Spring Framework

1. Write a program to print “Hello World” using spring framework.
2. Write a program to demonstrate dependency injection via setter method.
3. Write a program to demonstrate dependency injection via Constructor.

Problem Statement 1 : Write a program to print “Hello World” using spring framework.

Solution :

HelloWorld.java

```
package spring1;
public class HelloWorld {
  String name;

public String getName() {
        return name;
}
public void setName(String name) {
        this.name = name;
}
@Override
public String toString() {
        return "HelloWorld [name=" + name + "]";
}
}
```

appctx3.xml

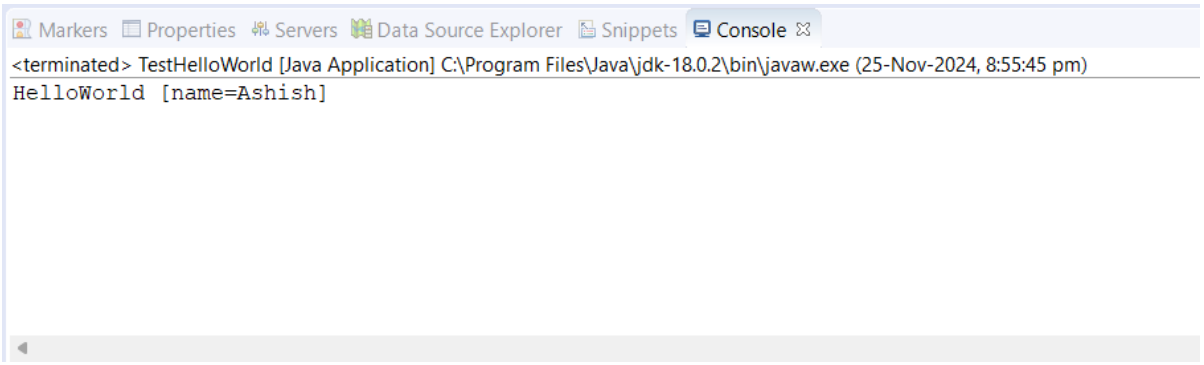
```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
```

```
<bean id="hw" class="spring1.HelloWorld">
        <property name="name" value="Ashish"/>
    </bean> </beans>
```

TestHelloWorld.java

```
package spring1;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class TestHelloWorld {


    public static void main(String[] args) {

        ClassPathXmlApplicationContext app = new
ClassPathXmlApplicationContext("appctx3.xml");


        HelloWorld hw = (HelloWorld) app.getBean("hw");


        System.out.println(hw.toString());

    }

}
```

Output:



Problem Statement 2 : Write a program to demonstrate dependency injection via setter method.

Solution:

Account.java

```
package spring2;

public class Account {
	int id;
	String name;
	int balance;
	public Account(int id, String name, int balance) {
		super();
		this.id = id;
		this.name = name;
		this.balance = balance;
	}
	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 getBalance() {
		return balance;
	}
```

```
	public void setBalance(int balance) {
		this.balance = balance;
	}
	@Override
	public String toString() {
		return "Account [id=" + id + ", name=" + name + ", balance=" + balance + "]";
	}
}
```

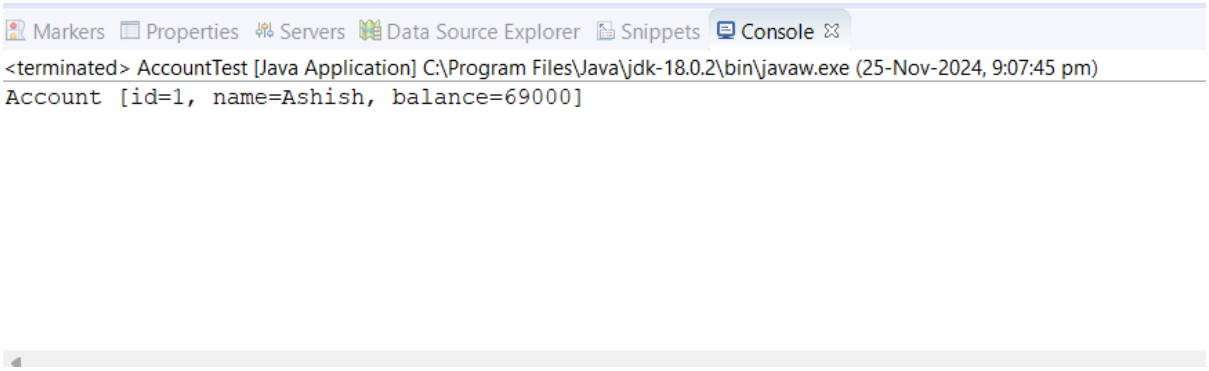
appctx2.xml

```
<?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.xsd">
<bean id="Account" class="spring2.Account">
	<constructor-arg name="id" value="1"></constructor-arg>
	<constructor-arg name="name" value="Ashish"></constructor-arg>
	<constructor-arg name="balance" value="69000"></constructor-arg>
</bean>
</beans>
```

AccountTest.java

```
package spring2;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class AccountTest {
	public static void main(String[] args) {
		ApplicationContext con = new ClassPathXmlApplicationContext("appctx2.xml");
		Account acc = (Account) con.getBean("Account");
		System.out.println(acc.toString());
	}
}
```

Output:



Problem Statement 3 : Write a program to demonstrate dependency injection via Constructor.

Solution:

Singer.java

```
package spring1;

public class Singer {

String name;

 int age;

 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;

}

 void displayInfo() {

        System.out.println("Name:" +name+" Age:" +age);

 }

}
```

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"

        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="Singer" class="spring1.Singer">

<property name="name" value="Ashish"></property>

<property name="age" value="21"></property>

</bean>

</beans>
```

SingerTest.java

```
package spring1;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SingerTest {

        private static ApplicationContext ctx;

        public static void main(String[] args) {

                ctx=new ClassPathXmlApplicationContext("appctx.xml");

                Singer singer=(Singer)ctx.getBean("Singer");

                        singer.displayInfo();

        }

}
```

Output: