

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:

TASK#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.

```
1 package labtask1;
2
3
4 public class Labtask1 {
5
6     public static void main(String[] args) {
7         String s1="data";
8         String s2="structure";
9         String s3="algorithm";
10        String s4="java";
11        String s5="python";
12        String s6=new String("data");
13        String s7=new String("structure");
14        String s8=new String("algorithm");
15        String s9=new String("java");
16        String s10=new String("python");
17        System.out.println(s1);
18        System.out.println(s2);
19        System.out.println(s3);
20        System.out.println(s4);
21        System.out.println(s5);
22        System.out.println(s6);
23        System.out.println(s7);
24        System.out.println(s8);
25        System.out.println(s9);
```

```

26      System.out.println(s10);
27      // using intern method
28      String A=s3.intern();
29      System.out.println(A);
30      //showing string immutability
31      String B="DS";
32      B.concat("A");
33      System.out.println("B");
34  }
35
36  }

```

Output:

```

Output - labtask1 (run) x
data
structure
algorithm
java
python
data
structure
algorithm
java
python
algorithm
B
BUILD SUCCESSFUL (total time: 0 seconds)

```

Task#2

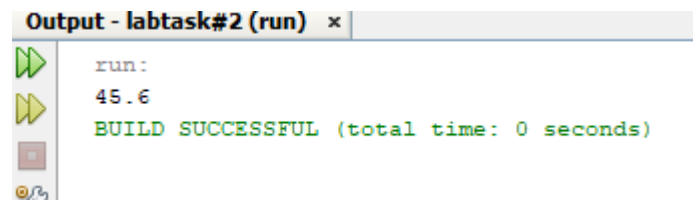
Write a program to convert primitive data type Double into its respective wrapper object.

```

1
2      package labtask.pkg2;
3
4      public class Labtask2 {
5
6          public static void main(String[] args) {
7              double primitiveD=45.6;
8              Double wrapperD=primitiveD;
9              System.out.println(wrapperD);
10         }
11
12     }
13

```

Output:



```
Output - labtask#2 (run) x
run:
45.6
BUILD SUCCESSFUL (total time: 0 seconds)
```

Task#3

Write a program that initialize five different strings and perform the following operations.

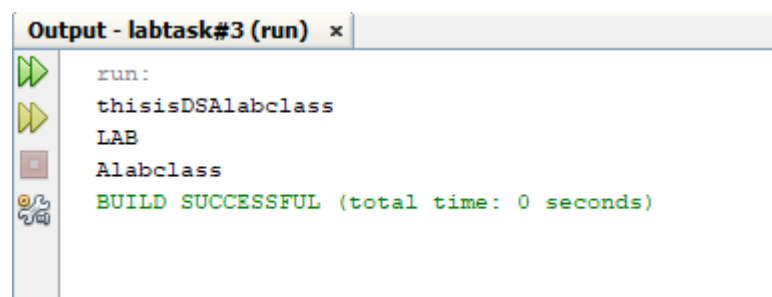
- Concatenate all five strings.
- Convert fourth string to uppercase.
- Find the substring from the concatenated string from 8 to onward

```

1
2 package labtask.pkg3;
3
4 public class Labtask3 {
5
6     public static void main(String[] args) {
7         String s1="this";
8         String s2="is";
9         String s3="DSA";
10        String s4="lab";
11        String s5="class";
12        String concat=s1+s2+s3+s4+s5;
13        System.out.println(concat);
14        String up=s4.toUpperCase();
15        System.out.println(up);
16        String sub=concat.substring(8);
17        System.out.println(sub);
18    }
19
20 }
21

```

Output:



```
Output - labtask#3 (run) x
run:
thisisDSAlabclass
LAB
Alabclass
BUILD SUCCESSFUL (total time: 0 seconds)
```

