**🧩 1. length()**

**👉 What it does:** Returns the number of characters in a string.  
**🕓 When to use:** When you need the size or length of a string (for loops, validations, etc.).  
**💡 Why:** Helps in iteration and checking if a string is empty.

String s = "Hello";

System.out.println(s.length()); // 5

**🧩 2. charAt(int index)**

**👉 What it does:** Returns the character at a specific position.  
**🕓 When to use:** When you need to access a single character from a string.  
**💡 Why:** Useful in string traversal or character-based operations.

String s = "Java";

System.out.println(s.charAt(2)); // v

**🧩 3. substring(int beginIndex, int endIndex)**

**👉 What it does:** Extracts part of a string.  
**🕓 When to use:** When you want to get a specific portion of a string (like username from email).  
**💡 Why:** Simplifies string slicing.

String s = "Programming";

System.out.println(s.substring(3, 6)); // gra

**🧩 4. equals(String anotherString)**

**👉 What it does:** Checks if two strings are exactly the same (case-sensitive).  
**🕓 When to use:** When comparing string content (like passwords, input text).  
**💡 Why:** == compares memory addresses, but equals() compares actual content.

System.out.println("Java".equals("Java")); // true

**🧩 5. equalsIgnoreCase(String anotherString)**

**👉 What it does:** Compares strings ignoring uppercase/lowercase.  
**🕓 When to use:** When case doesn’t matter (like usernames, commands).  
**💡 Why:** Avoids false mismatches due to letter case.

System.out.println("Hello".equalsIgnoreCase("hello")); // true

**🧩 6. compareTo(String anotherString)**

**👉 What it does:** Compares two strings lexicographically (dictionary order).  
**🕓 When to use:** When sorting or comparing alphabetical order.  
**💡 Why:** Returns a number — helps in sorting logic.

System.out.println("apple".compareTo("banana")); // negative value

**🧩 7. compareToIgnoreCase(String anotherString)**

**👉 What it does:** Same as compareTo() but ignores case.  
**🕓 When to use:** When sorting but case shouldn’t affect order.  
**💡 Why:** Keeps sorting consistent regardless of case.

System.out.println("APPLE".compareToIgnoreCase("apple")); // 0

**🧩 8. toLowerCase() / toUpperCase()**

**👉 What it does:** Converts all letters to lower/upper case.  
**🕓 When to use:** When comparing text, formatting, or standardizing input.  
**💡 Why:** Ensures consistency for case-insensitive operations.

System.out.println("Java".toUpperCase()); // JAVA

**🧩 9. trim()**

**👉 What it does:** Removes spaces from the beginning and end.  
**🕓 When to use:** Before saving or comparing user input.  
**💡 Why:** Avoids unwanted spaces causing mismatches.

String s = " Hello ";

System.out.println(s.trim()); // "Hello"

**🧩 10. contains(CharSequence seq)**

**👉 What it does:** Checks if a substring exists in the string.  
**🕓 When to use:** To verify if a keyword or phrase appears.  
**💡 Why:** Helpful in search and validation features.

System.out.println("Hello World".contains("World")); // true

**🧩 11. replace(char oldChar, char newChar)**

**👉 What it does:** Replaces one character with another.  
**🕓 When to use:** When fixing typos or formatting text.  
**💡 Why:** Quick way to clean or modify text.

System.out.println("banana".replace('a', 'o')); // bonono

**🧩 12. replaceAll(String regex, String replacement)**

**👉 What it does:** Replaces all substrings matching a pattern.  
**🕓 When to use:** When you need to replace multiple or pattern-based matches.  
**💡 Why:** Works with regex for advanced replacements.

System.out.println("I like apples".replaceAll("apples", "mangoes")); // I like mangoes

**🧩 13. indexOf() / lastIndexOf()**

**👉 What it does:** Finds the index of the first or last occurrence of a character or substring.  
**🕓 When to use:** When you need to locate a specific word or symbol.  
**💡 Why:** Useful in parsing and search operations.

String s = "banana";

System.out.println(s.indexOf('a')); // 1

System.out.println(s.lastIndexOf('a')); // 5

**🧩 14. startsWith(String prefix) / endsWith(String suffix)**

**👉 What it does:** Checks if a string starts or ends with specific text.  
**🕓 When to use:** When validating file types, URLs, or prefixes.  
**💡 Why:** Common in validation and filtering.

