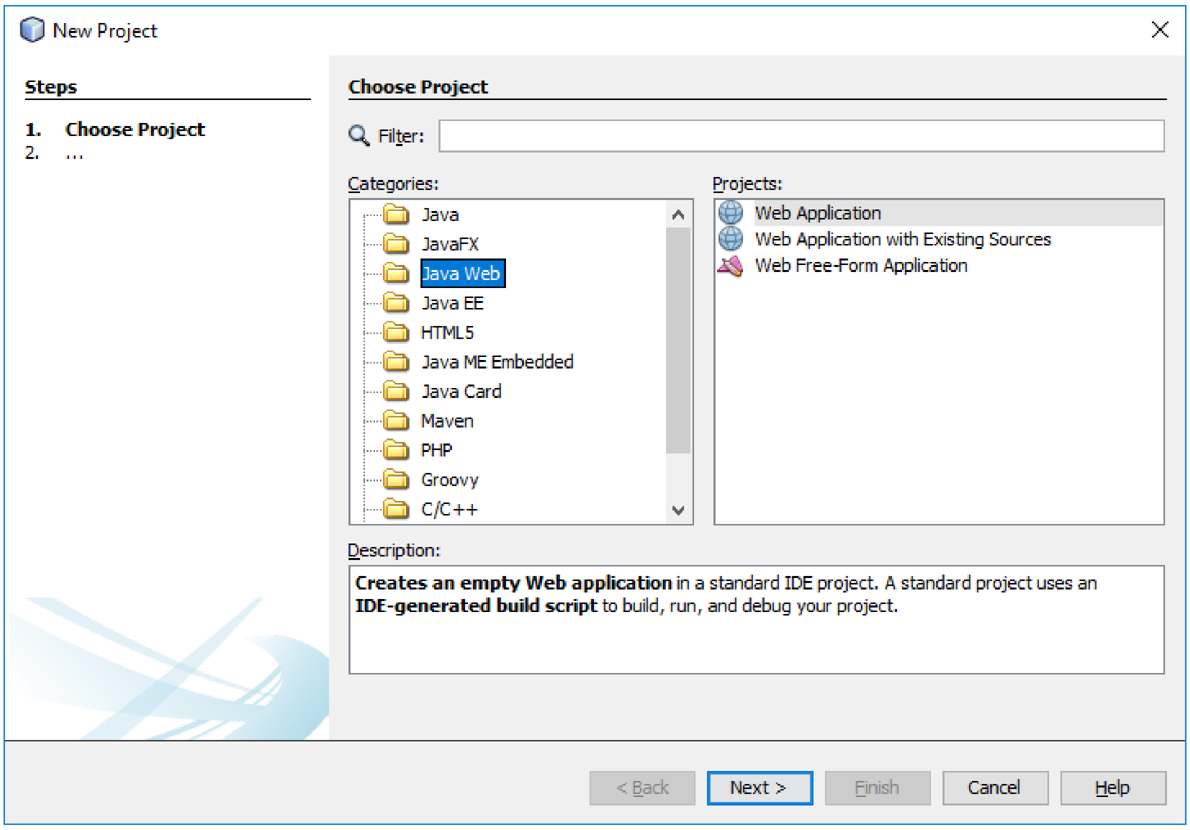
PRACTICAL 1

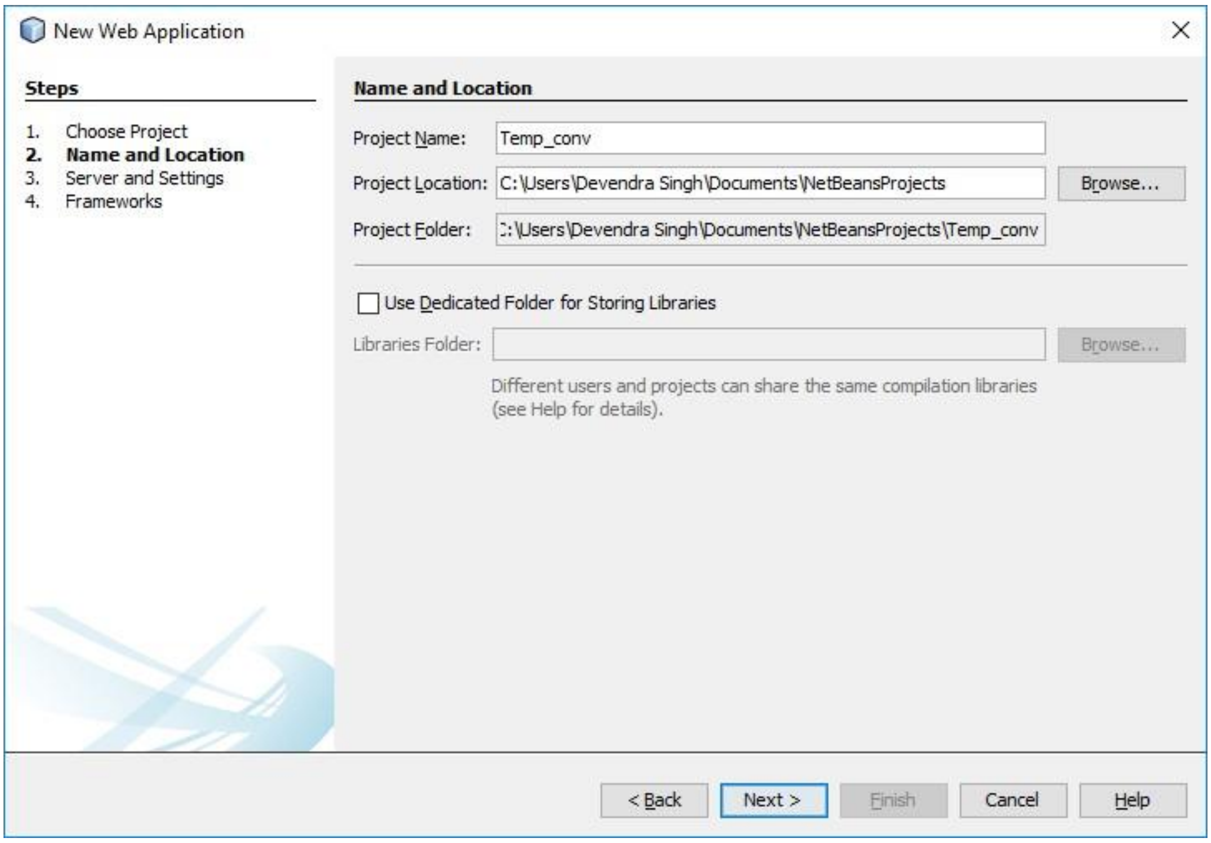
Q1. Write a program to implement to create a simple web service that converts the temperature from Fahrenheit to Celsius and vice versa.

1 Go to File -> New Project. Select Java Web in categories and Web Application in Projects. Click on Next to create web based project.



