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

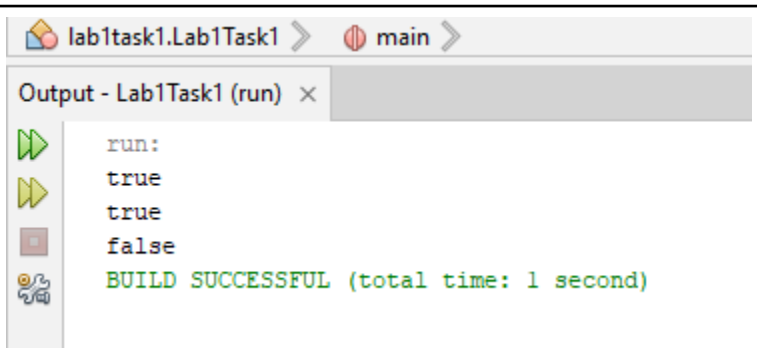
b) new keyword

also use intern method and show string immutability.

Source Code:

```
public class Lab1Task1 {
   public static void main(String[] args) {
   String str1 = "Java";
   String str2 = "Python";
   String str3 = "Java";
   String str4 = new String ("Java").intern();
   String str5 = new String ("Hello World");
   String str6 = new String ("Python");
   System.out.println(str1==str3);
   System.out.println(str1==str4);
   System.out.println(str2==str6);
   }
}
```

Output:



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

Source Code:

```
public class Lab1Task2 {
   public static void main(String[] args) {
   double no1 = 10.0;
   Double obj1 = Double.valueOf(no1);
   if(obj1 instanceof Double) {
      System.out.println("Object Of Double is created: " + obj1);
   } } }
```

Output:

```
Output - Lab1Task2 (run)
run:
Object of Double is created: 10.0
BUILD SUCCESSFUL (total time: 1 second)
```

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.

Source Code:

```
public class Lab1Task3 {
   public static void main(String[] args) {
   String str1 = "I";
   String str2 = " Love";
   String str3 = " Coding.";
   String str4 = " Just";
   String str5 = " Kidding!";
   String str6=str1+str2+str3+str4+str5;
   System.out.println("String: " + str6);
   System.out.println("UpperCase: " + str4.toUpperCase());
   System.out.println("SUBSTRING: " + str6.substring(8));
   } }
```

Output:

```
Output - Lab1Task3 (run)
run:
String: I Love Coding. Just Kidding!
UpperCase:  JUST
SUBSTRING: oding. Just Kidding!
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

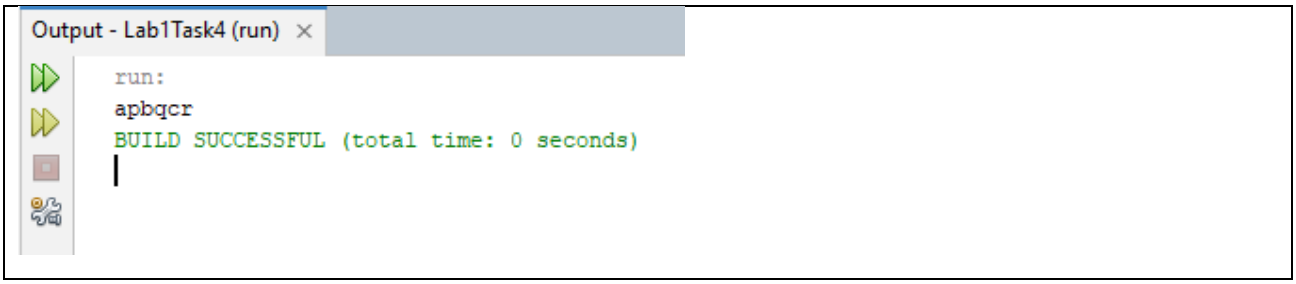
word2: p q r

merged: a p b q c r

Source Code:

```
package lab1task4;

public class Lab1Task4 {

   public static String mergeAlternately(String word1, String word2) {

      StringBuilder merged = new StringBuilder();

      int length1 = word1.length();

      int length2 = word2.length();

      int minLength = Math.min(length1, length2);

      for (int i = 0; i < minLength; i++) {

         merged.append(word1.charAt(i));

         merged.append(word2.charAt(i));}

      if (length1 > minLength) {

         merged.append(word1.substring(minLength));}

      if (length2 > minLength) {

         merged.append(word2.substring(minLength));}

      return merged.toString();}

   public static void main(String[] args) {

      String word1 = "abc";

      String word2 = "pqr";

