**INTRODUCTION TO JAVA ASSIGNMENT**

1. **Given two arrays: array1={5,10,15,20,25,30}, array2={50,60,70,80,90,100}. Write a Java code to merge two arrays and display the result.**

import java.util.Arrays;

public class Main

{

public static void main(String args[]) {

int a1[]=new int[]{5,10,15,20,25,30};

int a2[]=new int[]{50,60,70,80,90,100};

int l1=a1.length;

int l2=a2.length;

int res[]=new int[l1+l2];

System.arraycopy(a1,0,res,0,l1);

System.arraycopy(a2,0,res,l1,l2);

for(int i=0;i<res.length;i++)

{

System.out.println(res[i]+" ");

}

}

}

1. **Why String objects are immutable?**

The String objects are cached in the String pool, and it makes the String immutable. In the String constant pool, a String object is likely to have one or many references. If several references point to same String without even knowing it, it would be bad if one of the references modified that String value. That’s why String objects are immutable.

1. **How to create an immutable class?**

To create an immutable class in Java,

* Declare the class as final so it can’t be extended.
* Make all fields private so that direct access is not allowed.
* Don’t provide setter methods for variables.
* Make all mutable fields final so that its value can be assigned only once.
* Initialize all the fields via a [constructor](https://www.journaldev.com/18899/constructor-in-java) performing deep copy.
* Perform [cloning](https://www.journaldev.com/60/java-clone-object-cloning-java) of objects in the getter methods to return a copy rather than returning the actual object reference.

1. **What is string constant pool?**

A string constant pool is a separate place in the heap memory where the values of all the strings which are defined in the program are stored. When we declare a string, an object of type String is created in the stack, while an instance with the value of the string is created in the heap. The distinct values are stored in the String pool. The main advantage of the string pool in Java is to reduce memory usage.

1. **What code is written by the compiler if you concatenate any string by + (string concatenation operator)?**

The Java compiler transforms above code to : String s=(new StringBuilder()). append(String name).

1. **What is the difference between StringBuffer and StringBuilder class?**

StringBuilder is same as the StringBuffer , that is it stores the object in heap and it can also be modified . The main difference between the StringBuffer and StringBuilder is that StringBuilder is not thread safe while StringBuffer is thread safe. StringBuilder is faster than StringBuilder as it is not thread safe .

1. **How to prove String is immutable programatically?**

public class Main

{

public static void main(String[] args) {

String str1 = "Java";

String str2 = "Java";

System.out.println("Before Modification");

if (str1 == str2) {

System.out.println("Both pointing to the same reference");

} else {

System.out.println("Both are pointing to different reference");

}

str1 = str1 + "Hello";

str2 = str2 + "Hello";

System.out.println("After Modification");

if (str1 == str2) {

System.out.println("Both pointing to the same reference");

} else {

System.out.println("Both are pointing to different reference");

}

}

}

**O/P:**

Before Modification

Both pointing to the same reference

After Modification

Both are pointing to different reference

1. **How to swap two Strings without using a third variable?**

public class Main

{

public static void main(String[] args) {

String a="Welcome";

String b="Aspire";

a=a+b;

b=a.substring(0,a.length()-b.length());

a=a.substring(b.length());

System.out.println("After swap:a-"+a+" b-"+b);

}

}

**O/P:**

After swap : a-Aspire b-Welcome