# **OBJECT ORIENTED PROGRAMMING AND DESIGN**

**Strings**

1. **String Constructor Examples**

public class Main

{

public static void main(String[] args)

{

String emptyString = new String();

System.out.println("Empty String: \"" + emptyString + "\"");

char[] charArray = {'H', 'e', 'l', 'l', 'o'};

String stringFromCharArray = new String(charArray);

System.out.println("String from char array: " + stringFromCharArray);

String stringFromCharArrayPart = new String(charArray, 1, 3);

System.out.println("String from part of char array: " + stringFromCharArrayPart);

// 4. String(String str) – Creates a string from another string

String anotherString = new String("Hello World");

String stringFromAnotherString = new String(anotherString);

System.out.println("String from another String: " + stringFromAnotherString);

// 5. String(byte[] bytes) – Creates a string from byte array

byte[] byteArray = {72, 101, 108, 108, 111};

String stringFromByteArray = new String(byteArray);

System.out.println("String from byte array: " + stringFromByteArray);

// 6. String(byte bytes[], int start, int numChars) – Creates a string from byte array starting from index 'start' to 'start + numChars - 1'

byte[] byteArrayPart = {72, 101, 108, 108, 111, 87, 111, 114, 108, 100};

String stringFromByteArrayPart = new String(byteArrayPart, 6, 2);

System.out.println("String from part of byte array: " + stringFromByteArrayPart);

}

}

**2. The usage of the equals() method to compare two strings:**

public class Main {

public static void main(String[] args) {

// Create two strings

String s1 = "xyz";

String s2 = "abc";

String s3 = "xyz";

// Compare s1 and s2

System.out.println("s1.equals(s2): " + s1.equals(s2)); // Returns false because "xyz" is not equal to "abc"

// Compare s1 and s3

System.out.println("s1.equals(s3): " + s1.equals(s3)); // Returns true because both are "xyz"

// Compare s2 and s3

System.out.println("s2.equals(s3): " + s2.equals(s3)); // Returns false because "abc" is not equal to "xyz"

// Compare with different case (e.g., "abc" vs "ABC")

String s4 = "ABC";

System.out.println("s2.equals(s4): " + s2.equals(s4)); // Returns false because "abc" is not equal to "ABC"

// Compare strings with the same value but different references

String s5 = new String("xyz");

System.out.println("s1.equals(s5): " + s1.equals(s5)); // Returns true because the content is the same

}

}

**3. The usage of the length() method:**

public class Main {

public static void main(String[] args) {

// Create strings of different lengths

String s1 = "Hello, World!";

String s2 = "Java";

String s3 = "";

// Get the length of each string

int length1 = s1.length(); // Length of "Hello, World!" is 13

int length2 = s2.length(); // Length of "Java" is 4

int length3 = s3.length(); // Length of an empty string is 0

// Print the lengths

System.out.println("Length of s1: " + length1); // Output: 13

System.out.println("Length of s2: " + length2); // Output: 4

System.out.println("Length of s3: " + length3); // Output: 0

}

}

**4. Concatenate (join) strings using the + operator:**

public class Main {

public static void main(String[] args) {

// Declare strings

String firstName = "John";

String lastName = "Doe";

String greeting = "Hello";

// Concatenate strings using '+' operator

String fullName = firstName + " " + lastName; // Concatenates first name and last name with a space

String fullGreeting = greeting + ", " + fullName + "!"; // Concatenates greeting and full name

// Print the concatenated strings

System.out.println("Full Name: " + fullName); // Output: Full Name: John Doe

System.out.println("Full Greeting: " + fullGreeting); // Output: Full Greeting: Hello, John Doe!

}

}

**5. Concatenate strings using the concat() method:**

public class Main {

public static void main(String[] args) {

// Declare strings

String firstName = "Jane";

String lastName = "Smith";

String greeting = "Welcome";

// Concatenate strings using concat() method

String fullName = firstName.concat(" ").concat(lastName); // Concatenate first name and last name with a space

String fullGreeting = greeting.concat(", ").concat(fullName).concat("!"); // Concatenate greeting and full name

// Print the concatenated strings

System.out.println("Full Name: " + fullName); // Output: Full Name: Jane Smith

System.out.println("Full Greeting: " + fullGreeting); // Output: Full Greeting: Welcome, Jane Smith!

}

}

**6. The charAt() method in Java:**

public class Main {

public static void main(String[] args) {

// Declare a string

String text = "Hello, World!";

// Extract a character at a specific index using charAt()

char firstChar = text.charAt(0); // First character (index 0)

char fifthChar = text.charAt(4); // Fifth character (index 4)

char lastChar = text.charAt(text.length() - 1); // Last character

// Print the extracted characters

System.out.println("First character: " + firstChar); // Output: H

System.out.println("Fifth character: " + fifthChar); // Output: o

System.out.println("Last character: " + lastChar); // Output: !

