***JavaScript***

**Key Words:**

* break, case, catch, continue, debugger, default, delete, do, else, finally, for, function, if, in, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with.

**Relational Operators:**

1. **Equality (==):**
   * Compares two values for equality.
   * Example: **a == b**
2. **Inequality (!=):**
   * Compares two values for inequality.
   * Example: **a != b**
3. **Strict Equality (===):**
   * Compares two values for strict equality (checks both value and type).
   * Example: **a === b**
4. **Strict Inequality (!==):**
   * Compares two values for strict inequality.
   * Example: **a !== b**
5. **Greater Than (>):**
   * Checks if the value on the left is greater than the value on the right.
   * Example: **a > b**
6. **Less Than (<):**
   * Checks if the value on the left is less than the value on the right.
   * Example: **a < b**
7. **Greater Than or Equal To (>=):**
   * Checks if the value on the left is greater than or equal to the value on the right.
   * Example: **a >= b**
8. **Less Than or Equal To (<=):**
   * Checks if the value on the left is less than or equal to the value on the right.
   * Example: **a <= b**

**Logical Operators:**

1. **Logical AND (&&):**
   * Returns **true** if both operands are **true**.
   * Example: **a && b**
2. **Logical OR (||):**
   * Returns **true** if at least one of the operands is **true**.
   * Example: **a || b**
3. **Logical NOT (!):**
   * Returns **true** if the operand is **false**, and vice versa.
   * Example: **!a**

**JavaScript Array Library Functions:**

JavaScript provides a rich set of built-in methods for working with arrays. These methods can be broadly categorized into different functionalities:

**Manipulation:**

* **Adding/Removing Elements:**
  + push(): Adds one or more elements to the end of the array and returns the new length.
  + pop(): Removes the last element from the array and returns it.
  + shift(): Removes the first element from the array and returns it.
  + unshift(): Adds one or more elements to the beginning of the array and returns the new length.
* **Combining Arrays:**
  + concat(): Creates a new array by merging multiple arrays (does not modify original arrays).
  + splice(): Removes or replaces elements in the array, and optionally inserts new elements at a specified position.

**Iteration:**

* forEach(): Executes a provided function once for each array element.

**Transformation:**

* map(): Creates a new array with the results of calling a function on every element in the original array.
* filter(): Creates a new array with elements that pass a test implemented by the provided function.
* reduce(): Reduces the array to a single value by applying a function against an accumulator and each element (useful for calculations or aggregations).
* sort(): Sorts the array in place (or returns a new sorted array) based on a comparison function.
* reverse(): Reverses the order of the elements in the array in place (or returns a new reversed array).

**Searching:**

* indexOf(): Returns the first index at which a given element can be found in the array, or -1 if it's not found.
* lastIndexOf(): Returns the last index at which a given element can be found in the array, or -1 if it's not found.
* find(): Returns the first element in the array that satisfies a provided testing function.
* findIndex(): Returns the index of the first element in the array that satisfies a provided testing function.
* includes(): Determines whether an array contains a specific element (returns true or false).

**Other Utilities:**

* join(): Joins all elements of an array into a string, separated by a specified separator.
* slice(): Extracts a section of the array and returns a new array (copies elements).
* fill(): Fills all or part of an array with a static value.
* isArray(): Determines whether an object is an array.