List of Experiments:

1. # To implement bounded types with generics

A. 1. Create a class shape with method Area() create circle and Square which extends Class Shape. Create a generic class BoundedShape that extends shape. And implement the generics and use area function accordingly:

```
package something;
import java.util.*;
abstract class shape
{
Scanner sc = new Scanner(System.in);
abstract void Area();
}

class circle extends shape
{
{System.out.println("Enter Radius of Circle: ");}
int r = sc.nextInt();
public void Area()
{
System.out.println("AREA OF CIRCLE: "+3.14*r*r);
}
}

class square extends shape
{
{System.out.println("Enter Side of Square: ");}
int s = sc.nextInt();
public void Area()
{
System.out.println("AREA OF SQUARE: "+s*s);
}
}

class BoundedShape<T extends shape>
{
private T obje;
public BoundedShape(T obje)
{

ATHARVA KALE PRACTICAL 1 ROLL NO: 27

this.obje= obje;
}
public void gotcha()
{
this.obje.Area();
```

```
}
public static void main(String args[])
{
BoundedShape<circle> s = new BoundedShape<circle>(new circle());
s.gotcha();

BoundedShape<square> t = new BoundedShape<square>(new square());
t.gotcha();
}
}
```

B. Create an Interface shape with method Area() create Circle and Square which implements Class Shape. Create a generic class BoundedShape that extends shape. And implement the generics and use area function accordingly:

```
package something;
import java.util.*;
interface A
{
Scanner sc = new Scanner(System.in);
public void Area();
}

class circle implements A
{
{System.out.println("Enter Radius of Circle: ");}
int r = sc.nextInt();
public void Area()
{
System.out.println("AREA OF CIRCLE: "+3.14*r*r);
}
}

class square implements A
{
{System.out.println("Enter Side of Square: ");}
int s = sc.nextInt();
public void Area()
{
System.out.println("AREA OF SQUARE: "+s*s);
}
}

class triangle
{
```

```
public void Area()
{
System.out.println("TRIANGLE");
}
}

class BoundedShape<T extends A>
{
private T obje;
public BoundedShape(T obje)
{
this.obje= obje;
}
public void gotcha()
{
this.obje.Area();
}
public static void main(String args[])
{
BoundedShape<circle> s = new BoundedShape<circle>(new circle());
s.gotcha();

BoundedShape<square> t = new BoundedShape<square>(new square());
t.gotcha();
}
}
```

2. To implement List Interface, Set Interface, Map Interface, Lambda Expression.

    A. Create an ArrayList of type Integer, add element into it traverse the arraylist and print the elements

```
package something;
import java.util.*;
class BoundedShape
{
static Scanner sc = new Scanner(System.in);
static void al()
{
ArrayList<Integer> nums = new ArrayList<Integer>();
System.out.println("ENTER NO OF ELEMENTS: ");
int n=sc.nextInt();
System.out.println("Enter the "+n+" elements");
for(int i=0;i<n;i++)
{
```

```
int inp = sc.nextInt();
nums.add(inp);
}
Iterator<Integer> it = nums.iterator();
System.out.println("ARRAY IS\n");
while(it.hasNext()) {
System.out.println(it.next());
}
System.out.println(' ');
}
public static void main(String args[])
{
System.out.println(' ');
al();
}
}
```

B. Create a LinkedList of type String add 5 elements and traverse the list and from both sides

```
package something;
import java.util.*;
class BoundedShape
{
static Scanner sc = new Scanner(System.in);
static void ll()
{
LinkedList<String> li = new LinkedList<String>();
System.out.println("ENTER NO OF ELEMENTS: ");
int n=sc.nextInt();
System.out.println("Enter the "+n+" elements");
for(int i=0;i<n;i++)
{
String inp = sc.next();
li.add(inp);
}
System.out.println("LINK LIST IS\n");
for(String i:li)
{
System.out.println(i);
}
System.out.println(' ');
}

public static void main(String args[])
{
```

```
System.out.println(' ');
ll();
}
}
```