Creates an empty Web application in a standard IDE project. A standard project uses an **IDE-generated build script** to build, run, and debug your project.

2 Enter a project name whatever you want and then click on Next. On next page click Finish.



Name and Location

Project Name: Temp_conv

Project Location: C:\Users\Devendra Singh\Documents\NetBeansProjects

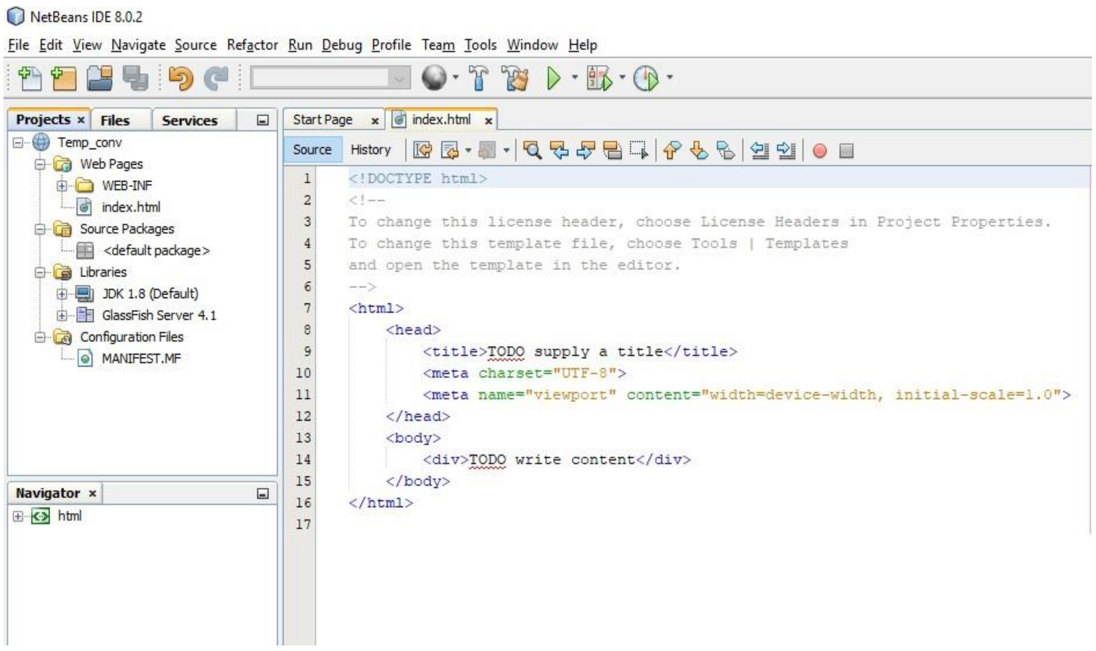
Project Folder: :\Users\Devendra Singh\Documents\NetBeansProjects\Temp_conv

