JAVA FOR SOFTWARE DEVELOPMENT
PRACTICAL NO : 01

Aim: Understanding Null and Empty String

Code:

```
package string_manipulation;
public class Nullemptystring {
	public static void main(String[] args) {
		String str1 = null;
		String str2 = "";
		String str3 = "Hello World!";

		System.out.println("Handling str1: ");
		checkString(str1);

		System.out.println("Handling str2");
		checkString(str2);

		System.out.println("Handling str3");
		checkString(str3);

	}
	private static void checkString(String str) {
		if (str == null) {
			System.out.println("The string is null");
		}
		else if (str.isEmpty())
		{
			System.out.println("The string is empty");
		}
		else {
			System.out.println("The string is :" + str);
		}
	}
}
```

Output:

```
Handling str1:
The string is null
Handling str2
The string is empty
Handling str3
The string is :Hello World!
```

Aim: Write a program that checks for substring presence.

Code:

```
package Assignment1;
public class Substring {
	public static void main(String[] args)
	{
		String str1 = "Hello World";
		String substring = "World";

		if (str1.contains(substring))
		{
			System.out.println("The second string is the substring of the
first.");
		}
		else
		{
			System.out.println("The second string is NOT a substring of the
first.");
		}
	}
}
```

Output:

```
Console ×
<terminated> Substring [Java Application] C:\Users\Ramkrishna\.p2\pool\plugins\org.eclipse.ju
The second string is the substring of the first.
```

Aim: Write a program that compares two strings using

a. equals
b. ==
c. compareTo
d. equalsIgnoreCase
e. Object.equals()

a) Code:

```java
package string_manipulation;

public class StringComparison {

    public static void main(String[] args) {
        String str1 = "Hello";
        String str2 = "Hello";
        String str3 = "Another String";
        System.out.println(str1.equals(str2));
        System.out.println(str1.equals(str3));

    }

}
```

Output:

```
true
false
```

Aim: Write a program that takes user input for multiple strings and append them using String Builder.

Code:

```
package Assignment1;
import java.util.Scanner;
public class StringAppender {
	public static void main (String [] args) {
		Scanner scanner = new Scanner (System.in);
		StringBuilder sb = new StringBuilder();
		System.out.println("Enter strings to concatenate (type 'exit' to stop):");
		while (true) {
			String input = scanner.nextLine();
			if (input.equalsIgnoreCase("exit")) {
				break;
			}
			sb.append(input).append(" ");
		}
		System.out.println("Concatenated result: " + sb.toString ().trim());
		scanner.close();
	}
}
```

Output:

```
Enter strings to concatenate (type 'exit' to stop) :

Samruddhi
Avhad
exit
Concatenated result: Samruddhi Avhad
```

Aim: Write a program to insert a substring into a string at a specific position using StringBuilder.

Code:

```
package Assignment1;
public class Insertion {
	public static void main (String [] args) {
		String originalString = "Hello World!";
		String substring = "Java ";
		int position = 6;
		StringBuilder sb = new StringBuilder (originalString);
		sb.insert(position,  substring);
		System.out.println("Original String: "+originalString);
		System.out.println("After Insertion: "+sb.toString());
	}
}
```

Output:

```
<terminated> Insertion [Java Application] C:\Program Files\Java\jdk-2
Original String: Hello World!
After Insertion: Hello Java World!
```

Aim: Program to encode characters to their Unicode representations and decode them back.

Code:

```
package Assignment1;
import java.nio.charset.StandardCharsets;
import java.util.Scanner;
public class Unicode {
	public static void main(String[] args)
	{
		Scanner sc = new Scanner (System.in);
		System.out.println("Enter a string: ");
		String input = sc.nextLine();
		System.out.println("\n Unicode code points for the input string:");
		for (int i=0; i<input.length(); i++)
		{
			int codepoint = input.codePointAt(i);
			System.out.printf("Character: '%c' -> Unicode: \\u%04X\n",
input.charAt(i), codepoint);
		}
		byte[] utf8Bytes = input.getBytes(StandardCharsets.UTF_8);
		System.out.println("\n String in UTF-8 byte encoding:");
		for (byte b : utf8Bytes)
		{
			System.out.printf("%02X", b);
		}
		String decodedString = new String (utf8Bytes,
StandardCharsets.UTF_8);
		System.out.println("\n\n Decoded string from UTF-8 bytes:
"+decodedString);
		sc.close();
	}
}
```