C. Create an employee class (id, name, salary) create an Arralist of type employee, add 5 employees, traverse the ArrayList and print the elements, Remove one element and print the list.

```
package something;
import java.util.*;
class employee {
static class emp{
int id;
String name;
Double sal;
emp(int id,String name,Double sal)
{
this.id=id;
this.name= name;
this.sal = sal;
}
}
static Scanner sc= new Scanner(System.in);
static ArrayList<emp> data= new ArrayList<emp>();
static void add()
{
System.out.println("Enter Details Of Employee");
System.out.print("ID : ");
int id1=sc.nextInt();
System.out.print("NAME : ");
String name1=sc.next();
System.out.print("SALARY : ");
Double salary1=sc.nextDouble();
data.add(new emp(id1, name1, salary1));
}
static void pri()
{
for(int j=0;j<data.size();j++)
{
emp e= data.get(j);
System.out.println(e.id);
System.out.println(e.name);
System.out.println(e.sal);
}
}
```

```
static void remov()
{

System.out.println("Enter the index");
int inde = sc.nextInt();
data.remove(inde);
System.out.println("Arraylist after removing data!");
pri();
}
public static void main(String args[])
{
while(true)
{
System.out.println(" ");
System.out.println("Enter the no: \n 1)ADD \n

2)Print\n 3)Remove");

int ip=sc.nextInt();
if(ip==1)
{
add();
}
else if(ip==2)
{
pri();
}
else if(ip==3)
{
remov();
}
else
{
break;
}}}}
```

D. 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.

```
package something;
import java.util.*;
```

```
public class sets {
public static void main(String args[]) {
Set<String> s= new HashSet<String>();
s.add("IRON MAN");
s.add("DOCTOR STRANGE");
s.add("CAPTAIN AMERICA");
s.add("THOR");
s.add("HULK");
s.add("VISION");
System.out.println("Priting values of SET A: \n");
Iterator<String> it=s.iterator();
while(it.hasNext())
{
System.out.println(it.next());



}
System.out.println("\n");
Set<String> s2=new HashSet<String>(s);
System.out.println("Entering values from Set A to Set B and

printing Set B:\n");

Iterator<String> it2=s2.iterator();
while(it2.hasNext())
{
System.out.println(it2.next());
}
System.out.println("\n");
System.out.println("Removing items VISION AND IRON MAN from

Set B\n ");

s2.remove("VISION");
s2.remove("IRON MAN");
System.out.println("Items removed from Set B\n");
System.out.println("Serching for removed item IRON MAN\n");
System.out.println("Does Set B contains IRON MAN?

"+s2.contains("IRON MAN"));
}
}
```

E. Create a class Customer(Account_no Integer, Name Sting), Create a HashMap of type Customer put elements, print elements, check if element with account number 101 is present or not? What is the value for Customer 101.

```
package mylab;

import java.util.*;

public class Customer
{
	int Account_No;
	String Name;
	public Customer(int Account_No,String Name)
	{
		this.Account_No=Account_No;
		this.Name=Name;
	}
	public String getname()
	{
		return this.Name;
	}
	public int getacc()
	{
		return this.Account_No;
	}
public static void main(String args[])
{
	HashMap<Customer,Integer> cus = new HashMap<>();
	Customer c1 = new Customer(101,"IRON MAN");
	Customer c2=new Customer(102,"DOCTOR STRANGE");
	Customer c3 = new Customer(103,"THOR");
	cus.put(c1, c1.getacc());
	cus.put(c2, c2.getacc());
	cus.put(c3, c3.getacc());
	System.out.println("PRINTING ELEMENTS FROM HASHMAP:\n");
	for (Customer i : cus.keySet())
	{
		System.out.println("ACCOUNT NO:" + cus.get(i) + " NAME:" + i.getname());

	}
	System.out.println("\nWhether there is a customer with ACCOUNT NO: 101?
"+cus.containsValue(101));
	for (Customer i : cus.keySet())
	{
		if(cus.get(i)==101)
		{
			System.out.println("ACCOUNT NO:" + cus.get(i) + " NAME:" +
i.getname());
		}
		break;

	}
}
```

```
}
```

F. Write a Java program using Lambda Expression to calculate the following:
   a. Convert Fahrenheit to Celsius
   b. Convert Kilometers to Miles.

1)Fahrenheit to celcius

```
package something;
interface ftoc
{
public int convert(int c);
}

public class lamb {
public static void main(String[] args)
{
ftoc s1=(c)->{
return ((c-32)*5/9);

};

System.out.println("Value in celcius is:

"+s1.convert(50));
}
}
```

2) Kilometers to miles:

```
package something;
interface ktom
{
public double travel(double c);
}
public class lamb {
public static void main(String[] args)
{
ktom a=(double c)->{
return (c*0.6213);
};
System.out.println("Value in km to miles is:
```

```
"+a.travel(5));
}
}
```

3. Assignments based on web application development using JSP.

A. To design a form and use of JSP Scripting Element and JSP Directive. Display Grade of a student by accepting marks in five subjects.

