LAB # 01

INTRODUCTION TO STRING POOL, LITERALS, AND WRAPPER CLASSES

OBJECTIVE: To study the concepts of String Constant Pool, String literals, String immutability and Wrapper classes.

LAB TASKS

1. Write a program that initialize five different strings using all the above mentioned ways, i.e., a)string literals b)new keyword also use intern method and show string immutability.

CODE

```java
package java3;
public class Java3 {
    public static void main(String[] args) {
        String str1="hello"; // String Initialization using String Literal
        String str2=new String("hello");// String Initialization using new keyword
        String str3=str2.intern();// Returns reference from string pool
        String str4= new String ("world").intern(); // String Initialization with new keyword and intern method combi
        String str5 = "Hello" + "World"; // String Initialization using concatenation, which results in a new object
        System.out.println("str1: " + str1);
        System.out.println("str2: " + str2);
        System.out.println("str3: " + str3);
        System.out.println("str4: " + str4);
        System.out.println("str5: " + str5);
        // Checking references to show immutability and intern behavior
        System.out.println("\nComparing references:");
        System.out.println("str1 == str2: " + (str1 == str2));
        System.out.println("str1 == str3: " + (str1 == str3));
        System.out.println("str3 == str2.intern(): " + (str3 == str2.intern()));
        System.out.println("str4 == \"World\": " + (str4 == "World"));

    }

}
```

OUTPUT:

```
run:
str1: hello
str2: hello
str3: hello
str4: world
str5: HelloWorld

Comparing references:
str1 == str2: false
str1 == str3: true
str3 == str2.intern(): true
str4 == "World": false
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Write a program to convert primitive data type Double into its respective wrapper object.

CODE

```java
package java3;
public class Java3 {
    public static void main(String[] args) {
        // Primitive data type
        double primitiveDouble = 10.5;
        // Convert primitive double to wrapper Double
        Double wrapperDouble = Double.valueOf(primitiveDouble);
        System.out.println("Primitive double: " + primitiveDouble);
        System.out.println("Wrapper Double: " + wrapperDouble);

    }
 }
```

OUTPUT:

```
run:
Primitive double: 10.5
Wrapper Double: 10.5
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. Write a program that initialize five different strings and perform the following operations. a. Concatenate all five stings. b. Convert fourth string to uppercase. c. Find the substring from the concatenated string from 8 to onward

CODE:

```java
package java3;
public class Java3 {
    public static void main(String[] args) {
        String str1 = "Hello";
        String str2 = "Java";
        String str3 = "World";
        String str4 = "Programming";
        String str5 = "Language";
        String concatenatedString = str1 + str2 + str3 + str4 + str5;
        System.out.println("Concatenated String: " + concatenatedString);
        String str4UpperCase = str4.toUpperCase();
        System.out.println("Fourth String in Uppercase: " + str4UpperCase);
        String substringFrom8 = concatenatedString.substring(8);
        System.out.println("Substring from index 8 onward: " + substringFrom8);
    }
 }
```

OUTPUT:

```
run:
Concatenated String: HelloJavaWorldProgrammingLanguage
Fourth String in Uppercase: PROGRAMMING
Substring from index 8 onward: aWorldProgrammingLanguage
BUILD SUCCESSFUL (total time: 0 seconds)
```

4.You are given two strings word1 and word2. Merge the strings by adding letters in alternating order, starting with word1. If a string is longer than the other, append the additional letters onto the end of the merged string. Return *the merged string*.

Example:

Input: word1 = "abc", word2 = "pqr"
Output: "apbqcr"
Explanation: The merged string will be merged as so:
word1: a b c
word2: p q r
merged: a p b q c r

CODE:

```java
package java3;
public class Java3 {
    public static void main(String[] args) {
        String word1 = "abc"; // First input string
        String word2 = "pqr"; // Second input string

        String merged = mergeAlternately(word1, word2); // Merging the strings
        System.out.println(merged); // Output the merged string
    }

    public static String mergeAlternately(String word1, String word2) {
        String merged = ""; // Initialize the merged string
        int length = Math.max(word1.length(), word2.length()); // Get the length of the longer string

        for (int i = 0; i < length; i++) {
            if (i < word1.length()) { // Check if there's a character in word1
                merged += word1.charAt(i); // Add it to merged
            }
            if (i < word2.length()) { // Check if there's a character in word2
                merged += word2.charAt(i); // Add it to merged
            }
        }

        return merged; // Return the merged string
    }
}
```