Use Dedicated Folder for Storing Libraries

Libraries Folder:

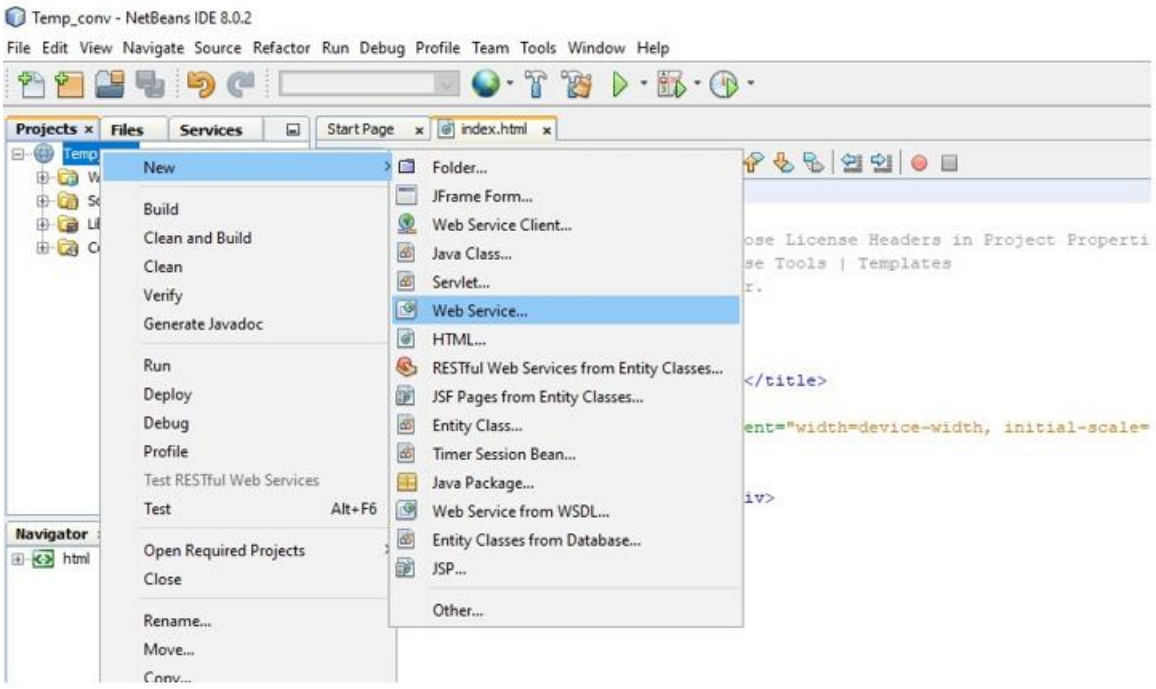
Different users and projects can share the same compilation libraries (see Help for details).