Q1) To design a form and use of JSP Scripting Element and JSP Directive. Display Grade of a student by accepting marks in five subjects.
CODE:
a) index.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<form action="find.jsp" method="POST">
<h1>Enter Your Name:</h1><input type="text"
name="name"><br>
<h1>Enter Your Marks:</h1><br>
```

```
<h2> MFCS: </h2><input type="text"
name="mfcs"><br>
<h2> AJAVA: </h2><input type="text"
name="java"><br>
<h2> ADBMS: </h2><input type="text"
name="dbms"><br>
<h2> SPM: </h2><input type="text"
name="spm"><br>
<h2> WTL: </h2><input type="text"
name="wtl"><br>
<br><br>
<input type="submit" value="SUBMIT"/>
</form>
</body>
</html>
```

b) grade.jsp

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1>YOUR GRADE</h1>
<%
String name=request.getParameter("name");
int mfcs =
Integer.parseInt(request.getParameter("mfcs"));
int adbms =
Integer.parseInt(request.getParameter("dbms"));
int spm =
Integer.parseInt(request.getParameter("spm"));
int java =
Integer.parseInt(request.getParameter("java"));
int wtl =
Integer.parseInt(request.getParameter("wtl"));
int percent;
percent=(mfcs+adbms+spm+java+wtl)/5;
%>
<h1><%out.println("Name: "+name);%></h1>
<h2>YOUR GRADE IS: </h2>
<%
if(percent<40){
%>
<h1>F</h1>
```

```
<%}
else if(percent>=40&&percent<50){
%>
<h1>D</h1>
<% }
else if(percent>=50 && percent<60){
%>
<h1>B</h1>
<%}
else if(percent>=60 && percent<70){
%>
<h1>A</h1>
<% }
else if(percent>=70 && percent<80){
%>
<h1>A+</h1>
<% }
else{
%>
<h1>O</h1>
<%}
%>
</body>
</html>
```

B. Write a program to design a simple web-based interface to a currency converter application. The interface should consist of a title, suitable instructions, and a form for entering the amount to be converted and an optional currency rate. Use text fields for entering the amount and rate.

```
a) index.html
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>CURRENCY CONVERTER</title>
</head>
<body>
<form method="post" action="lol.jsp">
<h3>Enter the Value:</h3>
<input type="text" name="va" /><br>
<h3>Enter the rate of currency you want to convert

into</h3>
```

```
<input type="text" name="re"/><br><br>
<input type="submit" value="CONVERT" /> <br>
</form>
</body>
</html>
b) conv.jsp

<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<%
int v = Integer.parseInt(request.getParameter("va"));
int r = Integer.parseInt(request.getParameter("re"));
out.print("<h2>VALUE IN CURRENT CURRENCY: "+v+"</h2>");
out.print("<h2>RATE OF CURRENCY YOU WANT TO CONVERT:
"+r+"</h2>");
out.print("<h2>CONVERTED VALUE IN OTHER CURRENCY:
"+r*v+"</h2>");

%>
</body>
</html>
```

C. Design loan calculator using JSP which accepts Period of Time (in years) and Principal Loan Amount. Display the payment amount for each loan and then list the loan balance and interest paid for each payment over the term of the loan for the following time period and interest rate:
    a. 1 to 7 year at 5.35%
    b. 8 to 15 year at 5.5%
    c. 16 to 30 year at 5.75%

```
a) index.html

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
```

```
<title>Loan Calculator</title>
</head>
<body>
<form action = "new.jsp">
<h3>Enter Principle Loan Amount</h3><input type="text"
name='pri'/><br>
<h3>Enter Time Period (IN YEARS)</h3><input type="text"
name='ti'/><br><br>
<input type="submit" value="CALCULATE"/>
</form>
</body>
</html>
```

b) cal.jsp

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>LOAN CALCULATOR</title>
</head>
<body>
<%
double p=Double.parseDouble(request.getParameter("pri"));
int t=Integer.parseInt(request.getParameter("ti"));
double ci;
if(t<=7)
{
double n=5.35/1200;
double temp=Math.pow(1+n,t*12);
double emi=(p*n)*((temp)/(temp-1));
double total=emi*12*t;
ci=(total-p)/t;
int ci2=0;
out.print("<h3>EMI: RS."+(int)emi+"</h3><br>");
out.print("<h3>TOTAL PAYMENT AMOUNT:
RS."+(int)total+"</h3><br>");
for(int i=1;i<=t;i++)
{
total=total-emi*12;
ci2+=ci;
out.print("<h3>YEAR "+i+" <br>LOAN BALANCE:

RS."+(int)total+" <br>INTEREST PAID:
RS."+(int)ci2+"</h3><br>");
}
}
else if(t>=8 && t<=15)
```

```
{
double n=5.5/1200;
double temp=Math.pow(1+n,t*12);
double emi=(p*n)*((temp)/(temp-1));
double total=emi*12*t;
ci=(total-p)/t;
int ci2=0;
out.print("<h3>EMI: RS."+(int)emi+"</h3><br>");
out.print("<h3>TOTAL PAYMENT AMOUNT:
RS."+(int)total+"</h3><br>");
for(int i=1;i<=t;i++)
{
total=total-emi*12;
ci2+=ci;
out.print("<h3>YEAR "+i+" <br>LOAN BALANCE:

