Servlet, JSP and Spring Core Skill Based Assessment

1. Write a JSP that takes the user’s name and age from a form. -Echo backs the name and age along with a message stating the price of movie tickets. -The price is determined by the age passed to the JSP. -If the age is greater than 62, the movie ticket price is $7.00. -If the user is less than 10 years old, the price is $5.00. -For everyone else, the price is $9.50.

Movie.jsp-

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<%

String name=request.getParameter("uname");

**int** age=Integer.parseInt(request.getParameter("age"));

**if**(age>62)

{

out.println("My Name is "+name+" "+"and age is "+age+" "+" price of movie ticket is $7.00");

}

**else** **if**(age<10)

{

out.println("My Name is "+name+" "+"and age is "+age+" "+" price of movie ticket is $5.00");

}

**else**

{

out.println("My Name is "+name+" "+" and age is "+age+" "+" price of movie ticket is $9.50");

}

%>

</body>

</html>

MovieIndex.jsp-

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<form action=*"Movie.jsp"*>

Enter Name:<input type=*"textbox"* name=*"uname"*/></br>

Enter Age:<input type=*"number"* name=*"age"*></br>

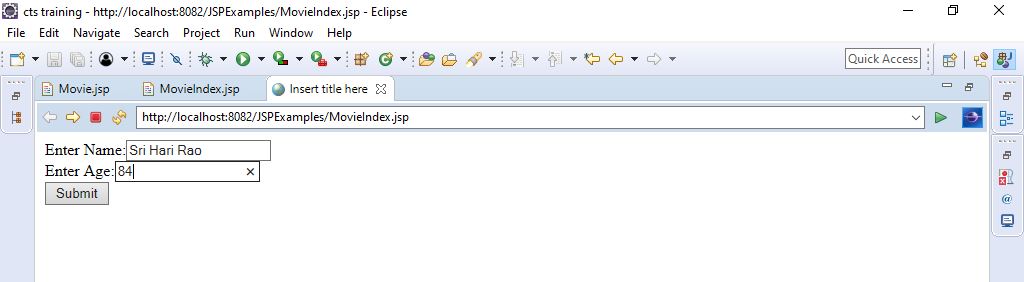
<input type=*"submit"* value=*"Submit"*/>