3. After completion of project creation process a window will appear like below.

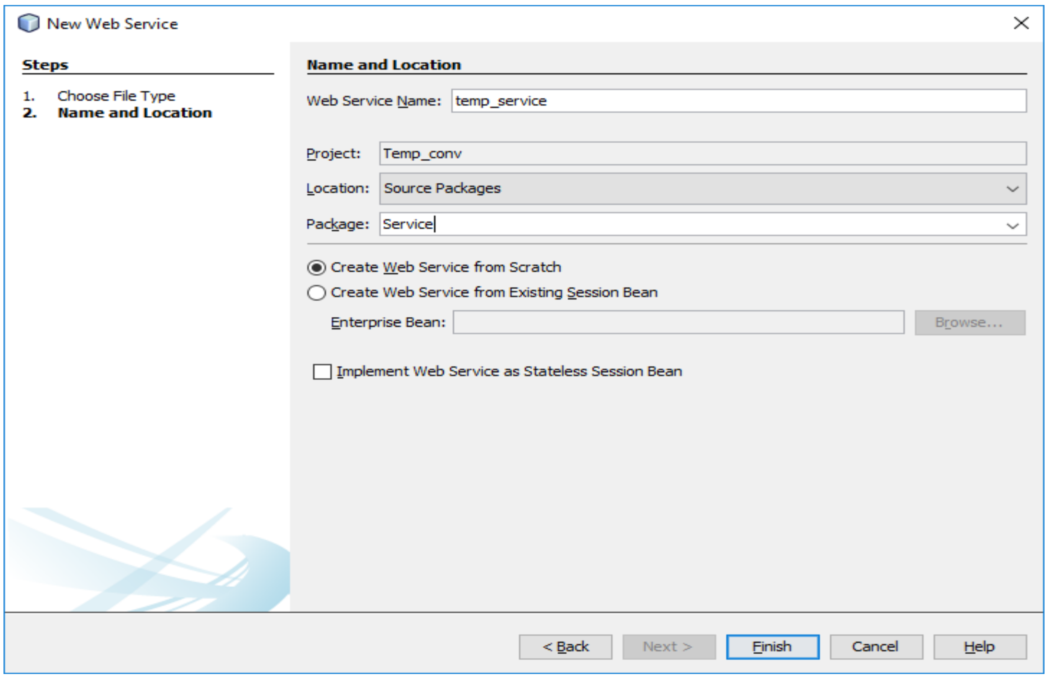


```
<!DOCTYPE html>
<!--
To change this license header, choose License Headers in Project Properties.
To change this template file, choose Tools | Templates
and open the template in the editor.
-->
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <div>TODO write content</div>
    </body>
</html>
```

4. Create web service.
Right click on Project -> New -> Web Service



5. **Enter a Web Service Name and package name** and **then click on Finish** to create a Web Service.