RS."+(int)total+" <br>INTEREST PAID:
RS."+(int)ci2+"</h3><br>");
}
}
else
{
double n=5.75/1200;
double temp=Math.pow(1+n,t*12);
double emi=(p*n)*((temp)/(temp-1));
double total=emi*12*t;
ci=(total-p)/t;
int ci2=0;
out.print("<h3>EMI: RS."+(int)emi+"</h3><br>");
out.print("<h3>TOTAL PAYMENT AMOUNT:

RS."+(int)total+"</h3><br>");
for(int i=1;i<=t;i++)
{
total=total-emi*12;
ci2+=ci;
out.print("<h3>YEAR "+i+" <br>LOAN BALANCE:

RS."+(int)total+" <br>INTEREST PAID:
RS."+(int)ci2+"</h3><br>");
}
}
%>
</body>
</html>
```

D. Write a program using JSP that displays a webpage consisting of Application form for change of Study Center which can be filled by any student who wants to change his/ her study center. Make necessary assumptions:

a) index.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<form action = "change.jsp">
<h2>Enter Your Name:</h2>
<input type="text" name ='name'/><br>
<br>
<h2>Enter Your Roll Number:</h2>
<input type="text" name ='roll'/><br>
<br>
<h2>Enter Your Course Name:</h2>
<input type="text" name ='cor'/><br>
<br>
<h2>Select Your Present Study Center</h2>
<select name="vak2">

<option >MUMBAI</option>
<option >KOCHI</option>
<option >CHENNAI</option>
<option >KANPUR</option>

</select>
<h2>Select The Study Center Preference You Want</h2>
<select name="vak3">

<option >MUMBAI</option>
<option >KOCHI</option>
<option >CHENNAI</option>
<option >KANPUR</option>

</select>
<br><br>
<input type="submit" value="SUBMIT"/>
</form>
</body>
</html>
```

b) change.jsp

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>CHAGNE OF DATA</title>
</head>
<body>
<%
String name=request.getParameter("name");
String p1=request.getParameter("vak2");
String p2=request.getParameter("vak3");
String c = request.getParameter("cor");
int roll=Integer.parseInt(request.getParameter("roll"));
%>
<%out.print("<h3>Your Name: "+name+"</h3>"); %>
<%out.print("<h3>Your Roll No: "+roll+"</h3>"); %>



<%out.print("<h3>Your Course Name: "+c+"</h3>"); %>
<%out.print("<h3>Your Current Study Center: "+p1+"</h3>");
%>
<%out.print("<h3>Your Preference Of Study Center:
"+p2+"</h3>"); %>
</body>
</html>
```

Assignment based on Spring Framework

A. Write a program to print “Hello World” using spring framework:

a) Hello.java

```
package hi;
public class hello {
public void dis()
{
System.out.println("Hello World!!!");
}
```

```
}

b) Test.java
package hi;
import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.*;
import org.springframework.core.io.ClassPathResource;
import org.springframework.core.io.Resource;
public class test {
public static void main(String[] args) {
Resource resource=new
ClassPathResource("hi.xml");

@SuppressWarnings("deprecation")
BeanFactory factory=new
XmlBeanFactory(resource);

hello world=(hello)factory.getBean("disp1");
world.dis();
}

}

c) Hi.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:p="http://www.springframework.org/schema/p"
xsi:schemaLocation="http://www.springframework.org/sc
hema/beans

http://www.springframework.org/schema/beans/spring-
beans-3.0.xsd">

<bean id="disp1" class="hi.hello">
</bean>
</beans>
```

B. Write a program to demonstrate dependency injection via setter method.

a) Data.java

```
package sat;
public class dat {
public String name;
public int roll_no;
public String addr;
public String getname()
{
return name;
}
public void setname(String name)
{
this.name=name;
}
public int getroll_no()

{
return roll_no;
}
public void setroll_no(int roll_no)
{
this.roll_no=roll_no;
}

public String getaddr()
{
return addr;
}
public void setaddr(String addr)
{
this.addr=addr;
}
public void dis()
{
System.out.println("NAME: "+name+"\nROLL NO:
"+roll_no+"\nADDRESS: "+addr+"\n");
}
}
```

b) Main.java

```
package sat;
import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;

import org.springframework.core.io.*;
@SuppressWarnings("deprecation")
public class mai {
public static void main(String args[])
{
Resource r=new ClassPathResource("dataa.xml");
BeanFactory factory=new XmlBeanFactory(r);
```

```
dat e=(dat)factory.getBean("obj");
e.dis();
}
}


c) Data.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:p="http://www.springframework.org/schema/p"
xsi:schemaLocation="http://www.springframework.org/sc
hema/beans

http://www.springframework.org/schema/beans/spring-
beans-3.0.xsd">

<bean id="obj" class="sat.dat">
<property name="name">
<value>Tony Stark</value>
</property>
<property name="roll_no">
<value>27</value>
</property>
<property name="addr">
<value>California</value>
</property>
</bean>
</beans>
```

C. Write a program to demonstrate dependency injection via Constructor.

