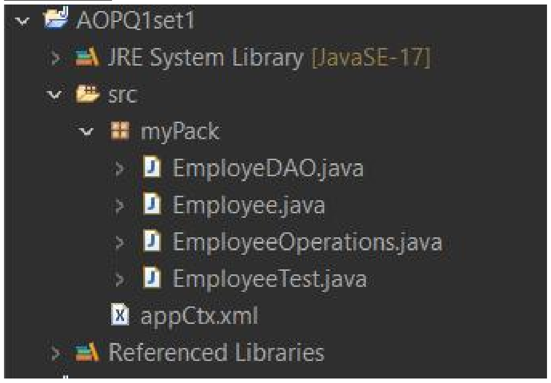
Set1- Write a program to demonstrate Spring AOP-after returning advice

Code:-



```
AOPQ1set1
  JRE System Library [JavaSE-17]
  src
    myPack
      EmployeDAO.java
      Employee.java
      EmployeeOperations.java
      EmployeeTest.java
    appCtx.xml
  Referenced Libraries
```

Employee.java Class:-

```
package myPack;

public class Employee {
String name;
int deptID;
String address;
double sal;
public Employee(String name, int deptID, String address, double sal) {
super();
this.name = name;
this.deptID = deptID;
this.address = address;
this.sal = sal;
}
public String getName() {
return name;
}

public void setName(String name) {
this.name = name;
}

public int getDeptID() {
return deptID;
}

public void setDeptID(int deptID) {
this.deptID = deptID;
}

public String getAddress() {
return address;
}

public void setAddress(String address) {
```

```
this.address = address;
}

public double getSal() {
return sal;
}

public void setSal(double sal) {
this.sal = sal;
}

@Override
public String toString() {
return "Employee [name=" + name + ", deptID=" + deptID + ", address=" + address + ",
sal=" + sal + "]";
}
}
```

EmployeeDAO.java Class:-

```
package myPack;

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

@Aspect
public class EmployeDAO.java {
        @Pointcut("execution(* EmployeeOperations.calSal (..))")
        public void pointcutMethod() {

        }
        @AfterReturning("pointcutMethod()")
        public void afterReturing(JoinPoint jp)
        {
                System.out.println("Inside afterReturing()  ");
        }
}
```

EmployeeOperations.java Class:-

```
package myPack;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

@Component
public class EmployeeOperations {
        @Autowired
        public EmployeeOperations()
```

```
	{
		super();
	}
	public double calSal(double salary) {
		System.out.println("salary Calculated ");
		return salary+20000;
	}

}
```

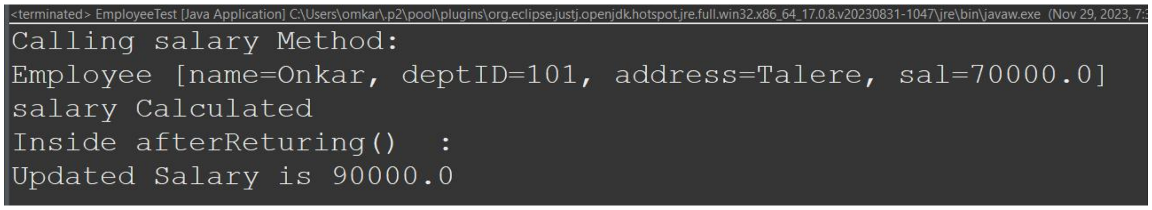
appCtx.xml:-

```
<?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:aop="http://www.springframework.org/schema/aop"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-3.0.xsd">
<aop:aspectj-autoproxy proxy-target-class="true" />
<bean id="operationBean" class="myPack.EmployeeOperations"></bean>
<bean id="employeeDAO" class="myPack.EmployeDAO"></bean>
</beans>
```

EmployeeTest.java Class:-

```
package myPack;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class EmployeeTest {
	private static ApplicationContext ctx;
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ctx = new ClassPathXmlApplicationContext("appCtx.xml");
		Employee e = new Employee("Onkar", 101, "Talere", 70000);
		EmployeeOperations                    op                    =
(EmployeeOperations)ctx.getBean("operationBean");
		System.out.println("Calling salary Method:");

		double sal = op.calSal(e.getSal());
		System.out.println("Updated Salary is "+sal);

	}
}
```