Web Service Name: temp_service

Project: Temp_conv

Location: Source Packages

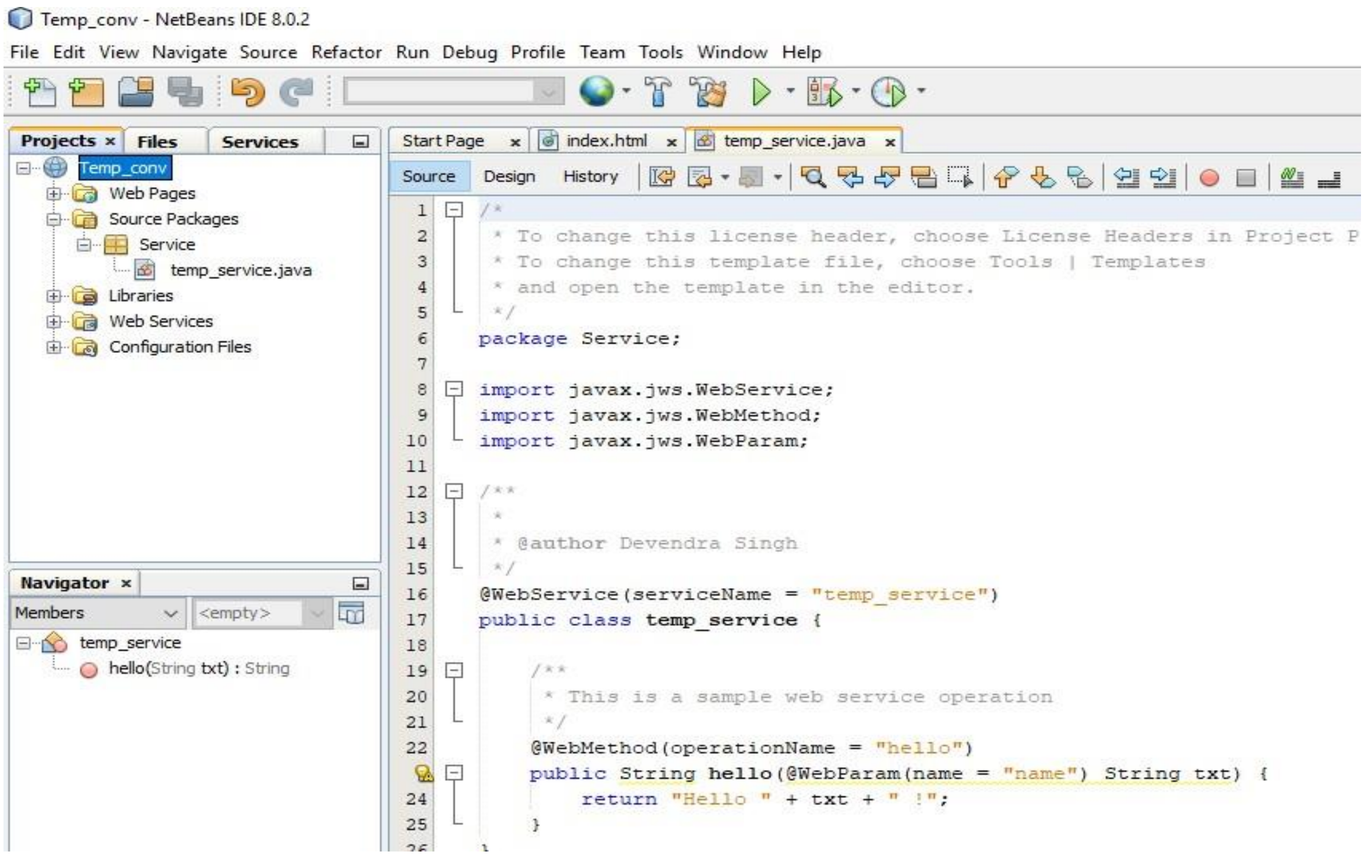
Package: Service

(●) Create Web Service from Scratch
( ) Create Web Service from Existing Session Bean

Enterprise Bean:

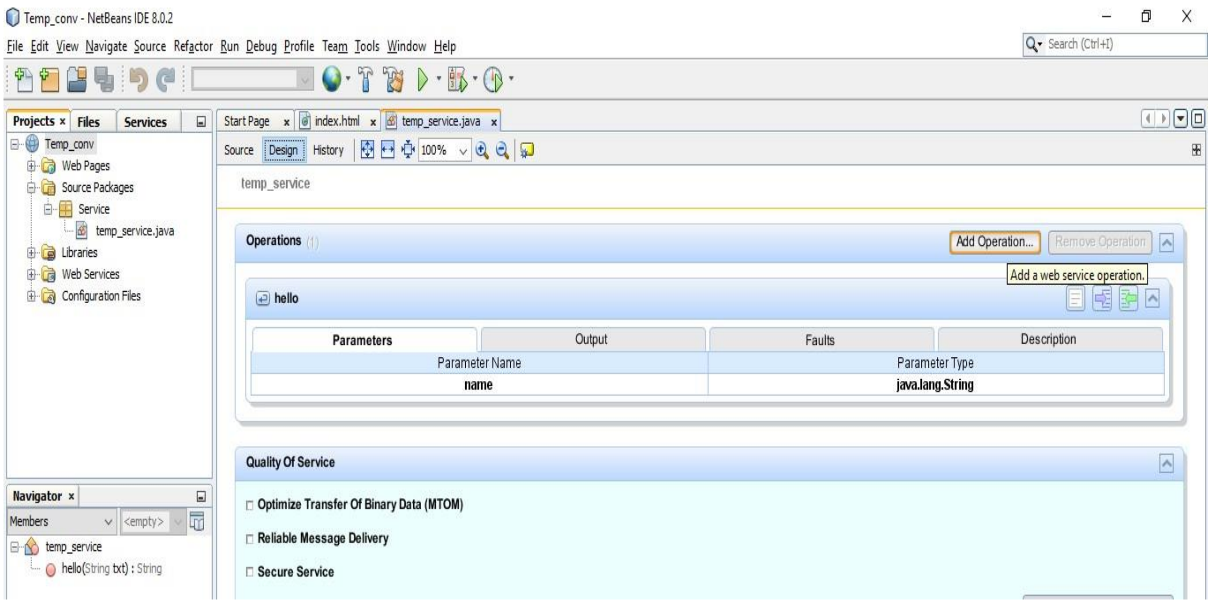
[ ] Implement Web Service as Stateless Session Bean

6. As you can see in following pic; In **Source Packages** there is a package **Service** which contains the service file **temp_service.java**. Open this file by double click on it, So that we can add two operation that will convert temperature from celcious to farhenheit and vice-versa.
Go to design mode by click on **Design**.