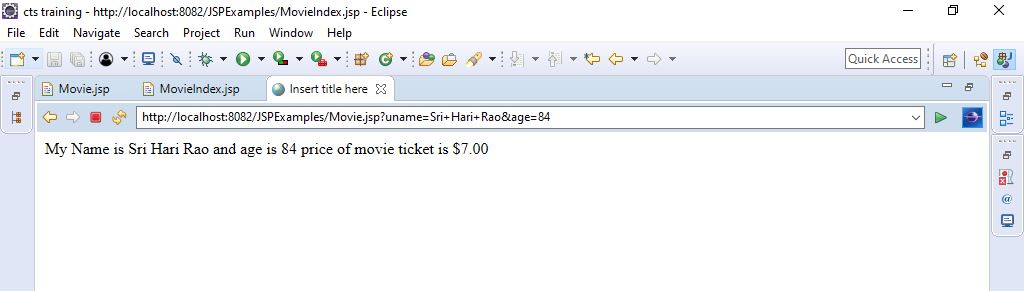
</form>

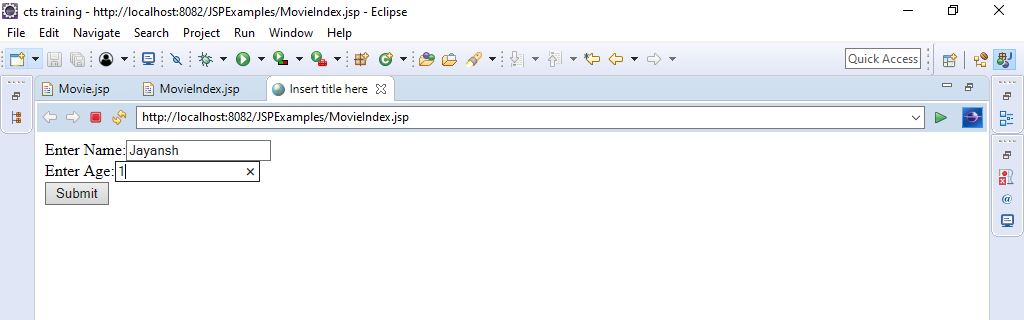
</body>

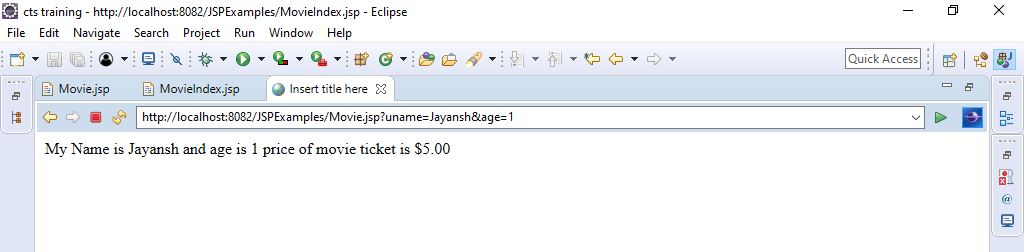
</html>

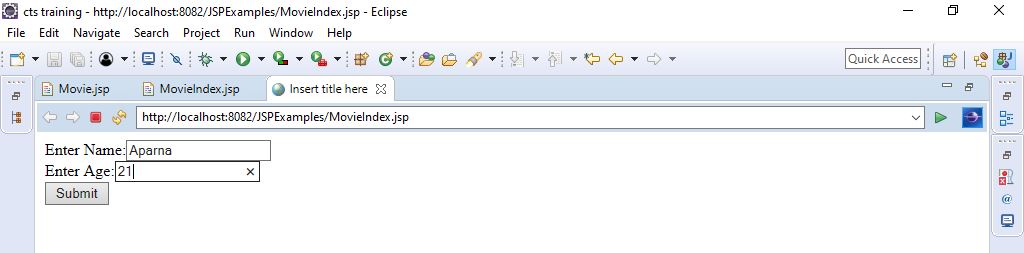
Output –

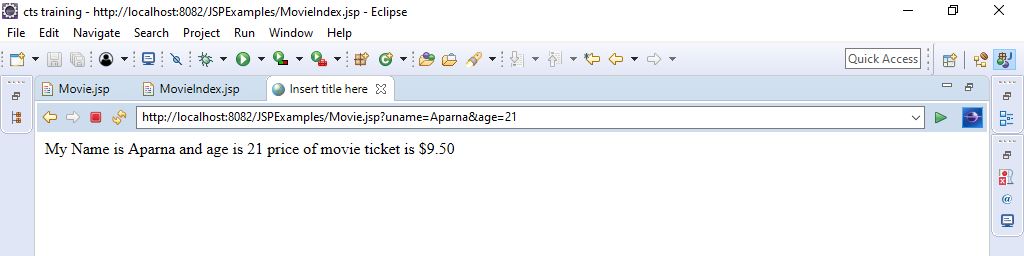












3. Write a spring program which will demonstrate the spring life cycle bean post processor methods.

Welcome.java –

**package** exam;

**public** **class** Welcome {

**private** String message;

**public** **void** setMessage(String message){

**this**.message=message;

}

**public** **void** getMessage(){

System.***out***.println("Your Message : "+ message);

}

**public** **void** init(){

System.***out***.println("Bean is going through init");

}

**public** **void** destroy(){

System.***out***.println("Bean will destroy now");

}

}

InitWelcome.java –

package exam;

import org.springframework.beans.factory.config.BeanPostProcessor;

import org.springframework.beans.BeansException;

public class InitWelcome implements BeanPostProcessor

{

public Object postProcessBeforeInitialization(Object bean, String beanName)

throws BeansException

{

System.out.println("Before Initialization : " + beanName);

return bean;

}

public Object postProcessAfterInitialization(Object bean, String beanName)

throws BeansException

{

System.out.println("After Initialization : " + beanName);

return bean;

}

}

Welcome.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=*"welcome"* class=*"exam.Welcome"*

init-method=*"init"* destroy-method=*"destroy"*>

<property name=*"message"* value=*"Welcome"*/>

</bean>

<bean class=*"exam.InitWelcome"* />

</beans>

Main.java –

**package** exam;

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

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** Main

{

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

AbstractApplicationContext context = **new** ClassPathXmlApplicationContext("welcome.xml");

Welcome a = (Welcome) context.getBean("welcome");

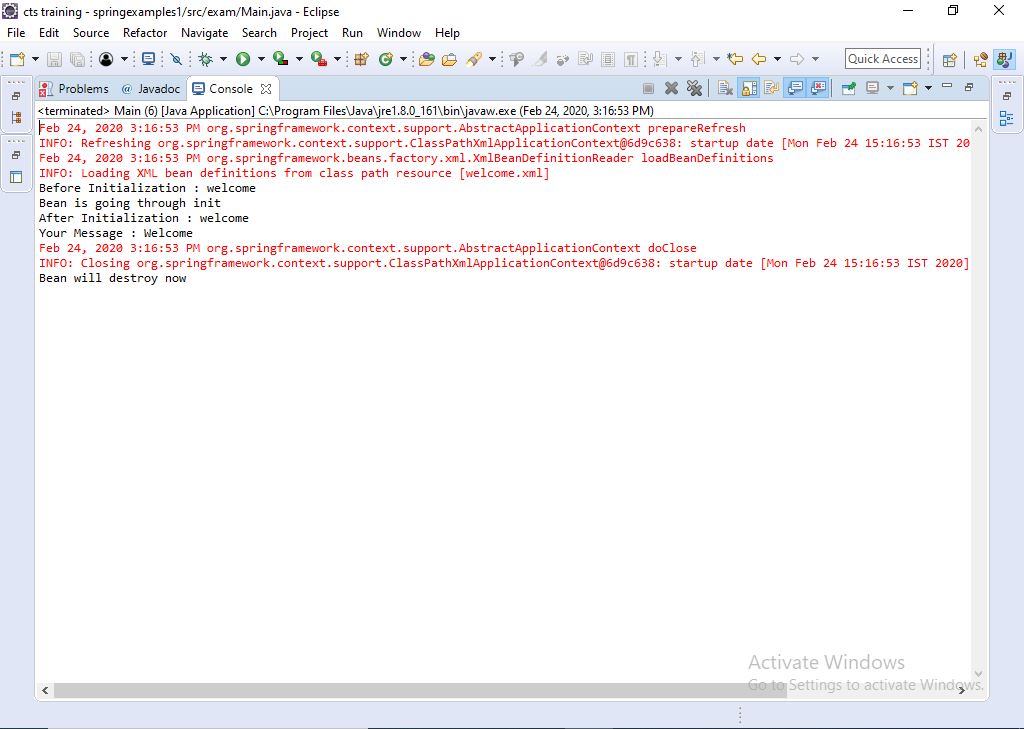
a.getMessage();

context.registerShutdownHook();

}

}

Output –



4. Write a simple spring program to implement Dependency injection using constructor method for dependent objects and Map objects.

Mapping.java -

**package** exam;

**import** java.util.Map;

**public** **class** Mapping

{

**int** qid;

String question;

Map<String,String> answers;

**public** Mapping(**int** qid,String question,Map<String,String>answers)

{

**this**.qid=qid;

**this**.question=question;

**this**.answers=answers;

}

**public** **int** getQid()

{

**return** qid;

}

**public** **void** setQid(**int** qid)

{

**this**.qid = qid;

}

**public** String getQuestion()

{

**return** question;

}

**public** **void** setQuestion(String question)

{

**this**.question = question;

}

**public** Map<String, String> getAnswers()

{

**return** answers;

}

**public** **void** setAnswers(Map<String, String> answers)

{

**this**.answers = answers;

}

**public** **void** showAnswers()

{

System.***out***.println(qid+question);

**for** (Map.Entry<String,String> entry : answers.entrySet())

System.***out***.println(entry.getKey() + entry.getValue());

}

}

ApplicationContext.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=*"que"* class=*"exam.Mapping"*>

<constructor-arg value=*"1 "*/>

<constructor-arg value=*"What is the full form of JDBC?"*/>

<constructor-arg>

<map>

<entry key=*"JDBC stands for Java DataBase Connectivity. "* value=*"A1"*></entry>

<entry key=*"JDBC is a Java API to connect and execute the query with the database. "* value=*"A2"*></entry>

</map>

</constructor-arg>

</bean>

</beans>

TestMap.java-

**package** exam;

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

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** TestMain

{

**public** **static** **void** main(String[] args)

{

ApplicationContext cont=**new** ClassPathXmlApplicationContext("ApplicationContext.xml");

Mapping qump=(Mapping)cont.getBean("que");

qump.showAnswers();

}

}

Person.java-

**package** exam;

**import** springexamples1.Address;

**public** **class** Person {

String name;

**int** age;

Address address;

**public** Person(String name,**int** age,Address address){

**this**.name=name;

**this**.age=age;

**this**.address=address;

}

**public** **void** show() {

System.***out***.println(name);

System.***out***.println(age);

System.***out***.println(address);

}

}

Address.java –

**package** exam;

**public** **class** Address {

String street;

String area;

String city;

