Name: Rishi Yadav
Roll no: 238
Class: D
Subject: Java for Software Development
Module 3: Assignment 3

Practical 1

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

Code:

Source Code:

FullName.java

```
package com.spring;
public class FullName {

	private String firstName;
	private String lastName;

	public FullName() {

	}
	public FullName(String firstName, String lastName)
{
		super();
		this.firstName = firstName;
		this.lastName = lastName;
	}
	public String getFirstName() {
		return firstName;
	}
	public String getLastName() {
		return lastName;
	}
	@Override
	public String toString() {
		return "FullName [firstName=" + firstName + ",
lastName=" + lastName + "]";
	}
}
```

Employee.java

```
package com.spring;
public class Employee {
	private String id;
	private String name;
	private FullName fullName;

	public Employee() {
	}

	public Employee(String id, String name, FullName
fullName) {
		super();
		this.id = id;
		this.name = name;
		this.fullName = fullName;
	}
	public String getId() {
		return id;
	}

	public String getName() {
		return name;
	}
	public FullName getFullName() {
		return fullName;
	}
	@Override
	public String toString() {
		return "Employee [id=" + id + ", name=" + name
+ ", fullName=" + fullName + "]";
	}
}
```

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:p="http://www.springframework.org/schema/p"
xsi:schemaLocation="http://www.springframework.org/sche
ma/beans
http://www.springframework.org/schema/beans/spring-
beans-3.0.xsd"
>
<bean id="fullName" class="com.spring.FullName">
<constructor-arg index="0" value="Sam"></constructor-
arg>
<constructor-arg index="1" value="Doe"></constructor-
arg>
</bean>
<bean id="emp" class="com.spring.Employee">
<constructor-arg index="0" value="1"></constructor-arg>
<constructor-arg index="1" value="Sam"></constructor-
arg>
<constructor-arg><ref
bean="fullName"></ref></constructor-arg>
</bean>
</beans>
```

App.java

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

```
        Employee e = (Employee)
context.getBean("emp",Employee.class);
        System.out.println(e);

    }
}
```

Output:

```
Employee [id=1, name=John, fullName=FullName [firstName=John, lastName=Yadav]]
```

Practical 2

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

Code:

Source Code:

AccountHolder.java

```
package com.spring;
public class AccountHolder {
	private String accNo;
	private String accName;
	public AccountHolder() {
		super();
		// TODO Auto-generated constructor stub
	}
	public AccountHolder(String accNo, String accName) {
		super();
		this.accNo = accNo;
		this.accName = accName;
	}
	public String getAccNo() {
		return accNo;
	}
	public void setAccNo(String accNo) {
		this.accNo = accNo;
	}
	public String getAccName() {
		return accName;
	}
	public void setAccName(String accName) {
		this.accName = accName;
	}

}
```

config.xml

```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/schema/beans
http://www.springframework.org/schema/beans/spring-beans-
3.0.xsd"
>
<bean id="accholder" class="com.spring.AccountHolder" >
<property name="accNo" value="808"></property>
<property name="accName" value="John Doe"></property>
</bean>
</beans>
```

App_accountHolder.java

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

		AccountHolder aH =
context.getBean("accholder",AccountHolder.class);
		System.out.println("Account Number: "+aH.getAccNo());
		System.out.println("Account Name: "+aH.getAccName());
	}
}
```

Output: Directory Structure

```
Account Number: 808
Account Name: John Doe
```

Practical 3

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

Code:

Source Code:

FullName.java

```
package com.spring;
import org.springframework.stereotype.Component;
@Component
public class FullName {
    private String firstName;
    private String lastName;
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }


}
```

AppConfig.java

```
package com.spring;
import org.springframework.context.annotation.Bean;
```

```
import
org.springframework.context.annotation.Configuration;
public class AppConfig {

    @Bean
    public FullName fullName() {
        FullName fn = new FullName();
        fn.setFirstName("John");
        fn.setLastName("Doe");
        return fn;
    }

    @Bean
    public Account account() {
        Account acc = new Account();
        acc.setAccNo("103");
        return acc;
    }
}
```

Account.java

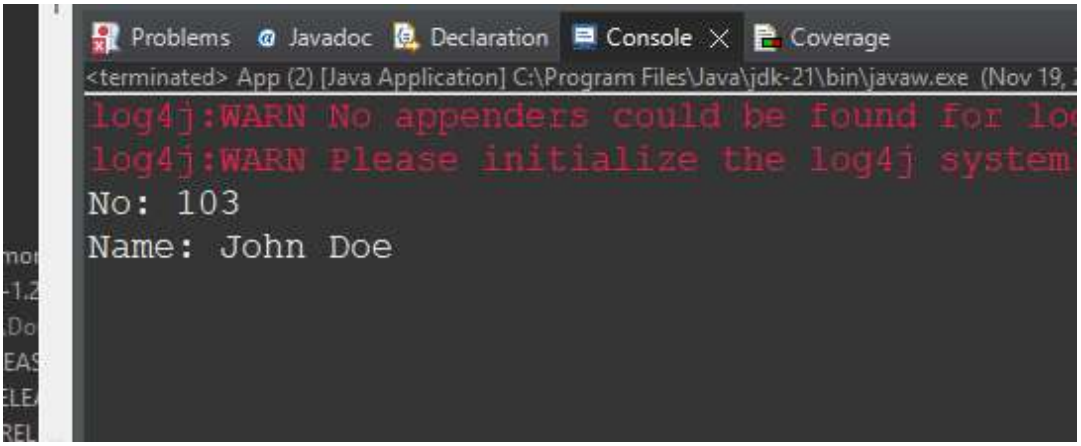
```
package com.spring;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
@Component
public class Account {
    private String accNo;
    private FullName fullName;
    public String getAccNo() {
        return accNo;
    }
    public void setAccNo(String accNo) {
```

```
        this.accNo = accNo;
    }
    public FullName getFullName() {
        return fullName;
    }
    @Autowired
    public void setFullName(FullName fullName) {
        this.fullName = fullName;
    }

    public void display() {
        System.out.println("No: "+accNo);
        System.out.println("Name:
"+fullName.getFirstName()+" "+fullName.getLastName());
    }

}
```

App.java

```
package com.spring;
import org.springframework.context.ApplicationContext;
import
org.springframework.context.annotation.AnnotationConfig
ApplicationContext;
public class App {
    public static void main(String[] args){
        ApplicationContext context =new
AnnotationConfigApplicationContext(AppConfig.class);
        Account ac = context.getBean(Account.class);
        ac.display();

    }
}
```

Output:


Problems
Javadoc
Declaration
Console
Coverage
<terminated> App (2) [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Nov 19,
log4j:WARN No appenders could be found for lo
log4j:WARN Please initialize the log4j system
No: 103
Name: John Doe