Task#4

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

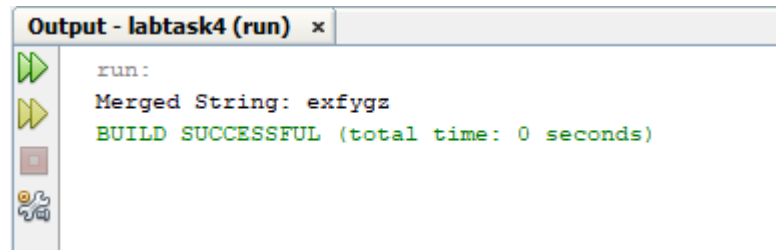
```

1  23
2  24      // Merging characters from both strings
3  25      for (int i = 0; i < maxLength; i++) {
4  26          if (i < length1) {
5  27              merged += s1.charAt(i);
6  28          }
7  29          if (i < length2) {
8  30              merged += s2.charAt(i);
9  31          }
10 32      }
11 33
12 34      return merged;
13 35  }
14 36  }

15
16
17 public static String mergeStrings(String s1, String s2) {
18     String merged = ""; // Initialize an empty string
19
20     int length1 = s1.length();
21     int length2 = s2.length();
22     int maxLength = Math.max(length1, length2);
23
24     // Merging characters from both strings
25     for (int i = 0; i < maxLength; i++) {

```

Output:



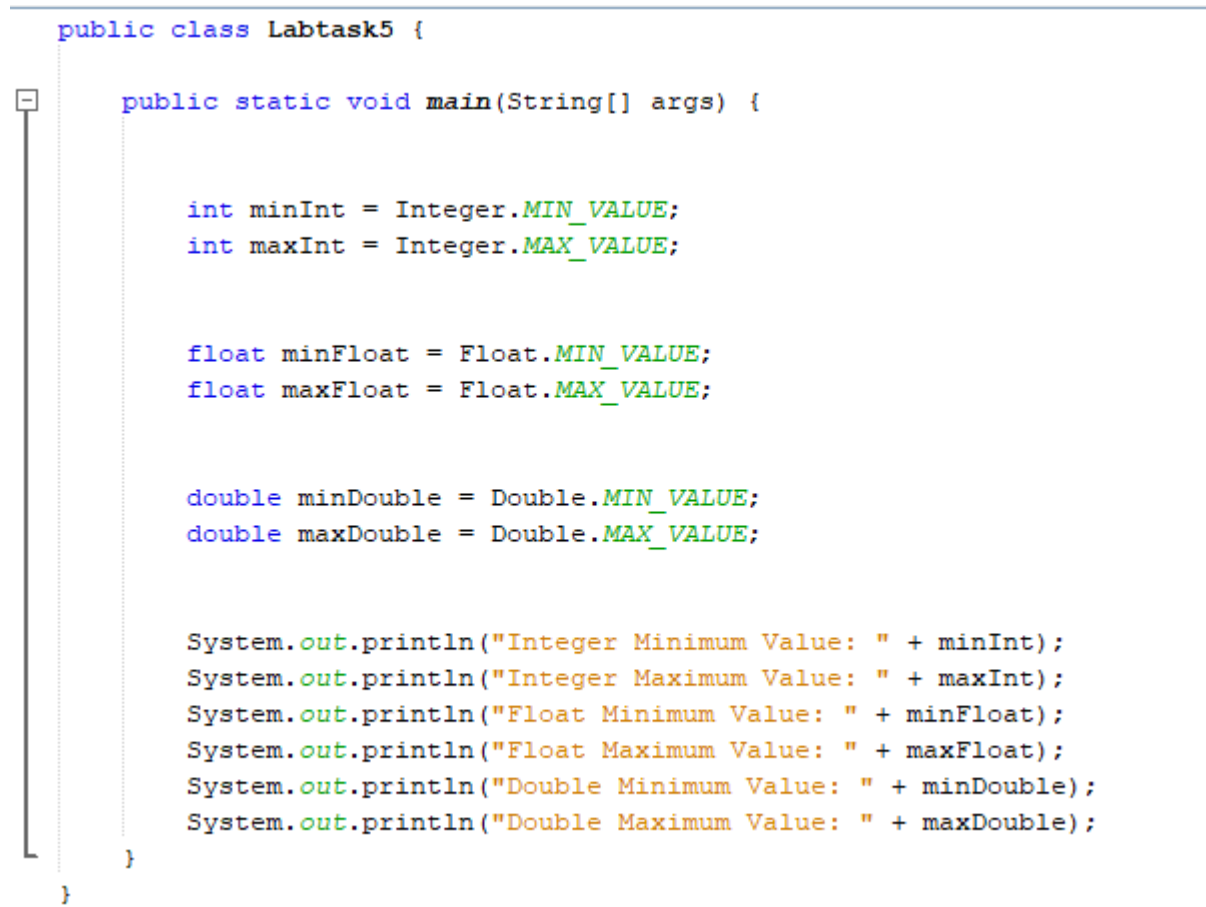
```

Output - labtask4 (run) x
run:
Merged String: exfygz
BUILD SUCCESSFUL (total time: 0 seconds)