```
a) Data.java
package cons;
public class dat {
private String name;
private String addr;
private int roll_no;
public dat(String name,String addr,int roll_no)
{
this.name=name;
```

```
this.addr=addr;
this.roll_no=roll_no;
}
public String getname()
{
return name;
}
public String getaddr()
{
return addr;
}
public int getroll()
{
return roll_no;


}
public void dis()
{
System.out.println("\nNAME: "+name+"\nADDRESS:
"+addr+"\nROLL NO: "+roll_no+"\n");
}
}
```

b) Main.java

```
package cons;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.beans.factory.BeanFactory;

public class test {
public static void main(String args[])
{
ApplicationContext a = new ClassPathXmlApplicationContext("mai.xml");
dat d = (dat) a.getBean("obj");

d.dis();
}
}
```

c) Data.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:p="http://www.springframework.org/schema/p"
xsi:schemaLocation="http://www.springframework.org/sc
hema/beans

http://www.springframework.org/schema/beans/spring-
beans-3.0.xsd">

<bean id="obj" class = "cons.dat">

<constructor-arg value="Atharva Kale" ></constructor-
arg>

<constructor-arg value="Kalyan" ></constructor-arg>

<constructor-arg value="27" type="int"></constructor-
arg>

</bean>
</beans>
```

Assignment based on AOP

A. Q1) Create class car, bike, airplane, create an aspect engine, create method drive() in car, ride() in bike, fly in airplane, the engine aspect has method enginestart() and enginestop(). When you run the application the enginestart() method should execute before drive(), ride() and fly() a enginestop() method should run after drive(), ride() and fly().

```
a. car.java
package some.pro1;
import org.springframework.stereotype.Component;
@Component
public class car {
public void drive()
{
System.out.println("CAR STARTED DRIVING");
}
}
b. bike.java
package some.pro1;
import org.springframework.stereotype.Component;
@Component
public class bike {
```

```
public void ride()
{
System.out.println("Biker riding bike");
}
}
```

c. airplane.java

```
package some.pro1;
import org.springframework.stereotype.Component;
@Component
public class airplane {
public void fly()
{
System.out.println("Airplane flying");
}
}
```

d. engine.java

```
package some.pro1;
import org.aspectj.lang.annotation.After;
import org.aspectj.lang.annotation.Before;
import org.aspectj.lang.annotation.Aspect;
import
org.springframework.context.annotation.EnableAspectJAutoProx
y;
import org.springframework.stereotype.Component;
@Component
@Aspect
@EnableAspectJAutoProxy
public class engine {
@Before("execution(public void drive())")
public void enginestart()
{
System.out.println("Enigine started");
}
@After("execution(public void fly())")
public void enginestop()
{


System.out.println("Enigine stopped");
}
}
```

e. app.java

```
package some.pro1;
import
org.springframework.context.annotation.AnnotationConfigApplicationCo
ntext;
```

```
import org.springframework.context.support.AbstractApplicationContext;

public class App
{
public static void main( String[] args )
{
AbstractApplicationContext context = new
AnnotationConfigApplicationContext(AppConfig.class);
car c = context.getBean("car",car.class);
c.drive();
bike b = context.getBean("bike",bike.class);
b.ride();
airplane a = context.getBean("airplane",airplane.class);
a.fly();
}
}
```

B. Create class car, bike, airplane, create an aspect engine, create method drive() in car, ride() in bike, fly in airplane, the engine aspect has method enginestart() and enginestop(). When you run the application the enginestart() method should execute before drive(), ride() and fly() a enginestop() method should run after drive(), ride() and fly() by using pointcuts.