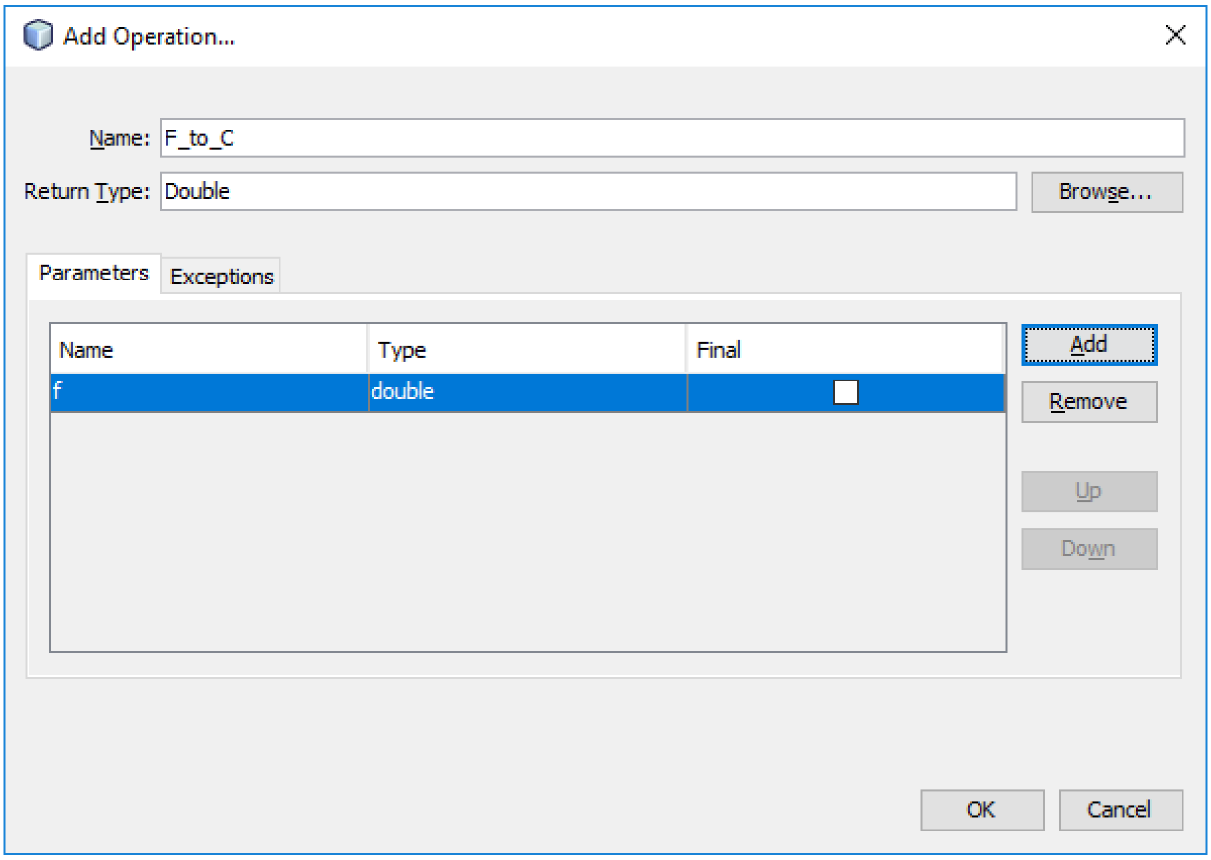
```java
/*
 * To change this license header, choose License Headers in Project P
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Service;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/**
 *
 * @author Devendra Singh
 */
@WebService(serviceName = "temp_service")
public class temp_service {

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "hello")
    public String hello(@WebParam(name = "name") String txt) {
        return "Hello " + txt + " !";
    }
}
```

7. **Click on Add Operation** to add operation.



Parameter Name	Parameter Type
name	java.lang.String

8. Give Operation name **F_to_C** and return type as **Double**. So, this method will return value in Double data type. After that click on **Add** button to give parameters for method. Give its name as **f** and type as **Double** and then click on **OK** button. Your one operation is now successfully created.


Add Operation...

