JavaScript Regex Rules and Syntax

1. What is a Regex in JavaScript?
A regular expression (regex) is a pattern used to match strings.
You can use regex in JS for:
- Searching (search, match, test)
- Replacing (replace)
- Validating inputs (emails, passwords, etc.)

2. Creating a Regex
In JS, there are two ways to create a regex:

```
// Literal notation
let pattern1 = /abc/;

// Constructor function
let pattern2 = new RegExp("abc");
```

3. Regex Syntax in JavaScript

Basic Characters:
abc - Matches exactly "abc"
. - Matches any single character except newline
\d - Digit (0–9)
\D - Non-digit
\w - Word character (A–Z, a–z, 0–9, _)
\W - Non-word character
\s - Whitespace (space, tab, newline)
\S - Non-whitespace
\t - Tab
\n - Newline

Anchors (Position Matching):
^ - Start of string
$ - End of string
\b - Word boundary
\B - Non-word boundary

Quantifiers:
* - 0 or more times
+ - 1 or more times
? - 0 or 1 time (optional)
{n} - Exactly n times
{n,} - At least n times
{n,m} - Between n and m times

Character Sets:
[abc] - Matches a, b, or c
[^abc] - Matches any char except a, b, c
[a-z] - Lowercase a to z
[A-Z] - Uppercase A to Z
[0-9] - Digits 0 to 9

Groups:
(abc) - Capturing group
(?:abc) - Non-capturing group
(a|b) - a or b (alternation)

Escaping:
To match special regex characters (. ^ $ * + ? { } [ ] \ | ( )) literally, escape them with \.

Flags in JS:
g - Global match (find all matches)
i - Case-insensitive
m - Multiline
s - Allows . to match newline
u - Unicode mode
y - Sticky matching

Example usage in JS:

```
let str = "Hello world 123";

// test() → true/false
console.log(/\d+/.test(str)); // true

// match() → returns matches
console.log(str.match(/\d+/)); // ["123"]

// replace()
console.log(str.replace(/\d+/, "XYZ")); // "Hello world XYZ"

// search()
console.log(str.search(/\d+/)); // index where match starts
```

Example — Password Validation:

```
let password = "Abc@1234";
let regex = /^(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[\W_]).{8,}$/;
console.log(regex.test(password)); // true if valid
```

This means:
- (?=.*[A-Z]) → at least 1 uppercase
- (?=.*[a-z]) → at least 1 lowercase
- (?=.*\d) → at least 1 digit
- (?=.*[\W_]) → at least 1 special char
- .{8,} → minimum length 8