a. car.java

```
package something.pro2;
import org.springframework.stereotype.Component;
@Component

public class car {
public void drive()
{
System.out.println("CAR STARTED DRIVING");
}
}
```

b. bike.java

```
package something.pro2;
import org.springframework.stereotype.Component;
@Component
public class bike {
public void ride()
{
System.out.println("Biker riding bike");
}
}
```

c. airplane.java

```
package something.pro2;
import org.springframework.stereotype.Component;
@Component
public class airplane {
public void fly()

{
System.out.println("Airplane Flying");
}
}
```

d. engine.java

```
package something.pro2;
import org.aspectj.lang.annotation.After;
import org.aspectj.lang.annotation.AfterReturning;
import org.aspectj.lang.annotation.Before;
import org.aspectj.lang.annotation.Pointcut;
import org.aspectj.lang.annotation.Aspect;
import
org.springframework.context.annotation.EnableAspectJAutoProx
y;
import org.springframework.stereotype.Component;
@Component
@Aspect
@EnableAspectJAutoProxy
public class engine {

@Before("getNamePointcut()")
public void enginestart()
{
System.out.println("Engine started");
}
@Pointcut("execution(public void drive())")
public void getNamePointcut(){}
@After("getNamePointcut1()")
public void enginestop()
{


System.out.println("Engine stopped");
}
@Pointcut("execution(public void fly())")
public void getNamePointcut1(){}
}
```

e. app.java

```
package something.pro2;
import
org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.support.AbstractApplicationContext;
```

```
public class App
{
public static void main( String[] args )
{
AbstractApplicationContext context = new
AnnotationConfigApplicationContext(AppConfig.class);
car c = context.getBean("car",car.class);
c.drive();
bike b = context.getBean("bike",bike.class);
b.ride();

airplane a = context.getBean("airplane",airplane.class);
a.fly();
}
}
```

C. Create a business class multiplier (int a, int b) which returns product of a and b create an aspect AdderAfterReturnAspect use After-returning advice.

```
a. multiplier.java
package something.pro3;
import org.springframework.stereotype.Component;

@Component
public class multiplier {
public int mul(int a,int b)
{
return a*b;
}
}

b. AdderAfterReturnAspect.java
package something.pro3;
import org.aspectj.lang.annotation.After;
import org.aspectj.lang.annotation.AfterReturning;
import org.aspectj.lang.annotation.Before;
import org.aspectj.lang.annotation.Aspect;
import
org.springframework.context.annotation.EnableAspectJAutoProx
y;
import org.springframework.stereotype.Component;

@Component
@Aspect
@EnableAspectJAutoProxy
public class AdderAfterReturnAspect {
@AfterReturning(pointcut="execution(*
```

```
something.pro3.multiplier.*(..))", returning="retVal")
public void afterreturning(Object retVal)

throws Throwable
{
System.out.println("Multiplication

is:"+retVal+"");
}

}
```

c. App.java

```
package something.pro3;
import
org.springframework.context.annotation.AnnotationConfigAppli
cationContext;
import
org.springframework.context.support.AbstractApplicationConte
xt;
public class App
{
public static void main( String[] args )
{
AbstractApplicationContext context = new
AnnotationConfigApplicationContext(AppConfig.class);

multiplier
m=context.getBean("multiplier",multiplier.class);
m.mul(5, 5);
}
}
```

D. Create a voter class with attribute name and age create an exception check if age is less than 18 throw exception. Create and aspect illegalVoter with a method if the voter class throws exception the illegalVoter aspect method will run

a. voter.java

```
package something.pro4;
import org.springframework.stereotype.Component;
@Component
public class voter {
public void validate() throws Exception
{
int age=19;
String name="Tony";
if(age<18)
{
```

```
throw new Exception(name+" you are under age!");
}
else
{
System.out.println("Do vote "+name+" as your age is
"+age);

}
}
}
```

b. illegalvoter.java

```
package something.pro4;
import org.aspectj.lang.annotation.AfterThrowing;
import org.aspectj.lang.annotation.Aspect;
import
org.springframework.context.annotation.EnableAspectJAutoProx
y;
import org.springframework.stereotype.Component;

@Component
@Aspect
@EnableAspectJAutoProxy

public class illegalvoter {
@AfterThrowing(pointcut="execution(*
something.pro4.voter.*(..))",throwing = "ex")
public void illegalVoter(Exception ex) throws
Throwable
{
System.out.println("LoggingAspect.logAfterThrowingAllMethods
() "+ ex);
}
}
```

c. app.java

```
package something.pro4;
import
org.springframework.context.annotation.AnnotationConfigAppli
cationContext;
import
org.springframework.context.support.AbstractApplicationConte
xt;
public class App
{
public static void main( String[] args ) throws
Exception
{
AbstractApplicationContext context = new
AnnotationConfigApplicationContext(AppConfig.class);
```

```
voter v=context.getBean("voter",voter.class);
v.validate();
}
}
```

Implementation Of JDBC

A. Write a program to insert, update and delete records from the given table. (Employee Table-Id ,Name, Age)

Create a database in MySQL and create table named in that database with attributes id int , name varchar(30) and age int.