Output:

```
Console ×
<terminated> Unicode [Java Application] C:\Users\Ramkrishna\.p2\pool\plugins\org.ecli
Enter a string:
abcd

 Unicode code points for the input string:
Character: 'a' -> Unicode: \u0061
Character: 'b' -> Unicode: \u0062
Character: 'c' -> Unicode: \u0063
Character: 'd' -> Unicode: \u0064

 String in UTF-8 byte encoding:
61626364

 Decoded string from UTF-8 bytes: abcd
```

Aim: Write a program to split a paragraph into individual sentences

Code:

```
package Assignment1;
public class SentenceSplitter {
	public static void main(String args[]) {
String paragraph = "This is the first sentence. Is this the second sentence? Yes! It is the third one.";
		String[]sentences = paragraph.split("[.?!]");
		for(String sentence : sentences) {
			System.out.println(sentence.trim());
		}
	}
}
```

Output:

```
This is the first sentence
Is this the second sentence
Yes
It is the third one
```

Aim: Write a program to convert a date object to a string in a specific format.

Code:

```
package Assignment1;
import java.text.SimpleDateFormat;
import java.util.Date;
public class DateToStringEx {
	public static void main (String args[]) {
		Date currentDate = new Date();
		SimpleDateFormat formatter = new SimpleDateFormat
("dd/MM/yyyy");
		String formattedDate = formatter.format(currentDate);
		System.out.println("Formatter Date: " + formattedDate);
	}
}
```

Output:

```
Formatter Date: 07/10/2024
```

Aim: Write a program to remove null values from an array of strings

Code:

```
package Assignment1;
import java.util.ArrayList;
public class RemoveNullValues {
	public static void main (String args[]) {
	String[] originalArray = {"Apple", null, "Banana", "Orange", null,
"Grapes"};
	ArrayList<String>nonNullList = new ArrayList<>();
	for (String element : originalArray) {
		if (element != null) {
			nonNullList.add(element);
		}
	}
	String[] nonNullArray = nonNullList.toArray(new String [0]);
	System.out.println("Array after removing null values:");
	for (String element : nonNullArray) {
		System.out.println(element);
	}
}
}
```

Output:

```
Array after removing null values:
Apple
Banana
Orange
Grapes
```

Aim: Write a program to count the number of vowels in a string using Character class

Code:

```
package Assignment1;
public class VowelCounter {
	public static void main(String[] args) {
		String inputString = "Samruddhi Avhad";
		int vowelCount = 0;
		inputString = inputString.toLowerCase();
		for (int i = 0; i< inputString.length(); i++) {
			char currentChar = inputString.charAt(i);
			if (Character.isLetter(currentChar) && isVowel(currentChar)) {
				vowelCount ++;
			}
		}
		System.out.println("Number of vowels in the string:" + vowelCount);
	}
	private static boolean isVowel (char c)
	{
		return c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u';
	}
}
```

Output:

```
Number of vowels in the string:5
```

Aim: Write a program to find all occurrences of a pattern in a string using Pattern and Matcher.

Code:

```
package Assignment1;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
public class FindPatternOccurences {
	public static void main(String[] args) {
		String patternString = "ab";
		String inputString = "abcabcababcdab";
		Pattern pattern = Pattern.compile(patternString);
		Matcher matcher = pattern.matcher(inputString);
		System.out.println("Pattern found at the following positions: ");
		while (matcher.find()) {
			System.out.println("Start index: "+ matcher.start() + " , End
index: " + matcher.end());
		}
	}
}
```

Output:

```
Pattern found at the following positions:
Start index: 0 , End index: 2
Start index: 3 , End index: 5
Start index: 6 , End index: 8
Start index: 8 , End index: 10
Start index: 12 , End index: 14
```

Aim: Write a program to check whether a given string is a palindrome or not by using:

