scope, while let does not.

**Explain JavaScript const**  
The const keyword is used to declare variables whose values cannot be reassigned after initialization. Like let, const is block-scoped and does not allow re-declaration in the same scope. Although the reference cannot be changed, the contents of objects or arrays declared with const can still be modified. It is commonly used for defining constants and ensuring immutability of variable bindings.

**Explain ES6 class fundamentals**  
ES6 introduced a new syntax for defining classes in JavaScript. A class is a blueprint for creating objects with shared methods and properties. The basic syntax includes the class keyword, a constructor method to initialize object properties, and other methods defined inside the class body. Classes support inheritance, encapsulation, and method overriding. This syntax provides a cleaner and more intuitive way to work with object-oriented programming concepts in JavaScript.

**Explain ES6 class inheritance**  
ES6 supports class inheritance using the extends keyword. A child class can inherit properties and methods from a parent class, allowing for code reuse and hierarchical structuring. The super() function is used within the constructor of the child class to call the constructor of the parent class and initialize inherited properties. Inheritance enables the creation of specialized classes that extend the functionality of base classes.

**Define ES6 arrow functions**  
Arrow functions are a concise way to write function expressions in ES6. They use the => syntax and do not have their own this, arguments, super, or new.target, making them especially useful in situations where function context should be inherited from the surrounding scope. They are often used for callbacks, array methods, and concise function declarations.

**Identify set(), map()**  
In ES6, Set and Map are new collection types for storing unique values and key-value pairs, respectively. A Set stores a collection of unique values of any type, automatically eliminating duplicates. A Map stores key-value pairs where keys can be of any type, unlike regular JavaScript objects which only accept strings or symbols as keys. Both provide useful methods such as add(), delete(), has(), and clear() for Set, and set(), get(), has(), and delete() for Map.