a. employee.java

```
package something.datapro1;
public class employee {
private int id;

private String name;
private int age;
public employee(int id, String name, int age) {
this.id = id;
this.name = name;
this.age = age;
}
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public int getAge() {
return age;
}
```

```
public void setAge(int age) {
this.age = age;
}
```
b. EmployeeDao.java

```
package something.datapro1;
import org.springframework.jdbc.core.JdbcTemplate;
public class EmployeeDao {
private JdbcTemplate jdbcTemplate;

public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
this.jdbcTemplate = jdbcTemplate;
}
public int saveEmployee(employee e){
String query="insert into employee_27
values('"+e.getId()+"','"+e.getName()+"','"+e.getAge()+"')";
return jdbcTemplate.update(query);
}
public int updateEmployee(employee e){
String query="update employee_27 set
name='"+e.getName()+"',age='"+e.getAge()+"' where
id='"+e.getId()+"' ";
return jdbcTemplate.update(query);
}
public int deleteEmployee(employee e){
String query="delete from employee_27 where
id='"+e.getId()+"' ";
return jdbcTemplate.update(query);
}
}
```

c. App.java

```
package something.datapro1;
import org.springframework.context.ApplicationContext;
import
org.springframework.context.support.ClassPathXmlApplicationContex
t;


public class App
{
public static void main( String[] args )
{
ApplicationContext ctx=new
ClassPathXmlApplicationContext("dbd.xml");
```

```
EmployeeDao dao=(EmployeeDao)ctx.getBean("edao");
dao.saveEmployee(new employee(1,"Atharva",21));
dao.saveEmployee(new employee(2,"Tony",48));
dao.saveEmployee(new employee(3,"Jarvis",50));
System.out.print("DATA INSERTED");
dao.updateEmployee(new employee(1,"Atharva",20));
System.out.print("DATA UPDATED");

System.out.print("DATA DELETED");
employee e=new employee(0, null, 0);
e.setId(3);
dao.deleteEmployee(e);
}
}
```

d. dbd.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"
xsi:schemaLocation =
"http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-
3.0.xsd ">

<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerData
Source">
<property name="driverClassName" value =
"com.mysql.jdbc.Driver" />
<property name="url" value =
"jdbc:mysql://localhost:3306/java" />
<property name="username" value = "COOLBOy" />
<property name="password" value = "hailhydra" />
</bean>

<bean id = "jdbcTemplate" class =
"org.springframework.jdbc.core.JdbcTemplate">
<property name = "dataSource" ref = "dataSource" />
</bean>
<bean id="edao" class="something.datapro1.EmployeeDao">
<property name="jdbcTemplate" ref="jdbcTemplate"></property>
```

```
</bean>
</beans>
```

B. Write a program to demonstrate PreparedStatement in Spring JdbcTemplate.

a. employee.java

```
package something.datapro2;
public class employee {
private Integer age;
private String name;
private Integer id;
public employee(){}
public employee(String name,Integer id, Integer age) {
this.id = id;
this.name = name;
this.age=age;
}
public void setAge(Integer age) {
this.age = age;
}
public Integer getAge() {
return age;
}
public void setName(String name) {
this.name = name;
}
public String getName() {
return name;
}
public void setId(Integer id) {
this.id = id;
}
public Integer getId() {
return id;
}
}
```

b. employeetemp.java

```
package something.datapro2;
import java.sql.PreparedStatement;
```

```
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.sql.ResultSet;
import javax.sql.DataSource;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.ResultSetExtractor;
import
org.springframework.jdbc.core.PreparedStatementCallback;
public class employeetemp implements EmployeeDao {
private DataSource dataSource;
private JdbcTemplate jdbcTemplateObject;
public void setDataSource(DataSource dataSource) {
this.dataSource = dataSource;
this.jdbcTemplateObject = new JdbcTemplate(dataSource);
}
public Boolean insertPreparedStatmentEmployee(final employee
e){
String query="insert into employee_27 (id,name, age) values
(?,?,?)";
return jdbcTemplateObject.execute(query,new
PreparedStatementCallback<Boolean>(){
public Boolean doInPreparedStatement(PreparedStatement ps)
throws SQLException, DataAccessException {
ps.setInt(1,e.getId());
ps.setString(2,e.getName());
ps.setInt(3,e.getAge());
System.out.println("Records Inserted Using Prepared
Statment");
return ps.execute();

}
});
}
@Override
public void insert(Integer id,String name, Integer age) {
// TODO Auto-generated method stub
}
@Override
public void update(Integer id, String name, Integer age) {
// TODO Auto-generated method stub
}
@Override
public void delete(Integer id) {
// TODO Auto-generated method stub
```