      System.out.println(mergeAlternately(word1, word2)); // Output: "apbqcr" }}
```

Output:

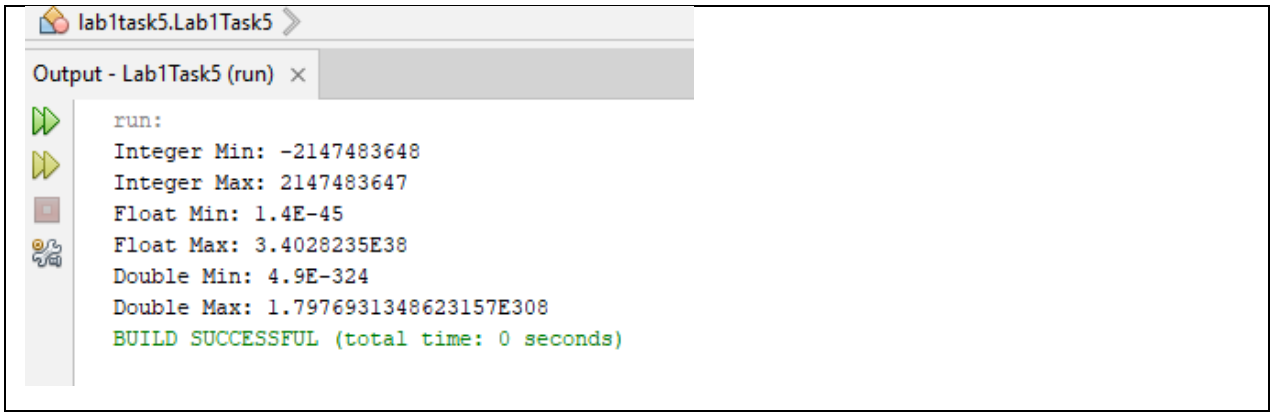


5. Write a Java program to find the minimum and maximum values of Integer, Float, and Double using the respective wrapper class constants.

Source Code:

```
package lab1task5;

public class Lab1Task5 {

    public static void main(String[] args) {

        System.out.println("Integer Min: " + Integer.MIN_VALUE);

        System.out.println("Integer Max: " + Integer.MAX_VALUE);

        System.out.println("Float Min: " + Float.MIN_VALUE);

        System.out.println("Float Max: " + Float.MAX_VALUE);

        System.out.println("Double Min: " + Double.MIN_VALUE);

        System.out.println("Double Max: " + Double.MAX_VALUE);}}
```

Output:



Home Tasks

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

Source Code:

```
package lab1hometask1;
public class Lab1HomeTask1 {
    public static void main(String[] args) {
        int primitiveInt = 10;
        Integer wrappedInt = primitiveInt; // autoboxing
        System.out.println("Autoboxed Integer: " + wrappedInt);
        System.out.println("Max Integer Value: " + Integer.MAX_VALUE);
        System.out.println("Min Integer Value: " + Integer.MIN_VALUE);
        System.out.println("Parsed Integer from String: " + Integer.valueOf("123"));
        System.out.println("Integer to String: " + Integer.toString(wrappedInt));
        System.out.println("Integer Comparison: " + wrappedInt.compareTo(15));}}
```

Output:

```
lab1hometask1.Lab1HomeTask1 > main >
Output - Lab1HomeTask1 (run)
run:
Autoboxed Integer: 10
Max Integer Value: 2147483647
Min Integer Value: -2147483648
Parsed Integer from String: 123
Integer to String: 10
Integer Comparison: -1
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.

Source Code:

```
package lab1hometask2;
public class Lab1HomeTask2 {
   public static void main(String[] args) {
      int number = 1234567890;
      countEvenOddDigits(number);    }
   public static void countEvenOddDigits(int num) {
      Integer numObj = num;
      int evenCount = 0, oddCount = 0;
      while (numObj > 0) {
         int digit = numObj % 10;
         if (digit % 2 == 0) {
            evenCount++;
         } else {
            oddCount++;}
         numObj /= 10; }
      System.out.println("Even Digits Count: " + evenCount);
      System.out.println("Odd Digits Count: " + oddCount); }}
```

Output:

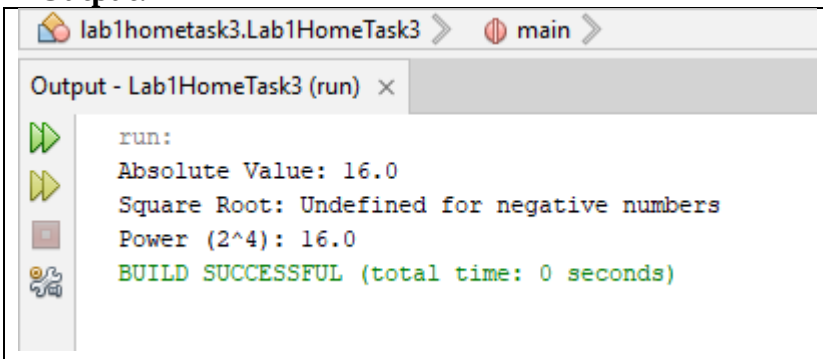
```
lab1hometask3.Lab1HomeTask3 > main >
Output - Lab1HomeTask2 (run)
run:
Even Digits Count: 5
Odd Digits Count: 5
BUILD SUCCESSFUL (total time: 0 seconds)
```