}

}

**7. The getChars() method:**

public class Main {

public static void main(String[] args) {

// Declare a string

String text = "Hello, World!";

//|0|1|2|3|4|5|6|7|8|9|10|11

// H e l l o , W o r l d

// Create a character array to hold the extracted characters

char[] charArray = new char[5];

// Extract characters from the string and store them in the charArray

text.getChars(7, 12, charArray, 0); // Extract characters from index 7 to 11 ("World")

// Print the character array

System.out.print("Extracted characters: ");

for (char c : charArray) {

System.out.print(c); // Output: World

}

}

}

**8. The getBytes() method:**

public class Main {

public static void main(String[] args) {

// Declare a string

String text = "Hello, World!";

// Convert the string into a byte array using getBytes()

byte[] byteArray = text.getBytes();

// Print the byte array

System.out.print("Byte array: ");

for (byte b : byteArray) {

System.out.print(b + " "); // Print each byte value

}

}

}

**9. The toCharArray() method:**

public class Main {

public static void main(String[] args) {

// Declare a string

String text = "Hello, World!";

// Convert the string into a character array using toCharArray()

char[] charArray = text.toCharArray();

// Print the character array

System.out.print("Character array: ");

for (char c : charArray) {

System.out.print(c + " "); // Output: H e l l o , W o r l d !

}

}

}

**10. Example demonstrating both equals() and equalsIgnoreCase():**

public class Main {

public static void main(String[] args) {

// Declare some strings

String str1 = "Hello";

String str2 = "hello";

String str3 = "Hello";

// Using equals() method (case-sensitive comparison)

System.out.println("Using equals():");

System.out.println("str1.equals(str2): " + str1.equals(str2)); // false, because of different cases

System.out.println("str1.equals(str3): " + str1.equals(str3)); // true, because both are "Hello"

// Using equalsIgnoreCase() method (case-insensitive comparison)

System.out.println("\nUsing equalsIgnoreCase():");

System.out.println("str1.equalsIgnoreCase(str2): " + str1.equalsIgnoreCase(str2)); // true, case is ignored

System.out.println("str1.equalsIgnoreCase(str3): " + str1.equalsIgnoreCase(str3)); // true, because both are "Hello"

}

}

**11. The regionMatches() method:**

public class Main {

public static void main(String[] args) {

// Declare some strings

String str1 = "Hello, World!";

String str2 = "World";

String str3 = "world";

// Using regionMatches() to compare regions

System.out.println("Using regionMatches() - Case-sensitive:");

// Compare substring of str1 (starting from index 7) with str2 (starting from index 0)

System.out.println("str1.regionMatches(7, str2, 0, 5): " + str1.regionMatches(7, str2, 0, 5)); // true

// Compare substring of str1 (starting from index 7) with str3 (starting from index 0)

System.out.println("str1.regionMatches(7, str3, 0, 5): " + str1.regionMatches(7, str3, 0, 5)); // false

// Using regionMatches() with case-insensitive comparison

System.out.println("\nUsing regionMatches() - Case-insensitive:");

System.out.println("str1.regionMatches(true, 7, str3, 0, 5): " + str1.regionMatches(true, 7, str3, 0, 5)); // true

}

}

**12. startsWith() Method & endsWith() Method:**

public class Main {

public static void main(String[] args) {

// Declare some strings

String str1 = "Hello, World!";

String str2 = "Java Programming";

String str3 = "Welcome to the Java World";

// Using startsWith() method

System.out.println("Using startsWith():");

System.out.println("str1.startsWith(\"Hello\"): " + str1.startsWith("Hello")); // true

System.out.println("str2.startsWith(\"Java\"): " + str2.startsWith("Java")); // true

System.out.println("str3.startsWith(\"Hello\"): " + str3.startsWith("Hello")); // false

// Using endsWith() method

System.out.println("\nUsing endsWith():");

System.out.println("str1.endsWith(\"World!\"): " + str1.endsWith("World!")); // true

System.out.println("str2.endsWith(\"Programming\"): " + str2.endsWith("Programming")); // true

System.out.println("str3.endsWith(\"World\"): " + str3.endsWith("World")); // false

}

}

**13. The compareTo() method:**

public class Main {

public static void main(String[] args) {

// Declare some strings

String str1 = "Apple";

String str2 = "Banana";

String str3 = "Apple";

