1. String

- **Immutable**: Cannot be modified after creation (operations return new objects).
- **Thread-safe**: Safe for multi-threaded environments.
- **Use Case**: For fixed text that won't change frequently.

Examples:

```
// Example 1: Immutability
String str1 = "Hello";
str1.concat(" World");  // Returns new string
System.out.println(str1);  // Output: "Hello" (original unchanged)


// Example 2: Inefficient concatenation in loops
String result = "";
for (int i = 0; i < 5; i++) {
   result += i;  // Creates 5 new String objects
}
System.out.println(result);  // Output: "01234"
```

---

2. StringBuffer

- **Mutable**: Can modify content after creation.
- **Thread-safe**: Synchronized methods (safe for multi-threading).
- **Use Case**: For heavy text manipulation in multi-threaded apps.

Examples:

```
// Example 1: Efficient appending
StringBuffer buffer = new StringBuffer("Hello");
buffer.append(" World");  // Modifies the same object
System.out.println(buffer);  // Output: "Hello World"

// Example 2: Thread-safe reversal
buffer.reverse();
System.out.println(buffer);  // Output: "dlroW olleH"
```

---

3. StringBuilder

- **Mutable**: Can modify content after creation.
- **Not thread-safe**: Faster than StringBuffer in single-threaded apps.
- **Use Case**: For heavy text manipulation in single-threaded apps.

Examples:

```
// Example 1: Efficient concatenation
StringBuilder builder = new StringBuilder();
for (int i = 0; i < 5; i++) {
   builder.append(i);  // No new objects created
}
System.out.println(builder);  // Output: "01234"


// Example 2: Insert/Delete operations
builder.insert(2, "X");
System.out.println(builder);  // Output: "01X234"
builder.delete(2, 3);
System.out.println(builder);  // Output: "01234"
```

Key Differences

Feature	String	StringBuffer	StringBuilder
Mutability	Immutable	Mutable	Mutable
Thread Safety	Yes (inherent)	Yes (synchronized)	No
Performance	Slow for edits	Moderate	Fastest
Use Case	Fixed text	Multi-threaded	Single-threaded

Tags

Two pointers

Sorting

Hash table

Prefix Sum

Heap(priority Queue)

Stack

Day 1

1249. Minimum Remove to Make Valid Parentheses

227. Basic Calculator II

3. Longest Substring Without Repeating Characters

Day 2

179. Largest Number

49. Group Anagrams

1371. Find the Longest Substring Containing Vowels in Even Counts

Day 3

916. Word Subsets

3163. String Compression III

1405. Longest Happy String

Day 4

2938. Separate Black and White Balls

2914. Minimum Number of Changes to Make Binary String Beautiful

1930. Unique Length-3 Palindromic Subsequences

Day 5

2337. Move Pieces to Obtain a String

2337. Move Pieces to Obtain a String

394. Decode String