Name: F_to_C

Return Type: Double

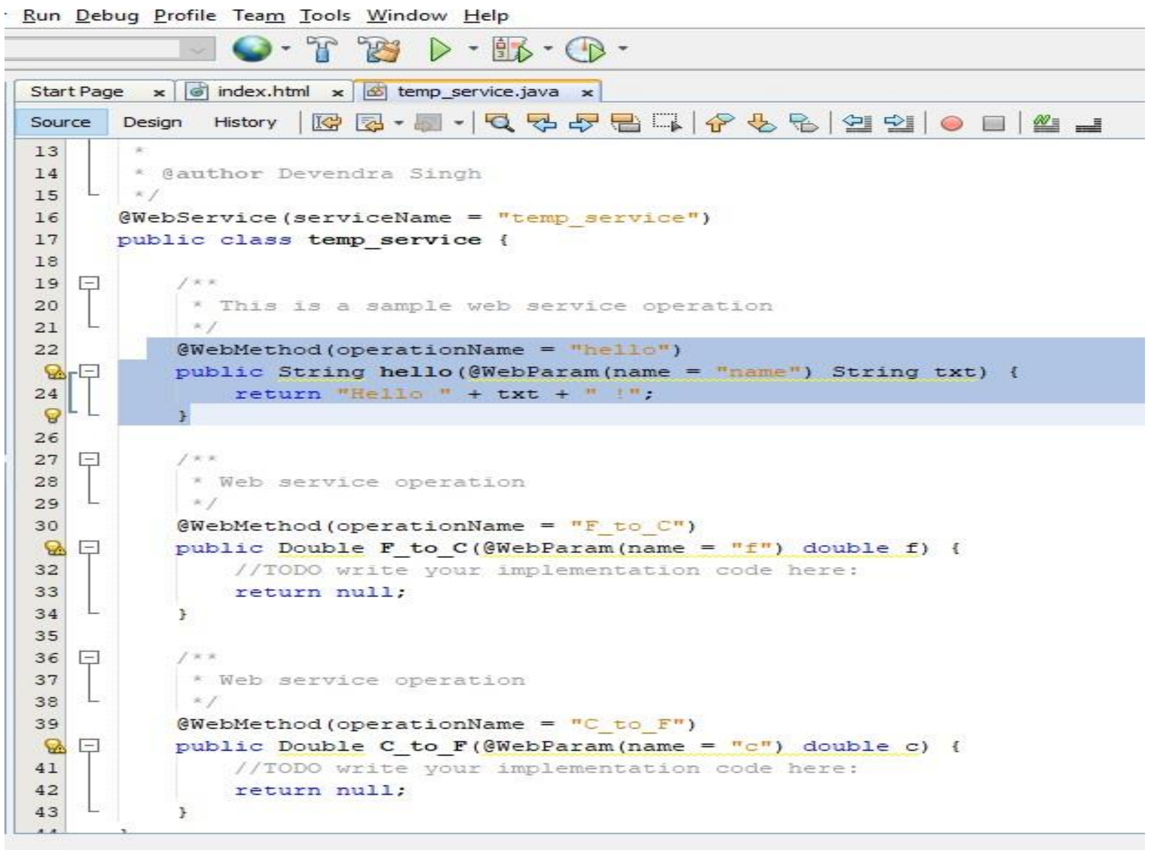
Name	Type	Final
f	double	☐

9. Repeat step 7 & 8 to create second operation. But this time replace some above entered data with following data.

F_to_C -> C_to_F

F -> c

10. Now go to source mode by click on **Source** and **delete the selected** segmentof code. Because it is default operation and we don’t need this.

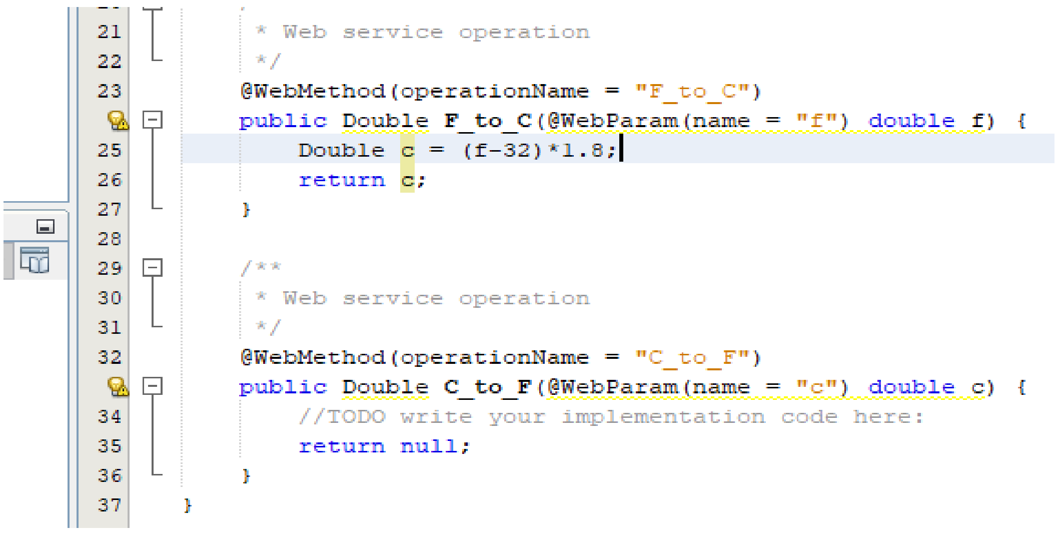


```
 *
 * @author Devendra Singh
 */
@WebService(serviceName = "temp_service")
public class temp_service {

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "hello")
    public String hello(@WebParam(name = "name") String txt) {
        return "Hello " + txt + " !";
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "F_to_C")
    public Double F_to_C(@WebParam(name = "f") double f) {
        //TODO write your implementation code here:
        return null;
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "C_to_F")
    public Double C_to_F(@WebParam(name = "c") double c) {
        //TODO write your implementation code here:
        return null;
    }
```