String str4 = "apple";

// Using compareTo() method

System.out.println("Using compareTo():");

System.out.println("str1.compareTo(str2): " + str1.compareTo(str2)); // Negative value (Apple < Banana)

System.out.println("str1.compareTo(str3): " + str1.compareTo(str3)); // 0 (Apple == Apple)

System.out.println("str1.compareTo(str4): " + str1.compareTo(str4)); // Negative value (Apple < apple)

}

}

**14. The indexOf() and lastIndexOf() methods:**

public class Main {

public static void main(String[] args) {

// Declare a string

String str = "Java programming is fun and Java is popular.";

// Using indexOf() method

System.out.println("Using indexOf():");

System.out.println("str.indexOf('a'): " + str.indexOf('a')); // Returns index of first 'a' (1)

System.out.println("str.indexOf('Java'): " + str.indexOf("Java")); // Returns index of first "Java" (0)

System.out.println("str.indexOf('Java', 5): " + str.indexOf("Java", 5)); // Returns index of "Java" after index 5 (29)

System.out.println("str.indexOf('xyz'): " + str.indexOf("xyz")); // Returns -1, as "xyz" is not found

// Using lastIndexOf() method

System.out.println("\nUsing lastIndexOf():");

System.out.println("str.lastIndexOf('a'): " + str.lastIndexOf('a')); // Returns index of last 'a' (43)

System.out.println("str.lastIndexOf('Java'): " + str.lastIndexOf("Java")); // Returns index of last "Java" (29)

System.out.println("str.lastIndexOf('Java', 20): " + str.lastIndexOf("Java", 20)); // Returns index of "Java" before index 20 (0)

System.out.println("str.lastIndexOf('xyz'): " + str.lastIndexOf("xyz")); // Returns -1, as "xyz" is not found

}

}

**15. The substring() method:**

public class Main {

public static void main(String[] args) {

// Declare a string

String str = "Java programming is fun";

// Using substring() with one argument (beginIndex)

System.out.println("Using substring(beginIndex):");

System.out.println("str.substring(5): " + str.substring(5)); // Extracts substring from index 5 to the end

// Using substring() with two arguments (beginIndex, endIndex)

System.out.println("\nUsing substring(beginIndex, endIndex):");

System.out.println("str.substring(0, 4): " + str.substring(0, 4)); // Extracts substring from index 0 to 3

System.out.println("str.substring(5, 16): " + str.substring(5, 16)); // Extracts substring from index 5 to 15

// Edge cases

System.out.println("\nEdge case examples:");

System.out.println("str.substring(0, 0): " + str.substring(0, 0)); // Empty string, as beginIndex and endIndex are the same

}

}

**16. The StringBuffer class in Java:**

public class Main {

public static void main(String[] args) {

// Creating a StringBuffer object

StringBuffer sb = new StringBuffer("Hello");

// 1. length() - returns the number of characters

System.out.println("Length: " + sb.length()); // Output: Length: 5

// 2. capacity() - returns the current capacity

System.out.println("Capacity: " + sb.capacity()); // Output: Capacity: 16

// 3. append() - appending text

sb.append(" World");

System.out.println("After append: " + sb); // Output: Hello World

// 4. insert() - inserting text at specified index

sb.insert(5, " Java");

System.out.println("After insert: " + sb); // Output: Hello Java World

// Insert integer at position

sb.insert(4, 123);

System.out.println("After inserting integer: " + sb); // Output: Hell123o Java World

}

}

**17. Example Using Both delete() and deleteCharAt():**

public class Main {

public static void main(String[] args) {

// Creating a StringBuffer object

StringBuffer sb = new StringBuffer("Hello World");

// Using delete(int startIndex, int endIndex)

sb.delete(5, 11); // Removes substring from index 5 to 10 (i.e., " World")

System.out.println("After delete(): " + sb); // Output: Hello

// Using deleteCharAt(int index)

sb.append(" World"); // Re-adding the string

sb.deleteCharAt(5); // Removes the character at index 5 (the space)

System.out.println("After deleteCharAt(): " + sb); // Output: HelloWorld

}

}

**18. Example Using Both replace() and toString():**

public class Main {

public static void main(String[] args) {

// Creating a StringBuffer object

StringBuffer sb = new StringBuffer("Hello World");

// Using replace(int startIndex, int endIndex, String str)

sb.replace(6, 11, "Java"); // Replaces "World" with "Java"

System.out.println("After replace(): " + sb); // Output: Hello Java

// Converting StringBuffer to String using toString()

String str = sb.toString();

System.out.println("Converted String: " + str); // Output: Hello Java

}

}