System.out.println("file.txt".endsWith(".txt")); // true

**🧩 15. isEmpty()**

**👉 What it does:** Checks if a string has zero length.  
**🕓 When to use:** To validate user input or prevent errors.  
**💡 Why:** Avoids NullPointerException or unnecessary operations.

String s = "";

System.out.println(s.isEmpty()); // true

**🧩 16. split(String regex)**

**👉 What it does:** Splits a string into parts using a delimiter.  
**🕓 When to use:** To break text (like CSV, emails, words).  
**💡 Why:** Simplifies text tokenization.

String s = "apple,banana,grape";

String[] fruits = s.split(",");

**🧩 17. join(CharSequence delimiter, CharSequence... elements)**

**👉 What it does:** Joins multiple strings with a separator.  
**🕓 When to use:** When building CSV or readable output.  
**💡 Why:** Cleaner than manual concatenation.

String joined = String.join("-", "Java", "is", "fun");

System.out.println(joined); // Java-is-fun

**🧩 18. concat(String str)**

**👉 What it does:** Combines two strings.  
**🕓 When to use:** When joining strings directly.  
**💡 Why:** Similar to using +, but more explicit.

String s1 = "Hello";

String s2 = "World";

System.out.println(s1.concat(s2)); // HelloWorld

**🧩 19. toCharArray()**

**👉 What it does:** Converts string into an array of characters.  
**🕓 When to use:** When you need to modify or analyze each character.  
**💡 Why:** Helps in algorithms or manual manipulation.

char[] arr = "Hello".toCharArray();

**🧩 20. valueOf()**

**👉 What it does:** Converts other data types to string.  
**🕓 When to use:** When combining numbers with strings.  
**💡 Why:** Prevents errors during concatenation.

int n = 10;

String s = String.valueOf(n);

System.out.println(s + 5); // 105

**🧩 21. format(String format, Object... args)**

**👉 What it does:** Formats a string using placeholders (like printf).  
**🕓 When to use:** To display formatted numbers, dates, or text.  
**💡 Why:** Cleaner output with proper alignment.

String s = String.format("Name: %s, Age: %d", "Deb", 21);

System.out.println(s);

**🧩 22. matches(String regex)**

**👉 What it does:** Checks if the entire string matches a pattern.  
**🕓 When to use:** To validate formats like email or phone number.  
**💡 Why:** Powerful for input validation.

System.out.println("abc123".matches("[a-z]+[0-9]+")); // true

| **String Methids in Java** |
| --- |

|  |  |  |
| --- | --- | --- |
| **length()** | :Find number of characters | **Used in**:Looping, validation |

|  |  |  |
| --- | --- | --- |
| charAt() | Get a specific character | **Used in**:Character operations |

|  |  |  |
| --- | --- | --- |
| substring() | Extract part of string | **Used in**:String slicing |

|  |  |  |
| --- | --- | --- |
| equals() | Compare strings | **Used in**:Exact match check |

|  |  |  |
| --- | --- | --- |
| equalsIgnoreCase() | Compare ignoring case | Case-insensitive comparison |

|  |  |  |
| --- | --- | --- |
| compareTo() | Lexicographical comparison | Sorting order |

|  |  |  |
| --- | --- | --- |
| toLowerCase()/toUpperCase() | Change case | Normalization |

|  |  |  |
| --- | --- | --- |
| trim() | Remove extra spaces | Clean input |

|  |  |  |
| --- | --- | --- |
| contains() | Check substring | Search |

|  |  |  |
| --- | --- | --- |
| replace()/replaceAll() | Replace text | Modify strings |

|  |  |  |
| --- | --- | --- |
| indexOf()/lastIndexOf() | Find position | Searching |

|  |  |  |
| --- | --- | --- |
| startsWith()/endsWith() | Check beginning/end | Validation |

|  |  |  |
| --- | --- | --- |
| isEmpty() | Check empty string | Input check |

|  |  |  |
| --- | --- | --- |
| split() | Break string into parts | Tokenization |

|  |  |  |
| --- | --- | --- |
| join() | Combine parts | Building output |

|  |  |  |
| --- | --- | --- |
| concat() | Combine strings | Simple joining |

|  |  |  |
| --- | --- | --- |
| toCharArray() | Convert to char array | Character access |

|  |  |  |
| --- | --- | --- |
| valueOf() | Convert to string | Type conversion |

|  |  |  |
| --- | --- | --- |
| format() | Format strings | Pretty output |

|  |  |  |
| --- | --- | --- |
| matches() | Pattern check | Validation |