Aspect Oriented Programming

1. Write a program to demonstrate Spring AOP – before advice.
2. Write a program to demonstrate Spring AOP – after advice.
3. Write a program to demonstrate Spring AOP – around advice.
4. Write a program to demonstrate Spring AOP – after returning advice.
5. Write a program to demonstrate Spring AOP – after throwing advice.
6. Write a program to demonstrate Spring AOP – pointcuts.

**Problem Statement 1 :** Write a program to demonstrate Spring AOP – before advice.

# Solution :

**beforeaop.java**

package bvimit.edu;

import org.aspectj.lang.JoinPoint;

import org.aspectj.lang.annotation.Aspect; import org.aspectj.lang.annotation.Before; import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class beforeaop {

@Pointcut("execution(int beforeoperation.\*(..))") public void p(){}

@Before("p()")

public void myadvice(JoinPoint jp)

{

System.out.println("before advice");

}

}

# beforeoperation.java

package bvimit.edu;

public class beforeoperation {

public void msg() {System.*out*.println("method 1");}

public int m(){System.*out*.println("method 2 with return");return 2;} public int k(){System.*out*.println("method 3 with return");return 3;}

}

# aopctx1.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"*[*http://www.springframework.org/schema/beans"*](http://www.springframework.org/schema/beans)xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"* xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)

[*http://www.springframework.org/schema/beans/spring-beans.xsd"*](http://www.springframework.org/schema/beans/spring-beans.xsd)>

<bean id=*"opBean"* class=*"bvimit.edu.beforeoperation"*> </bean>

<bean id=*"trackMyBean"* class=*"bvimit.edu.beforeaop"*></bean>

<bean class=*"org.springframework.aop.aspectj.annotation.AnnotationAwareAspectJAutoProxyCreator "*></bean>

</beans> **beforetest.java** package bvimit.edu;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext; public class beforetest {

public static void main(String[] args) { ApplicationContext context = new

ClassPathXmlApplicationContext("aopctx1.xml");

beforeoperation e = (beforeoperation) context.getBean("opBean"); System.out.println("calling m1. ");

e.msg();

System.out.println("calling m2. ");

e.m();

System.out.println("calling m3. ");

e.k();

}

}

# Output :

**Problem Statement 2 :** Write a program to demonstrate Spring AOP – after advice.

# Solution :

**Afteraopdata.java**

package bvimit.edu;

import org.aspectj.lang.JoinPoint; import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.Aspect; import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class afteraopdata {

@Pointcut("execution(int afteroperation.\*(..))") public void p(){}

@After("p()")

public void myadvice(JoinPoint jp)

{

System.out.println("after advice");

}

}

# afteroperation.java

package bvimit.edu;

public class afteroperation {

public void msg() {System.*out*.println("method 1");}

public int m(){System.*out*.println("method 2 with return");return 2;} public int k(){System.*out*.println("method 3 with return");return 3;}

}

# aopctx.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"*[*http://www.springframework.org/schema/beans"*](http://www.springframework.org/schema/beans)xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"* xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)

[*http://www.springframework.org/schema/beans/spring-beans.xsd"*](http://www.springframework.org/schema/beans/spring-beans.xsd)>

<bean id=*"opBean"* class=*"bvimit.edu.afteroperation"*> </bean>

<bean id=*"trackMyBean"* class=*"bvimit.edu.afteraopdata"*></bean>

<bean class=*"org.springframework.aop.aspectj.annotation.AnnotationAwareAspectJAutoProxyCreator "*></bean>

</beans> **aftertest.java** package bvimit.edu;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext; public class aftertest {

public static void main(String[] args) { ApplicationContext context = new

ClassPathXmlApplicationContext("aopctx.xml");

afteroperation e = (afteroperation) context.getBean("opBean"); System.out.println("calling m1. ");

e.msg();

System.out.println("calling m2. ");

e.m();

System.out.println("calling m3. ");

e.k();

}

}

# Output :

**Problem Statement 3 :** Write a program to demonstrate Spring AOP – around advice.

# Solution :

**Bankaopdata.java**

package bvimit.edu;

import org.aspectj.lang.ProceedingJoinPoint; import org.aspectj.lang.annotation.Around; import org.aspectj.lang.annotation.Aspect; import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class Bankaopdata {

@Pointcut("execution(\* Bank.\*(..))") public void a() {}

@Around("a()")

public Object myadvice(ProceedingJoinPoint p)throws Throwable

{

System.out.println("Around concern Before calling actual method"); Object obj=p.proceed();

System.out.println("Around Concern After calling actual method"); return obj;

}

}

# Bank.java

package bvimit.edu;

public class Bank {

public void welcome() {System.*out*.println("welcome to bank");}

public int icici() {System.*out*.println("icici bank interest rate");return 7;} public int pnb() {System.*out*.println("pnb bank interest rate");return 6;}

}

# Bankaopdata.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"*[*http://www.springframework.org/schema/beans"*](http://www.springframework.org/schema/beans)xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"* xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)

[*http://www.springframework.org/schema/beans/spring-beans.xsd"*](http://www.springframework.org/schema/beans/spring-beans.xsd)>

<bean id=*"opBean"* class=*"bvimit.edu.Bank"*> </bean>

<bean id=*"trackMyBean"* class=*"bvimit.edu.Bankaopdata"*></bean>

<bean class=*"org.springframework.aop.aspectj.annotation.AnnotationAwareAspectJAutoProxyCreator "*></bean>

</beans>

# Banktest.java

package bvimit.edu;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext; public class Banktest {

private static ApplicationContext *context*;

public static void main(String[] args) {

*context* = new ClassPathXmlApplicationContext("Bankaopdata.xml");