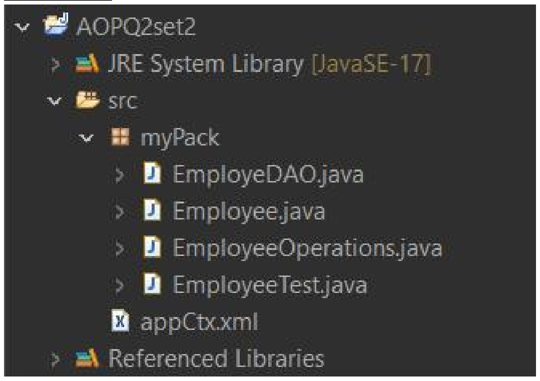
Output:-



```
<terminated> EmployeeTest [Java Application] C:\Users\omkar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (Nov 29, 2023, 7:
Calling salary Method:
Employee [name=Onkar, deptID=101, address=Talere, sal=70000.0]
salary Calculated
Inside afterReturing()  :
Updated Salary is 90000.0
```

Set2-Write a program to demonstrate Spring AOP-before advice

Code:-



```
AOPQ2set2
  JRE System Library [JavaSE-17]
  src
    myPack
      EmployeDAO.java
      Employee.java
      EmployeeOperations.java
      EmployeeTest.java
    appCtx.xml
  Referenced Libraries
```

Employee.java Class:-

```
package myPack;

public class Employee {
String name;
int deptID;
String address;
double sal;
public Employee(String name, int deptID, String address, double sal) {
super();
this.name = name;
this.deptID = deptID;
this.address = address;
this.sal = sal;
}
public String getName() {
return name;
}

public void setName(String name) {
this.name = name;
}

public int getDeptID() {
return deptID;
}

public void setDeptID(int deptID) {
this.deptID = deptID;
}

public String getAddress() {
return address;
}

public void setAddress(String address) {
this.address = address;
}
public double getSal() {
return sal;
}

public void setSal(double sal) {
this.sal = sal;
}

@Override
public String toString() {
return "Employee [name=" + name + ", deptID=" + deptID + ", address=" + address + ",
sal=" + sal + "]";
}
}
```

EmployeeDAO.java Class:-

```
package myPack;

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 EmployeDAO {
	@Pointcut("execution(* EmployeeOperations.calSal (..))")
	public void pointcutMethod() {

	}
	@Before("target(myPack.EmployeeOperations)")
	public void afterReturing(JoinPoint jp)
	{
		System.out.println("Inside afterReturing()  ");
	}
}
```

EmployeeOperations.java Class:-

```
package myPack;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

@Component
public class EmployeeOperations {
	@Autowired
	public EmployeeOperations()
	{
		super();
```

```
        }
        public double calSal(double salary) {
                System.out.println("salary Calculated ");
                return salary+2000;
        }
}
```

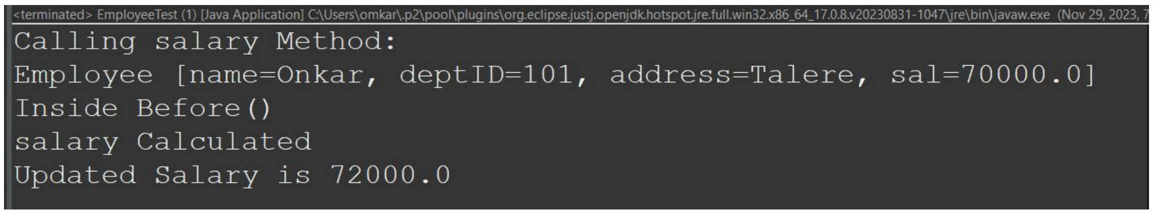
appCtx.xml:-

```
<?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:aop="http://www.springframework.org/schema/aop"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-3.0.xsd">

<aop:aspectj-autoproxy proxy-target-class="true" />

<bean id="operationBean" class="myPack.EmployeeOperations"></bean>
<bean id="employeeDAO" class="myPack.EmployeDAO"></bean>
</beans>
```

EmployeeTest.java Class:-

```
package myPack;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class EmployeeTest {
private static ApplicationContext ctx;
public static void main(String[] args) {
// TODO Auto-generated method stub
ctx = new ClassPathXmlApplicationContext("appCtx.xml");
Employee e = new Employee("Onkar", 101, "Talere", 70000);
EmployeeOperations op = (EmployeeOperations)ctx.getBean("operationBean");
System.out.println("Calling salary Method:");
double sal = op.calSal(e.getSal());
System.out.println("Updated Salary is "+sal);
}
}
```

Output:-



```
<terminated> EmployeeTest (1) [Java Application] C:\Users\omkar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (Nov 29, 2023, 7
Calling salary Method:
Employee [name=Onkar, deptID=101, address=Talere, sal=70000.0]
Inside Before()
salary Calculated
Updated Salary is 72000.0
```