OUTPUT:

```
run:
apbqcr
BUILD SUCCESSFUL (total time: 0 seconds)
```

4. Write a Java program to find the minimum and maximum values of Integer, Float, and Double using the respective wrapper class constants

CODE:

```java
package java3;
public class Java3 {
    public static void main(String[] args) {
        // Minimum and Maximum values for Integer
        System.out.println("Integer Min: " + Integer.MIN_VALUE);
        System.out.println("Integer Max: " + Integer.MAX_VALUE);

        // Minimum and Maximum values for Float
        System.out.println("Float Min: " + Float.MIN_VALUE); // Smallest positive value
        System.out.println("Float Max: " + Float.MAX_VALUE);

        // Minimum and Maximum values for Double
        System.out.println("Double Min: " + Double.MIN_VALUE); // Smallest positive value
        System.out.println("Double Max: " + Double.MAX_VALUE);

        System.out.println("Integer Range: " + (Integer.MIN_VALUE) + " to " + (Integer.MAX_VALUE));
        System.out.println("Float Range: " + (Float.MIN_VALUE) + " to " + (Float.MAX_VALUE));
        System.out.println("Double Range: " + (Double.MIN_VALUE) + " to " + (Double.MAX_VALUE));
    }
}
```

OUTPUT:

```
run:
Integer Min: -2147483648
Integer Max: 2147483647
Float Min: 1.4E-45
Float Max: 3.4028235E38
Double Min: 4.9E-324
Double Max: 1.7976931348623157E308
Integer Range: -2147483648 to 2147483647
Float Range: 1.4E-45 to 3.4028235E38
Double Range: 4.9E-324 to 1.7976931348623157E308
BUILD SUCCESSFUL (total time: 0 seconds)
```

HOME TASKS

1. Write a JAVA program to perform Autoboxing and also implement different methods of wrapper class.

CODE:

```java
package java3;
public class Java3 {
    public static void main(String[] args) {
        // Autoboxing: converting primitive int to Integer object
        int primitiveInt = 10;
        Integer wrappedInt = primitiveInt; // Autoboxing
        System.out.println("Autoboxed Integer: " + wrappedInt);
        // Unboxing: converting Integer object back to primitive int
        int unboxedInt = wrappedInt; // Unboxing
        System.out.println("Unboxed int: " + unboxedInt);
        // Using various methods of the Integer wrapper class
        // 1. Converting String to Integer

        String numberString = "123";
        Integer fromString = Integer.valueOf(numberString);
        System.out.println("Integer from String: " + fromString);
        // 2. Finding the maximum and minimum value
        System.out.println("Maximum Integer value: " + Integer.MAX_VALUE);
        System.out.println("Minimum Integer value: " + Integer.MIN_VALUE);

        // 3. Converting Integer to binary, octal, and hexadecimal strings
        System.out.println("Binary representation of " + wrappedInt + ": " + Integer.toBinaryString(wrappedInt));
        System.out.println("Octal representation of " + wrappedInt + ": " + Integer.toOctalString(wrappedInt));
        System.out.println("Hexadecimal representation of " + wrappedInt + ": " + Integer.toHexString(wrappedInt));

        // 4. Comparing two Integer objects
        Integer int1 = 100;
        Integer int2 = 100;
        System.out.println("Are int1 and int2 equal? " + (int1.equals(int2)));
    }
}
```

OUTPUT:

```
run:
Autoboxed Integer: 10
Unboxed int: 10
Integer from String: 123
Maximum Integer value: 2147483647
Minimum Integer value: -2147483648
Binary representation of 10: 1010
Octal representation of 10: 12
Hexadecimal representation of 10: a
Are int1 and int2 equal? true
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Write a Java program to count the number of even and odd digits in a given integer using Autoboxing and Unboxing

CODE:

```
package java3;
import java.util.Scanner;
public class Java3 {
    public static void main(String[] args) {
        System.out.print("Enter an integer: ");
         Scanner input = new Scanner(System.in);
        Integer number = input.nextInt(); // Read and autobox the intege
        int evenCount = 0;
        int oddCount = 0;
        // Unboxing: Convert Integer to int
        int num = number; // Unboxing

        while (num != 0) {
            int digit = num % 10; // Get the last digit
            if (digit % 2 == 0) {
                evenCount++; // Increment even counter
            } else {
                oddCount++; // Increment odd counter
            }
            num /= 10; // Remove the last digit
        }
        System.out.println("Number of even digits: " + evenCount);
        System.out.println("Number of odd digits: " + oddCount);