```
}}

c. EmployeeDao.java
package something.datapro2;
import java.util.List;
import javax.sql.DataSource;
import org.springframework.jdbc.core.JdbcTemplate;
public interface EmployeeDao {
public void setDataSource(DataSource ds);
public void insert(Integer id,String name, Integer age);
public void update(Integer id,String name,Integer age);
public void delete(Integer id);
}

d. App.java

package something.datapro2;
import java.util.List;

import org.springframework.context.ApplicationContext;
import
org.springframework.context.support.ClassPathXmlApplicationC
ontext;

public class App
{
public static void main( String[] args )
{
ApplicationContext context =new
ClassPathXmlApplicationContext("dbd.xml");
employeetemp et =
(employeetemp)context.getBean("employeejdbctemp");
System.out.println("Prepared Statement
Example");
employee e = (employee)
context.getBean("employee");
et.insertPreparedStatmentEmployee(e);
}

}

e. dbd.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"
```

```
xsi:schemaLocation="http://www.springframework.org/schema/be
ans


http://www.springframework.org/schema/beans/spring-beans-
3.0.xsd ">

<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerData
Source">
<property name="driverClassName"
value="com.mysql.jdbc.Driver"/>
<property name="url"
value="jdbc:mysql://localhost:3306/java"/>
<property name="username" value="COOLBOy"/>
<property name="password" value="hailhydra"/>
</bean>
<bean id="employeejdbctemp"
class="something.datapro2.employeetemp">
<property name="dataSource" ref="dataSource" />
</bean>
<bean id="employee" class="something.datapro2.employee">
<property name="id" value="3"></property>
<property name="name" value="Natasha"></property>
<property name="age" value="25"></property>
</bean>
</beans>
```

Implementation of RESTful Web Services

A. Write a program to demonstrate RESTful Web Services with spring boot.

```
a. pom.xml (FOR DEPENDENCIES AND PARENT TAG)
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<groupId>anything</groupId>
<artifactId>spriboot</artifactId>
```

```
<version>0.0.1-SNAPSHOT</version>
<name>spriboot</name>
<!-- FIXME change it to the project's website -->
<url>http://www.example.com</url>
<properties>

<project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>

<maven.compiler.source>1.7</maven.compiler.source>

<maven.compiler.target>1.7</maven.compiler.target>
</properties>
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>2.0.3.RELEASE</version>
</parent>
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
<groupId>junit</groupId>
<artifactId>junit</artifactId>
<version>4.11</version>
<scope>test</scope>
</dependency>
<dependency>
<groupId>javax.servlet</groupId>
<artifactId>javax.servlet-api</artifactId>
<version>3.1.0</version>
</dependency>
</dependencies>
<build>
<pluginManagement><!-- lock down plugins versions to
avoid using Maven defaults (may be moved to parent pom) -->
<plugins>
<!-- clean lifecycle, see
```

```
https://maven.apache.org/ref/current/maven-
core/lifecycles.html#clean_Lifecycle -->

<plugin>
<artifactId>maven-clean-plugin</artifactId>
<version>3.1.0</version>
</plugin>
<!-- default lifecycle, jar packaging: see

https://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin_bindings_for_jar_packaging -->

<plugin>
<artifactId>maven-resources-plugin</artifactId>
<version>3.0.2</version>
</plugin>
<plugin>
<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-
plugin</artifactId>

</plugin>
<plugin>
<artifactId>maven-compiler-plugin</artifactId>
<version>3.8.0</version>
</plugin>
<plugin>
<artifactId>maven-surefire-plugin</artifactId>
<version>2.22.1</version>
</plugin>
<plugin>
<artifactId>maven-jar-plugin</artifactId>
<version>3.0.2</version>
</plugin>
<plugin>
<artifactId>maven-install-plugin</artifactId>
<version>2.5.2</version>
</plugin>
<plugin>
<artifactId>maven-deploy-plugin</artifactId>
<version>2.8.2</version>
</plugin>
<!-- site lifecycle, see

https://maven.apache.org/ref/current/maven-
```

```
core/lifecycles.html#site_Lifecycle -->

<plugin>
<artifactId>maven-site-plugin</artifactId>
<version>3.7.1</version>
</plugin>
<plugin>

<artifactId>maven-project-info-reports-
plugin</artifactId>

<version>3.0.0</version>
</plugin>
</plugins>
</pluginManagement>
</build>
</project>

b. App.java
package anything.spriboot;
import org.springframework.boot.*;
import org.springframework.boot.autoconfigure.*;
import org.springframework.web.bind.annotation.*;
@RestController
@EnableAutoConfiguration
public class App
{
@RequestMapping("/")
String home() {
return "Hello World Spring Boot!";
}
public static void main( String[] args )
throws Exception {
SpringApplication.run(App.class, args);
}
}
```