```

Task#5:

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



```

public class Labtask5 {

    public static void main(String[] args) {

        int minInt = Integer.MIN_VALUE;
        int maxInt = Integer.MAX_VALUE;

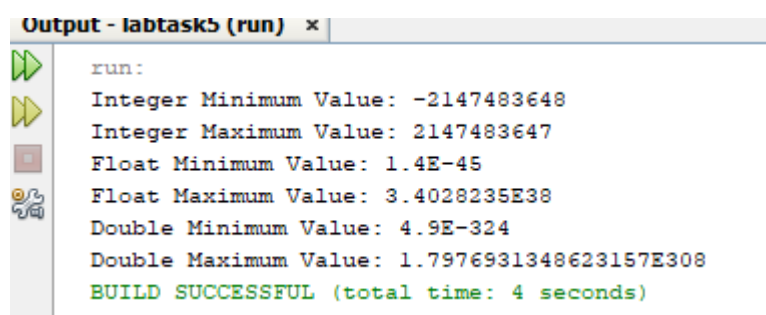
        float minFloat = Float.MIN_VALUE;
        float maxFloat = Float.MAX_VALUE;

        double minDouble = Double.MIN_VALUE;
        double maxDouble = Double.MAX_VALUE;

        System.out.println("Integer Minimum Value: " + minInt);
        System.out.println("Integer Maximum Value: " + maxInt);
        System.out.println("Float Minimum Value: " + minFloat);
        System.out.println("Float Maximum Value: " + maxFloat);
        System.out.println("Double Minimum Value: " + minDouble);
        System.out.println("Double Maximum Value: " + maxDouble);
    }
}

```

Output:



```

Output - labtask5 (run) x
run:
Integer Minimum Value: -2147483648
Integer Maximum Value: 2147483647
Float Minimum Value: 1.4E-45
Float Maximum Value: 3.4028235E38
Double Minimum Value: 4.9E-324
Double Maximum Value: 1.7976931348623157E308
BUILD SUCCESSFUL (total time: 4 seconds)

```

Hometasks:

Task#1

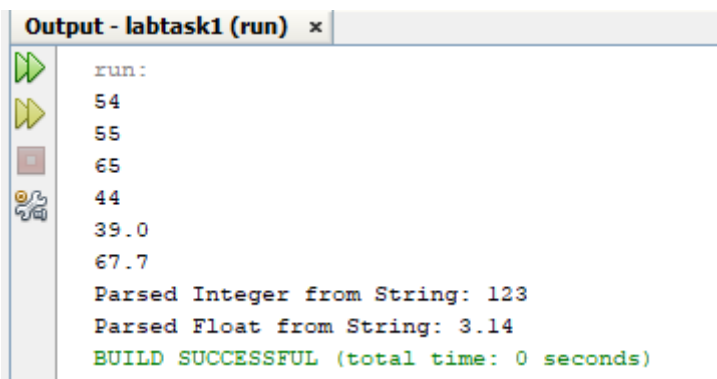
Write a JAVA program to perform Autoboxing and also implement different methods of Wrapper class.

```
public class Labtask1 {

    public static void main(String[] args) {
        int A=54;
        Integer B=A;
        System.out.println(B);
        short C= 55;
        Short D=C;
        System.out.println(D);
        byte E=65;
        Byte F=E;
        System.out.println(F);
        long G=44;
        Long H=G;
        System.out.println(H);
        float I=39;
        Float J=I;
        System.out.println(J);
        double K =67.7;
        Double L=K;
        System.out.println(L);

        System.out.println("Parsed Integer from String: " + Integer.parseInt("123")); // Parse from String
        System.out.println("Parsed Float from String: " + Float.parseFloat("3.14")); // Parse from String
    }
}
```

Output:



```
Output - labtask1 (run) x
run:
54
55
65
44
39.0
67.7
Parsed Integer from String: 123
Parsed Float from String: 3.14
BUILD SUCCESSFUL (total time: 0 seconds)
```