    }
}
```

OUTPUT

```
Enter an integer: 34
Number of even digits: 1
Number of odd digits: 1
BUILD SUCCESSFUL (total time: 3 seconds)
```

3. Write a Java program to find the absolute value, square root, and power of a number using Math class methods, while utilizing Autoboxing and Wrapper classes.

CODE:

```java
package java3;
import java.util.Scanner;
public class Java3 {
    public static void main(String[] args) {
            Scanner scanner = new Scanner(System.in);

        // Input a number from the user
        System.out.print("Enter a number: ");
        Double number = scanner.nextDouble(); // Read and autobox the number

        // Find the absolute value
        Double absoluteValue = Math.abs(number); // Using Math class method
        System.out.println("Absolute Value: " + absoluteValue);

        // Find the square root
        Double squareRoot = Math.sqrt(number); // Using Math class method
        System.out.println("Square Root: " + squareRoot);

        // Input the exponent for power calculation
        System.out.print("Enter the exponent: ");
        Integer exponent = scanner.nextInt(); // Read and autobox the exponent

        // Find the power
        Double power = Math.pow(number, exponent); // Using Math class method
        System.out.println(number + " raised to the power of " + exponent + " is: " + power);

    }
}
```

OUTPUT:

```
run:
Enter a number: 34
Absolute Value: 34.0
Square Root: 5.830951894845301
Enter the exponent: 36
34.0 raised to the power of 36 is: 1.359067457460612E55
BUILD SUCCESSFUL (total time: 6 seconds)
```

4. Write a Java program to reverse only the vowels in a string.

CODE:

```java
package java3;
import java.util.Scanner;
public class Java3 {
    public static void main(String[] args) {
          Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        // Reverse only the vowels in the string
        String result = reverseVowels(input);
        System.out.println("String with reversed vowels: " + result);
    }
    private static String reverseVowels(String str) {
        StringBuilder vowels = new StringBuilder();
        for (char ch : str.toCharArray()) {
            if (isVowel(ch)) {
                vowels.append(ch); // Add the vowel to the StringBuilder
            }
        }
        vowels.reverse();
        StringBuilder result = new StringBuilder();
        int vowelIndex = 0; // Index for vowels
        // Rebuild the string, replacing vowels with the reversed ones
        for (char ch : str.toCharArray()) {
            if (isVowel(ch)) {
                result.append(vowels.charAt(vowelIndex)); // Replace with the reversed vowel
                vowelIndex++; // Move to the next vowel
            } else {
                result.append(ch); // Keep consonants as they are
            }
        }
        return result.toString(); // Return the modified string
    }
```

```java
    private static boolean isVowel(char ch) {
        return "aeiouAEIOU".indexOf(ch) != -1; // Check if character is a vowel
    }
}
```

OUTPUT:

```
run:
Enter a string: AEIOU
String with reversed vowels: UOIEA
BUILD SUCCESSFUL (total time: 3 seconds)
```

5. Write a Java program to find the longest word in a sentence

CODE

```java
package java3;
import java.util.Scanner;
public class Java3 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Input a sentence from the user
        System.out.print("Enter a sentence: ");
        String sentence = scanner.nextLine();
        // Find the longest word
        String longestWord = findLongestWord(sentence);
        // Output the longest word
        System.out.println("The longest word is: " + longestWord);
        scanner.close(); // Close the scanner
    }
    private static String findLongestWord(String sentence) {
        // Split the sentence into words using space as a delimiter
        String[] words = sentence.split(" ");
        String longestWord = "";
        // Loop through each word to find the longest one
        for (String word : words) {
            // Check if the current word is longer than the longest found so far
            if (word.length() > longestWord.length()) {
                longestWord = word; // Update the longest word
            }
        }
        return longestWord; // Return the longest word
    }
}
```

OUTPUT

```
run:
Enter a sentence: I LOVE PROGRAMMING
The longest word is: PROGRAMMING
BUILD SUCCESSFUL (total time: 8 seconds)
```