Bank e =(Bank) *context*.getBean("opBean"); System.*out*.println("Calling welcome method..."); e.welcome();

System.*out*.println("Calling icici method..."); e.icici();

System.*out*.println("Calling pnb method..."); e.pnb();

}

}

Output :

**Problem Statement 4 :** Write a program to demonstrate Spring AOP – after returning advice.

# Solution :

**Bankaopdata.java**

package bvimit.edu;

import org.aspectj.lang.JoinPoint;

import org.aspectj.lang.ProceedingJoinPoint; import org.aspectj.lang.annotation.AfterReturning; import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect; import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class Bankaopdata {

@AfterReturning(

pointcut ="execution(\* Bank.\*(..))", returning="result")

public void myadvice(JoinPoint jp,Object result)

{

System.*out*.println("AfterReturning concern"); System.*out*.println("Result in advice" +result);

}

}

# Bank.java

package bvimit.edu;

public class Bank {

public void welcome() {System.*out*.println("welcome to bank");}

public int icici() {System.*out*.println("icici bank interest rate");return 7;} public int pnb() {System.*out*.println("pnb bank interest rate");return 6;}

}

# Bankaopdata.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"*[*http://www.springframework.org/schema/beans"*](http://www.springframework.org/schema/beans)xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"* xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)

[*http://www.springframework.org/schema/beans/spring-beans.xsd"*](http://www.springframework.org/schema/beans/spring-beans.xsd)>

<bean id=*"opBean"* class=*"bvimit.edu.Bank"*> </bean>

<bean id=*"trackMyBean"* class=*"bvimit.edu.Bankaopdata"*></bean>

<bean class=*"org.springframework.aop.aspectj.annotation.AnnotationAwareAspectJAutoProxyCreator "*></bean>

</beans>

# Banktest.java

package bvimit.edu;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext; public class Banktest {

private static ApplicationContext *context*;

public static void main(String[] args) {

*context* = new ClassPathXmlApplicationContext("Bankaopdata.xml");

Bank e =(Bank) *context*.getBean("opBean");

//System.out.println("Calling welcome method..."); e.welcome();

//System.out.println("Calling icici method..."); e.icici();

//System.out.println("Calling pnb method..."); e.pnb();

}

}

Output :

**Problem Statement 5 :** Write a program to demonstrate Spring AOP – after throwing advice.

# Solution :

**Operationaop\_at.java**

package bvimit.edu;

import org.aspectj.lang.JoinPoint;

import org.aspectj.lang.annotation.AfterThrowing; import org.aspectj.lang.annotation.Aspect;

@Aspect

public class Operationaop\_at { @AfterThrowing(

pointcut = "execution(\* Operation\_at.\*(..))", throwing = "error") public void myadvice(JoinPoint jp, Throwable error)

{

System.*out*.println("AfterThrowing concern"); System.*out*.println("Exception is: "+error); System.*out*.println("end of after throwing advice ");

}

}

# Operation\_at.java

package bvimit.edu;

public class Operation\_at {

public void validate(int att)throws Exception{ if(att<75) {

throw new ArithmeticException("Not eligible for exam");

}

else {

}

}

}

System.*out*.println("Eligible for exam");

# validctx.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"*[*http://www.springframework.org/schema/beans"*](http://www.springframework.org/schema/beans)xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"* xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)

[*http://www.springframework.org/schema/beans/spring-beans.xsd"*](http://www.springframework.org/schema/beans/spring-beans.xsd)>

<bean id=*"opBean"* class=*"bvimit.edu.Operation\_at"*></bean>

<bean id=*"trackMyBean"* class=*"bvimit.edu.Operationaop\_at"*></bean>

<bean class=*"org.springframework.aop.aspectj.annotation.AnnotationAwareAspectJAutoProxyCreator "*></bean></beans>

# TestValidation.java

package bvimit.edu;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class OperationTest\_at {

private static ApplicationContext context;

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("validctx.xml"); Operation\_at op = (Operation\_at) context.getBean("opBean"); System.out.println("calling validate ");

try {

op.validate(85);

}catch(Exception e){System.out.println(e);} System.out.println("calling validate again. ");

try {

op.validate(25);

}catch(Exception e){System.out.println(e);}

}

}

Q6. Program with pointcuts

# beforeaop.java

**import** org.aspectj.lang.JoinPoint;

**import** org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before; **import** org.aspectj.lang.annotation.Pointcut; @Aspect

**public class** beforeaop {

@Pointcut("execution(int beforeoperation.\*(..))")

**public void** p(){}

@Before("p()")

**public void** myadvice(JoinPoint jp)

{

System.***out***.println("before advice LOG FILES");

}

}

# beforeoperation.java

**public class** beforeoperation {

**public int** m(){System.***out***.println("project file");**return** 2;}

}

# aopctx1.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=[*"http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"* xmlns:xsi=[*"http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"* xsi:schemaLocation=[*"http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)

[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)*"*>

<bean id=*"opBean"* class=*"beforeoperation"*> </bean>

<bean id=*"trackMyBean"* class=*"beforeaop"*></bean>

<bean class=*"org.springframework.aop.aspectj.annotation.AnnotationAwareAspectJAuto ProxyCreator"*></bean>

</beans>

# beforetest.java

**import** org.springframework.context.ApplicationContext;

# import

org.springframework.context.support.ClassPathXmlApplicationContext;

**public class** beforetest {

**public static void** main(String[] args) { ApplicationContext context = **new** ClassPathXmlApplicationContext("aopctx1.xml");

beforeoperation e = (beforeoperation) context.getBean("opBean");

System.***out***.println("calling method ");

e.m();

}

}