**public** Address(String street, String area, String city) {

**super**();

**this**.street = street;

**this**.area = area;

**this**.city = city;

}

}

TestDependent.java –

package exam;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class TestDependent {

public static void main(String[] args) {

// TODO Auto-generated method stub

ApplicationContext cont=new ClassPathXmlApplicationContext("Dependent-bean.xml");

Person per=(Person)cont.getBean("per");

per.show();

}

}

Dependent-bean.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=*"addr"* class=*"spring.Address"*>

<constructor-arg value=*"IndiraNagar "*/>

<constructor-arg value=*"Gachibowli"*/>

<constructor-arg value=*"Hyderabad"*/>

</bean>

<bean id=*"per"* class=*"spring.Person"* autowire=*"constructor"*>

<constructor-arg value=*"Apparna"*/>

<constructor-arg value=*"22"*/>

<constructor-arg>

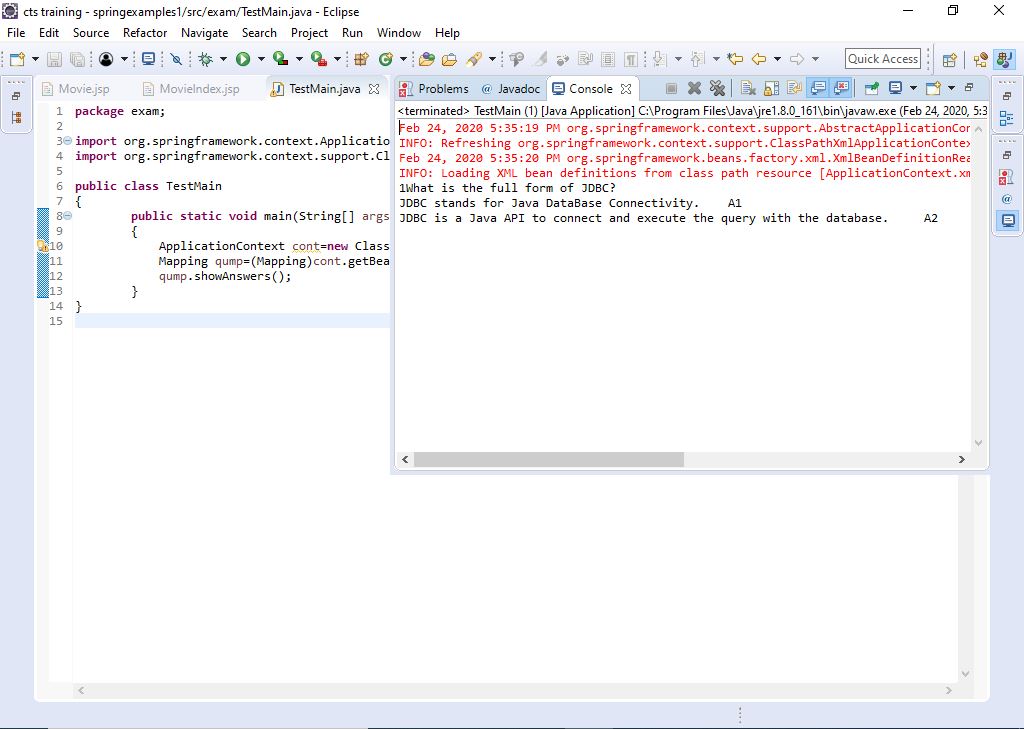
<ref bean=*"addr"*/>

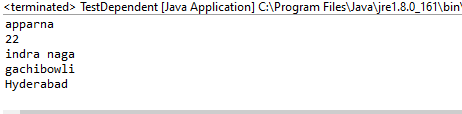
</constructor-arg>

</bean>

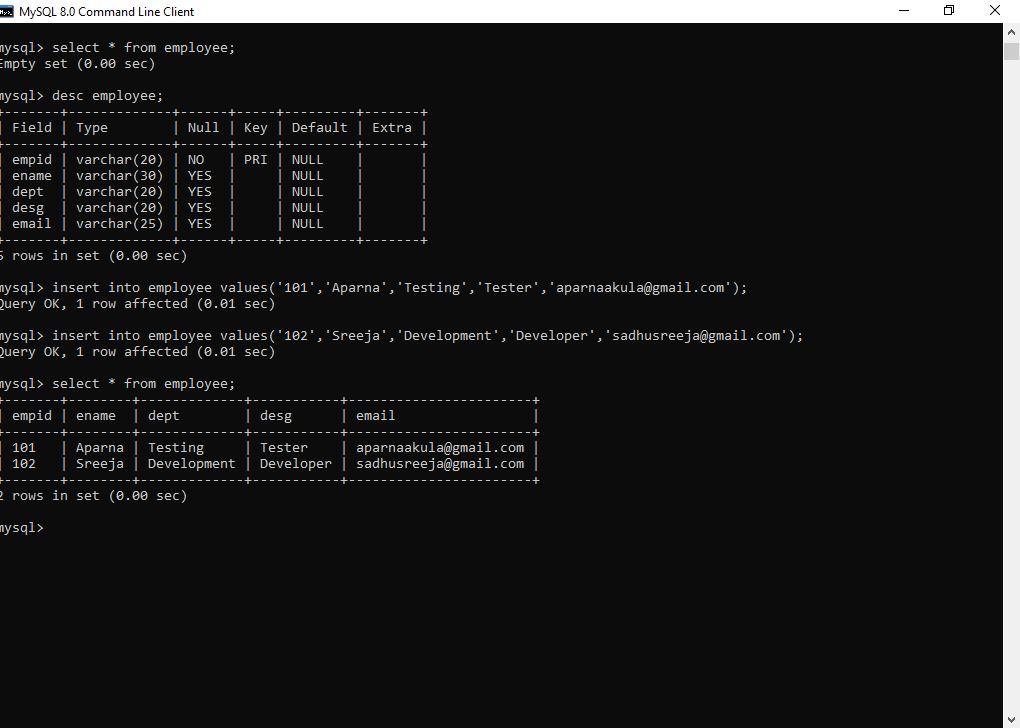
</beans>

Output-





5. Write a spring jdbc program to display all the records from any table from the mysql database.



Employee.java –

**package** jdbcprograms;

**public** **class** Employee

{

String empid;

String ename;

String dept;

String desg;

String email;

**public** Employee()

{

}

**public** Employee(String string, String string2, String string3, String string4) {

// **TODO** Auto-generated constructor stub

}

**public** String getEmpid() {

**return** empid;

}

**public** **void** setEmpid(String empid) {

**this**.empid = empid;

}

**public** String getEname() {

**return** ename;

}

**public** **void** setEname(String ename) {

**this**.ename = ename;

}

**public** String getDept() {

**return** dept;

}

**public** **void** setDept(String dept) {

**this**.dept = dept;

}

**public** String getDesg() {

**return** desg;

}

**public** **void** setDesg(String desg) {

**this**.desg = desg;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

}