A.StringBuffer Class

B.String Class

A. Code:

```
package Assignment1;
public class Palindrome {
        public static void main(String[] args)
        {
                String inputString = "SAMRUDDHI";
            StringBuffer stringBuffer = new StringBuffer(inputString);
            StringBuffer reversedString = stringBuffer.reverse();
            if(inputString.equals(reversedString.toString()))
            {
                System.out.println(inputString + " is palindrome");
            }
            else
            {
                System.out.println(inputString + " is not palindrome");
            }
        }
}
```

Output:

```
<terminated> Palindrome [Java Application] C:\Progr
SAMRUDDHI is not palindrome
```

B. Code:

```
package Assignment1;
public class PalindromeB
{
	public static void main(String args[])
	{
		String inputString = "MADAM";
		String reversedString = "";

		for (int i = inputString.length() - 1; i>=0; i--)
		{
			reversedString += inputString.charAt(i);
		}
		if (inputString.equals(reversedString))
		{
			System.out.println(inputString + " is palindrome.");
		}
		else
		{
			System.out.println(inputString + " is not palindrome.") ;
		}
	}
}
```

Output:

```
MADAM is palindrome.
```

PRACTICAL NO : 2

Aim: Write a Java program to create List containing list of items and use ListIterator interface to print items present in the list. Also print the list in reverse/ backward direction.

Code:

```
package Assignment2;
import java.util.ArrayList;
import java.util.List;
import java.util.ListIterator;
public class Iterator {
	public static void main(String[] args)
	{
		List<Integer> l1 = new ArrayList<>();
		l1.add(1);
		l1.add(2);
		l1.add(3);
		ListIterator<Integer> L = l1.listIterator();
		System.out.println("Traversing in Forward direction");
		while (L.hasNext())
		{
			System.out.println(L.next());
		}
		System.out.println("Traversing in reverse direction");
		while(L.hasPrevious())
		{
			System.out.println(L.previous());
		}
	}
}
```

Output:

```
Traversing in Forward direction
1
2
3
```

```
Traversing in reverse direction
3
2
1
```

Aim: Write a Java program to create a Set containing list of items of type String and print the items in the list using the Iterator interface. Also print the list in reverse/ backward direction.

Code:

```
package Assignment2;
import java.util.Set;
import java.util.TreeSet;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Iterator;
import java.util.List;
public class ListInterface {
	public static void main(String[] args)
	{
		Set<String> s1 = new TreeSet<>();
		s1.add("C++");
		s1.add("Java");
		s1.add("C#");
		Iterator<String> itr = s1.iterator();
		System.out.println("SAMRUDDHI AVHAD - 07");
		System.out.println("Traverse in Forward Direction");
		while(itr.hasNext())
		{
			System.out.println(itr.next());
		}
		System.out.println("Traverse in Reverse Direction");
		List<String> l1=new ArrayList<String>(s1);
		Collections.reverse(l1);
		for(String language : l1)
		{
			System.out.println(language);
		}
	}
}
```

Output:

```
<terminated> ListInterface [Java Application] C:\Program File
SAMRUDDHI AVHAD - 07
Traverse in Forward Direction
C#
C++
Java
Traverse in Reverse Direction
Java
C++
C#
```

Aim: Write a Java program using Set interface containing list of items and perform the following operations:

a. Add items in the set.
b. Insert items of one set into another set.
c. Remove items from the set
d. Search the specified item in the set

Code:

```
package Assignment2;
import java.util.Set;
import java.util.TreeSet;
public class Set1 {
	public static void main(String[] args)
	{
		Set<Integer> id= new TreeSet<>();
		id.add(1);
		id.add(9);
		id.add(3);
		System.out.println("SAMRUDDHI AVHAD - 07");
		System.out.println("Items in 1st Set: "+id);
		Set<Integer> id2 = new TreeSet<>();
		id2.add(12);
		id2.add(23);
		System.out.println("Items in 2nd Set: "+id2);
		System.out.println("Inserting items of 1st set into another: ");
		id.addAll(id2);
		System.out.println(id);
		if(id.contains(9))
		{
			id.remove(9);
		}
		System.out.println("After deletion of item in set: ");
		System.out.println(id);
	}
}
```