Task#2:

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

```
import java.util.Scanner;

public class Labtask2 {

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        System.out.print("Enter an integer: ");
        int number = input.nextInt();

        // Convert the integer to a string to iterate over each digit
        String numberStr = Integer.toString(Math.abs(number)); // Handle negative numbers
        Integer evenCount = 0; // Autoboxing
        Integer oddCount = 0; // Autoboxing

        for (char digitChar : numberStr.toCharArray()) {
            int digit = Character.getNumericValue(digitChar);
            if (digit % 2 == 0) {
                evenCount++; // Autoboxing
            } else {
                oddCount++; // Autoboxing
            }
        }

        // Unboxing to retrieve the values
        int evenDigits = evenCount; // Unboxing
        int oddDigits = oddCount;   // Unboxing

        System.out.println("Number of even digits: " + evenDigits);
        System.out.println("Number of odd digits: " + oddDigits);
    }
}
```

Output:

```
Output - Labtask#2 (run) x
run:
Enter an integer: 56789
Number of even digits: 2
Number of odd digits: 3
BUILD SUCCESSFUL (total time: 4 seconds)
```

Task#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.

```
public class Labtask3 {

    public static void main(String[] args) {

        Double number = -25.0;
        Double exponent = 2.0;

        // Use Math class methods and wrapper classes
        Double absValue = Math.abs(number); // Finds the absolute value
        Double sqrtValue = Math.sqrt(absValue); // Finds the square root of the absolute value
        Double powValue = Math.pow(absValue, exponent); // Finds the power

        // Display the results
        System.out.println("Number: " + number);
        System.out.println("Absolute Value: " + absValue);
        System.out.println("Square Root of Absolute Value: " + sqrtValue);
        System.out.println("Power (" + absValue + " ^ " + exponent + "): " + powValue);
    }
}
```

Output:

```
Output - labtask#2 (run) x
run:
Enter an integer: 56789
Number of even digits: 2
Number of odd digits: 3
BUILD SUCCESSFUL (total time: 4 seconds)
```

Task#4:

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


```

public class Labtask4 {

    public static void main(String[] args) {

        String input = "Hello, World!"; // Example input
        String result = reverseVowels(input);
        System.out.println("Reversed Vowels: " + result);
    }

    public static String reverseVowels(String s) {
        String vowels = "aeiouAEIOU";
        char[] chars = s.toCharArray();
        char[] reversedVowels = new char[countVowels(s)];
        int vowelIndex = 0;
        for (char c : chars) {
            if (vowels.indexOf(c) != -1) {
                reversedVowels[vowelIndex++] = c;
            }
        }
        for (int i = 0; i < vowelIndex / 2; i++) {
            char temp = reversedVowels[i];
            reversedVowels[i] = reversedVowels[vowelIndex - 1 - i];
            reversedVowels[vowelIndex - 1 - i] = temp;
        }
        vowelIndex = 0;
        for (int i = 0; i < chars.length; i++) {
            if (vowels.indexOf(chars[i]) != -1) {
                chars[i] = reversedVowels[vowelIndex++];
            }
        }

        return new String(chars);
    }

    private static int countVowels(String s) {
        int count = 0;
        for (char c : s.toCharArray()) {
            if ("aeiouAEIOU".indexOf(c) != -1) {
                count++;
            }
        }
        return count;
    }
}

```

Output:

```

run:
Reversed Vowels: Hollo, Werld!
BUILD SUCCESSFUL (total time: 0 seconds)

```

Task#5:

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

```
public class Labtask5 {

    public static void main(String[] args) {

        String sentence = "Find the longest word in this sentence";

        // Find and print the longest word
        String longestWord = findLongestWord(sentence);
        System.out.println("The longest word is: " + longestWord);

    }

    public static String findLongestWord(String sentence) {
        String[] words = sentence.split(" "); // Split the sentence into words
        String longestWord = ""; // Variable to store the longest word

        // Loop through each word to find the longest one
        for (String word : words) {
            if (word.length() > longestWord.length()) {
                longestWord = word; // Update if current word is longer
            }
        }

        return longestWord; // Return the longest word found
    }
}
```

Output:

```
Output - labtask5 (run) x
run:
The longest word is: sentence
BUILD SUCCESSFUL (total time: 0 seconds)
```