EmployeeDao.java –

**package** jdbcprograms;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**public** **class** EmployeeDao

{

**private** JdbcTemplate jdbcTemplate;

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate)

{

**this**.jdbcTemplate=jdbcTemplate;

}

**public** List<Employee> getEmployee()

{

String sql="select \* from employee";

List<Employee> list=jdbcTemplate.query(sql, **new** RowMapper<Employee>()

{

**public** Employee mapRow(ResultSet rs, **int** rowNum) **throws** SQLException

{

Employee e=**new** Employee();

e.setEmpid(rs.getString("empid"));

e.setEname(rs.getString("ename"));

e.setDept(rs.getString("dept"));

e.setDesg(rs.getString("desg"));

e.setEmail(rs.getString("email"));

**return** e;

}

});

**return** list;

}

}

applicationContext.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=*"ds"* class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.cj.jdbc.Driver"* />

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/cts"* />

<property name=*"username"* value=*"root"* />

<property name=*"password"* value=*"root"* />

</bean>

<bean id=*"jdbcTemplate"* class=*"org.springframework.jdbc.core.JdbcTemplate"*>

<property name=*"dataSource"* ref=*"ds"* />

</bean>

<bean id=*"edao"* class=*"jdbcprograms.EmployeeDao"* >

<property name=*"jdbcTemplate"* ref=*"jdbcTemplate"* />

</bean>

</beans>

CustomMain.java –

package jdbcprograms;

import java.util.Iterator;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class CustomMain

{

public static void main(String[] args)

{

// TODO Auto-generated method stub

ApplicationContext ctx=new ClassPathXmlApplicationContext("applicationContext.xml");

EmployeeDao dao=(EmployeeDao)ctx.getBean("edao");

List<Employee> employee =dao.getEmployee();

Iterator<Employee> itr=employee.iterator();

for(Employee e:employee)

{

System.out.println(e.getEmpid()+" " +e.getEname()+" "+e.getDept()+" "+e.getDesg()+" "+e.getEmail());

}

dao.getEmployee();

}

}

Output –

