Assume you have full experience in problem solving in javascript, and knows what string methods comes most often, then list all the string methods from high to low priority. make sure it must be straight forward with single sentence talks what the method does beside it

Got it 👍 — here's a **straightforward priority-based list of JavaScript string methods**, ranked from **most commonly used in real-world problem-solving** to less frequent ones, each with a **one-line explanation**:

🔝 High Priority (used very often)

1. `length` – returns the number of characters in a string.
2. `charAt(index)` – gets the character at the given index.
3. `charCodeAt(index)` – gets the Unicode value of the character at the given index.
4. `at(index)` – gets the character at the given index (supports negative indexing).
5. `slice(start, end)` – extracts a section of the string without modifying it.
6. `substring(start, end)` – extracts characters between two indexes.
7. `substr(start, length)` – extracts part of a string based on start and length (deprecated but still used).
8. `indexOf(searchValue, fromIndex)` – finds the first occurrence index of a substring.
9. `lastIndexOf(searchValue, fromIndex)` – finds the last occurrence index of a substring.

10. `includes(searchString, position)` – checks if the string contains the given substring.

11. `startsWith(searchString, position)` – checks if the string begins with the given substring.

12. `endsWith(searchString, length)` – checks if the string ends with the given substring.

13. `toLowerCase()` – converts the string to lowercase.

14. `toUpperCase()` – converts the string to uppercase.

15. `trim()` – removes whitespace from both ends of a string.

16. `trimStart()` / `trimEnd()` – removes whitespace only at the start or end.

17. `replace(searchValue, newValue)` – replaces the first match with a new substring.

18. `replaceAll(searchValue, newValue)` – replaces all matches with a new substring.

19. `split(separator, limit)` – splits a string into an array by a separator.

20. `concat(...strings)` – joins two or more strings together.

21. `repeat(count)` – returns a new string repeated a given number of times.

22. `match(regex)` – returns matches of a regex in the string.

23. `matchAll(regex)` – returns an iterator of all regex matches.

24. `search(regex)` – returns the index of the first regex match.

🔽 Medium Priority (useful but not in every project)

25. `padStart(targetLength, padString)` – pads the string from the start until it reaches the target length.

26. `padEnd(targetLength, padString)` – pads the string from the end until it reaches the target length.
27. `localeCompare(compareString)` – compares two strings based on locale rules.
28. `normalize(form)` – normalizes Unicode characters into a consistent form.
29. `valueOf()` – returns the primitive string value of a String object.
30. `toString()` – returns the string itself (useful when dealing with objects).

⬇️ Low Priority (rare or niche cases)

31. `fromCharCode(...codes)` *(static)* – creates a string from Unicode values.
32. `fromCodePoint(...codes)` *(static)* – creates a string from Unicode code points.
33. `raw(template, ...substitutions)` *(static)* – returns raw string from template literals.
34. `codePointAt(index)` – returns the Unicode code point of a character.
35. `anchor(name)`, `big()`, `blink()`, `bold()`, `fixed()`, `fontcolor()`, `fontsize()`, `italics()`, `link(url)`, `small()`, `strike()`, `sub()`, `sup()` – old/deprecated HTML wrapper methods (almost never used today).

👉 This list goes **from most useful in daily coding tasks → to rarely needed/deprecated ones**.

Would you like me to also **categorize them by task** (e.g., searching, modifying, formatting, etc.) so you can learn them in logical groups instead of just priority?