Output:

```
<terminated> Set1 [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
SAMRUDDHI AVHAD - 07
Items in 1st Set: [1, 3, 9]
Items in 2nd Set: [12, 23]
Inserting items of 1st set into another:
[1, 3, 9, 12, 23]
After deletion of item in set:
[1, 3, 12, 23]
```

Aim: Write a Java program using Map interface containing list of items having keys and associated values and perform the following operations:

a.Add items in the map.
b.Remove items from the map
c.Search specific key from the map
d.Get value of the specified key
e.Insert map elements of one map into another map.
f. Print all keys and values of the map.

Code:

```
package Assignment2;
import java.util.HashMap;
import java.util.Map;
public class MapEx {
	public static void main(String[] args)
	{
		Map<Integer,String> map=new HashMap<Integer, String>();
		map.put(1, "Sandeep M");
		map.put(2, "Afsa S");
		map.put(3, "Sandhya D");
		map.forEach((k,v)-> System.out.println(k+" "+v));
		if(map.containsKey(2))
		{
			map.remove(2);
		}
		System.out.println("SAMRUDDHI AVHAD - 07");
		System.out.println("After removing element with key as 2:");
		map.forEach((k,v) -> System.out.println(k+" "+v));
		String val = map.get(1);
		System.out.println("Name of Student with Roll no 1 is:  "+val);
		Map<Integer,String> map2=new HashMap<Integer, String>();
		map2.put(4, "Janki M");
		map2.put(5, "Seema E");
		map.putAll(map2);
		System.out.println("After adding one map element into another");
```

```
            map.forEach((k,v)-> System.out.println(k+" "+v));
            for (Integer key : map.keySet())
            {
                System.out.println("key:" +key);
            }
            for(String value: map.values())
            {
                System.out.println("value: "+value);
            }
        }
}
```

Output:

```
<terminated> MapEx [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (21-
SAMRUDDHI AVHAD - 07
1 Sandeep M
2 Afsa S
3 Sandhya D
After removing element with key as 2:
1 Sandeep M
3 Sandhya D
Name of Student with Roll no 1 is:  Sandeep M
After adding one map element into another
1 Sandeep M
3 Sandhya D
4 Janki M
5 Seema E
key:1
key:3
key:4
key:5
value: Sandeep M
value: Sandhya D
value: Janki M
value: Seema E
```

Aim: WAP using Lambda Expression with multiple parameters to print addition of two numbers.

Code:

```
package Assignment2;
	interface Arithmetic
	{
		public int add (int a, int b );
	}
		public class LambdaExpDemo
		{
			public static void main(String [] args)
			{
				Arithmetic arth = (a,b)-> a+b;
				int sum;
				sum = arth.add(5, 6);
				System.out.println("SAMRUDDHI AVHAD - 07");
				System.out.println("The sum of numbers are: "+ sum);

			}
	}
```

Output:

```
<terminated> LambdaExpDemo [Java Application] C:
SAMRUDDHI AVHAD - 07
The sum of numbers are: 11
```

Aim: WAP using Lambda Expression to calculate following:

a) Convert Fahrenheit to Celsius

b) Convert Kilometers to Meters

Code:

```
package Assignment2;
interface TemperatureInterface
{
	public void fahrenheitTocelcius (double fahrenheit);
}
interface DistanceInterface
{
	public void kilometersTometers(double kilometers);
}
public class LambdaExp
{
	public static void main(String [] args)
	{
		System.out.println("SAMRUDDHI AVHAD -07");
		TemperatureInterface temp = (fahrenheit)->
System.out.println((fahrenheit - 32)* (5.0/9.0));
		temp.fahrenheitTocelcius(97.6);
		DistanceInterface dist = (kilometers) ->
System.out.println(kilometers* 1000);
		dist.kilometersTometers(6.3);
	}
}
```

Output:

```
<terminated> LambdaExp [Java Applic
SAMRUDDHI AVHAD -07
36.44444444444444
6300.0
```

Aim: Write a Spring application that demonstrates Dependency Injection using the setter method, implemented through XML-based configuration or annotation-based configuration.

