1. var, let, const

Definition: These are used to declare variables in JavaScript.

Example Code:

```
var x = 10; // function scoped

let y = 20; // block scoped

const z = 30; // block scoped and can't be reassigned
```

Explanation: Use 'let' and 'const' in modern JS. 'const' for values that don't change.

2. Hoisting

Definition: Hoisting is JavaScript's behavior of moving declarations to the top.

Example Code:

```
console.log(a); // undefined

var a = 5;
```

Explanation: Only declarations are hoisted, not initializations. 'let' and 'const' are hoisted but not initialized.

3. Template Literals

Definition: Template literals allow embedded expressions and multiline strings.

Example Code:

```
let name = "Alice";

let greeting = `Hello, ${name}!`;
```

Explanation: Use backticks ` instead of quotes. Supports variables and expressions directly.

4. Ternary Operators

Definition: A shorthand for if-else conditions.

Example Code:

```
let age = 18;

let access = age >= 18 ? "Allowed" : "Denied";
```

Explanation: The syntax is: condition ? true_value : false_value

5. Short Circuits

Definition: Logical operators return the first truthy/falsy value.

Example Code:

```
let name = "";

let user = name || "Guest";
```

Explanation: OR (||) returns first truthy. AND (&&) returns first falsy.

6. Spread & Rest Operators

Definition: Spread (...) expands arrays/objects. Rest (...) collects multiple elements.

Example Code:

```
let nums = [1, 2, 3];

let newNums = [...nums, 4]; // Spread

function sum(...args) { return args.reduce((a,b)=>a+b); }
```

Explanation: Spread is for copying/spreading. Rest is for gathering parameters.

7. Destructuring of Array

Definition: Extract elements from an array into variables.

Example Code:

```
let [a, b] = [10, 20];
```

Explanation: Easily assign array items to variables.

8. Destructuring of Object

Definition: Extract properties from an object into variables.

Example Code:

```
let person = { name: 'John', age: 30 };

let { name, age } = person;
```

Explanation: Use same property names to extract from objects.

9. Pass By Value and Reference

Definition: Primitive types are passed by value, objects by reference.

Example Code:

```
let a = 5;

let b = a; // value copy

let obj1 = { x: 10 };

let obj2 = obj1; // reference copy
```

Explanation: Changes to obj2 affect obj1 because they reference the same memory.

10. Object Methods

Definition: Built-in methods to manipulate object data.

Example Code:

```
let obj = { a: 1, b: 2 };

Object.keys(obj);    // ['a', 'b']

Object.values(obj);  // [1, 2]

Object.freeze(obj);  // prevents modifications

Object.entries(obj); // [['a', 1], ['b', 2]]
```

Explanation: Useful for inspecting and controlling object properties.