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**.**

Source Code:

```
package lab1hometask3;

public class Lab1HomeTask3 {

    public static void main(String[] args) {

        Double number = -16.0; // Autoboxing

        System.out.println("Absolute Value: " + Math.abs(number));

        if (number < 0) {

            System.out.println("Square Root: Undefined for negative numbers");} else {

            System.out.println("Square Root: " + Math.sqrt(number));}

        System.out.println("Power (2^4): " + Math.pow(2, 4));}}
```

Output:



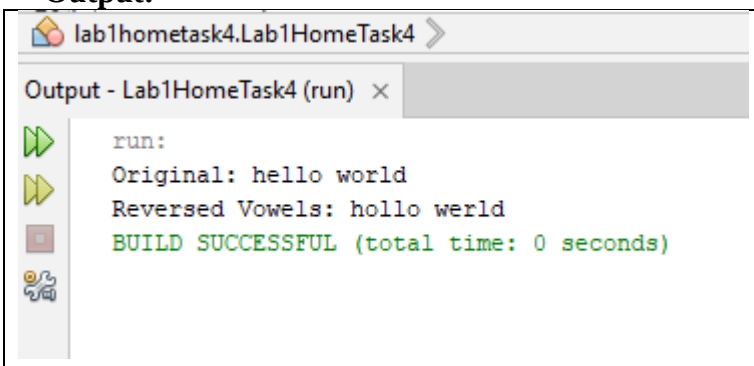
4. Write a Java program to **reverse only the vowels** in a string.

Source Code:

```
package lab1hometask4;

import java.util.ArrayList;

public class Lab1HomeTask4 {

    public static void main(String[] args) {

        String input = "hello world";

        System.out.println("Original: " + input);

        System.out.println("Reversed Vowels: " + reverseVowels(input));}

    public static String reverseVowels(String s) {

        ArrayList<Character> vowels = new ArrayList<>();

        for (char c : s.toCharArray()) {
```

```
            if ("AEIOUaeiou".indexOf(c) != -1) {
                vowels.add(c);}}
        StringBuilder result = new StringBuilder();
        int vowelIndex = vowels.size() - 1;
        for (char c : s.toCharArray()) {
            result.append(isVowel(c) ? vowels.get(vowelIndex--) : c);}
        return result.toString();}
    private static boolean isVowel(char c) {
        return "AEIOUaeiou".indexOf(c) != -1;}}
```

Output:

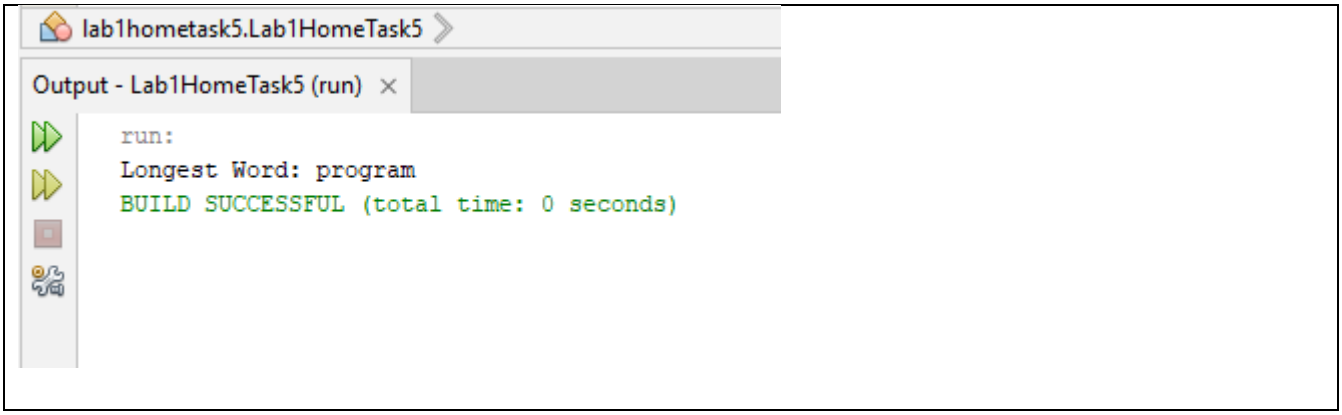


5. Write a Java program to **find the longest word** in a sentence.

Source Code:

```
package lab1hometask5;
public class Lab1HomeTask5 {
    public static void main(String[] args) {
        String sentence = "Write a Java program to find the longest word";
        String longest = findLongestWord(sentence);
        System.out.println("Longest Word: " + longest);}
    public static String findLongestWord(String sentence) {
        String[] words = sentence.split(" ");
        String longestWord = "";
        for (String word : words) {
            if (word.length() > longestWord.length()) {
                longestWord = word;}}
        return longestWord;}}
```

Output:


lab1hometask5.Lab1HomeTask5
Output - Lab1HomeTask5 (run)
run:
Longest Word: program
BUILD SUCCESSFUL (total time: 0 seconds)