Code:

Account.java (POJO class)

```
package com.setter;
public class Account
{
	private int acc;
	private String accName;
	public int getAcc() {
		return acc;
	}
	public void setAcc(int acc) {
		this.acc = acc;
	}
	public String getAccName() {
		return accName;
	}
	public void setAccName(String accName) {
		this.accName = accName;
	}

}
```

config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans
xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xsi:schemaLocation="
http://www.springframework.org/schema/beans
```

```
classpath:/org/springframework/beans/factory/xml/spring-beans-3.0.xsd
http://www.springframework.org/schema/context
classpath:/org/springframework/context/config/spring-context-3.0.xsd">
<bean id="accBean" class="com.setter.Account">
	<property name="acc" value="05"></property>
	<property name="accName" value="HDFC"></property>
</bean>
</beans>
```

App.java

```
package com.setter;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class App {
	public static void main(String[] args)
	{
		ApplicationContext context = new
ClassPathXmlApplicationContext("config.xml");

		Account a = (Account) context.getBean("accBean");
		System.out.println("SAMRUDDHI AVHAD - 07");
		System.out.println("Account No: "+a.getAcc());
		System.out.println("Account Name: "+a.getAccName());
	}
}
```

Output:

```
<terminated> App [Java Application] C:\Program
log4j:WARN No appenders co
log4j:WARN Please initiali
SAMRUDDHI AVHAD - 07
Account No: 5
Account Name: HDFC
```

Aim: Write a program to demonstrate dependency injection via Constructor (Employee Application)

Address.java

```
package com.constructorinject;
public class Address
{
	private String city;
	private String state;
	private String country;

	public Address (String city, String state, String country)
	{
		super();
		this.city = city;
		this.state= state;
		this.country = country;
	}
	public String toString() {
		return city +" "+ state +" " + country;
	}
}
```

Employee.java

```
package com.constructorinject;
public class Employee
{
	private int id;
	private String name;
	private Address address;
public Employee()
{
	System.out.println("def cons");
}
public Employee (int id, String name, Address address)
```

```
{
        super();
        this.id=id;
        this.name=name;
        this.address=address;
}
void show()
{
        System.out.println("SAMRUDDHI AVHAD - 07");
        System.out.println("The employee details are:");
        System.out.println("Employee id:" +id + " "+"Employee name:" +name);
        System.out.println("Address: "+address.toString());
}
}
```

config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans
xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xmlns:aop="http://www.springframework.org/schema/aop"
xsi:schemaLocation="
http://www.springframework.org/schema/beans
classpath:/org/springframework/beans/factory/xml/spring-beans-3.0.xsd
http://www.springframework.org/schema/context
classpath:/org/springframework/context/config/spring-context-3.0.xsd">
<bean id = "a1" class="com.constructorinject.Address">
<constructor-arg value ="Mumbai" index ="0" ></constructor-arg>
<constructor-arg value ="Maharashtra" index ="1" ></constructor-arg>
<constructor-arg value ="India" index ="2" ></constructor-arg>
</bean>
<bean id ="e" class="com.constructorinject.Employee">
<constructor-arg value ="101" type="int"></constructor-arg>
```

```
<constructor-arg value ="Shrikant"></constructor-arg>
<constructor-arg>
<ref bean = "a1"/>
</constructor-arg>
</bean>
</beans>
```

App.java

```
package com.constructorinject;
import org.springframework.core.io.ClassPathResource;
import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;
public class App {
	public static void main(String[] args)
	{
		//ApplicationContext context = new
ClassPathXmlApplicationContext("config.xml");
		ClassPathResource r = new ClassPathResource("config.xml");
		BeanFactory factory = new XmlBeanFactory(r);
		Employee s = (Employee) factory.getBean("e");
		s.show();
	}
}
```

Output:

```
log4j:WARN No appenders could be found f
log4j:WARN Please initialize the log4j s
SAMRUDDHI AVHAD - 07
The employee details are:
Employee id:101 Employee name:Shrikant
Address: Mumbai Maharashtra India
```

Aim: Write a Spring application that demonstrates Dependency Injection using the constructor, implemented through annotation-based configuration.