11. Now replace the selected area with following code to convert Fahrenheit to Celsius.

Double c = (f-32) * 5/9;

return c;

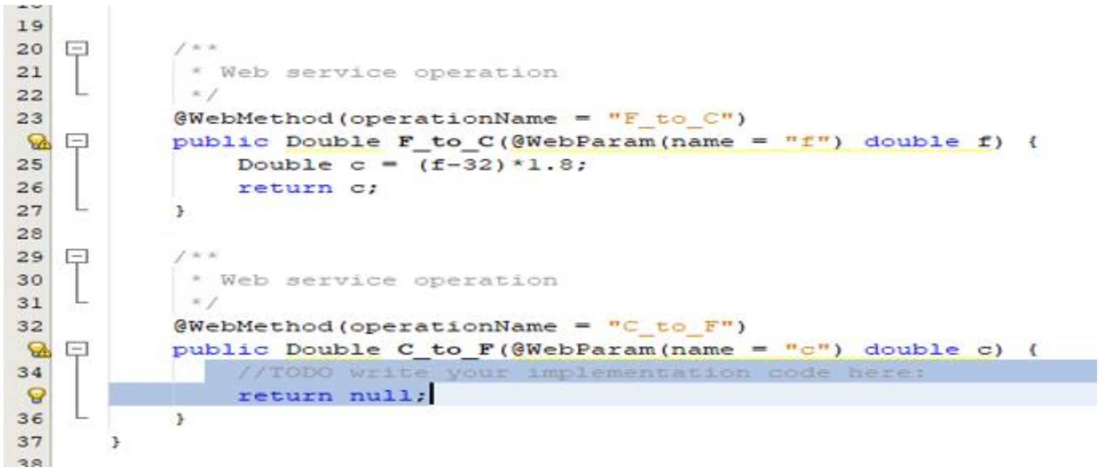


```
         * Web service operation
         */
        @WebMethod(operationName = "F_to_C")
        public Double F_to_C(@WebParam(name = "f") double f) {
            Double c = (f-32)*1.8;
            return c;
        }

        /**
         * Web service operation
         */
        @WebMethod(operationName = "C_to_F")
        public Double C_to_F(@WebParam(name = "c") double c) {
            //TODO write your implementation code here:
            return null;
        }
    }
```

12. Now **replace the selected area with following code** to convert Celsius toFahrenheit and then press Ctrl+S to save.

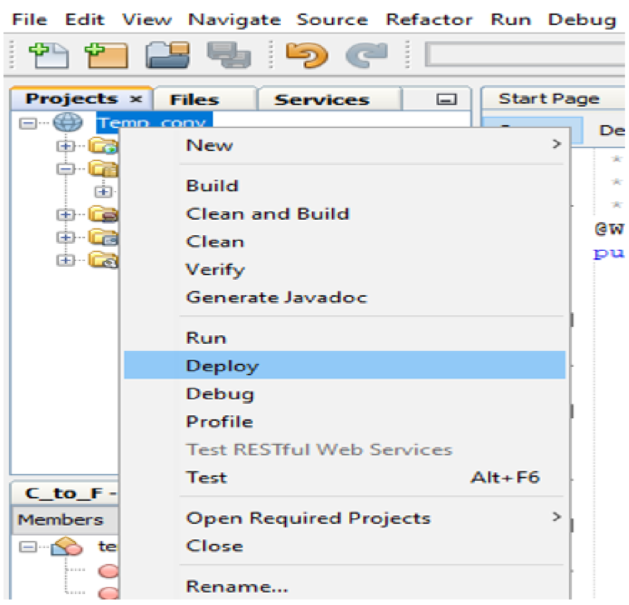
Double f = (c*9/5) + 32;

return f;

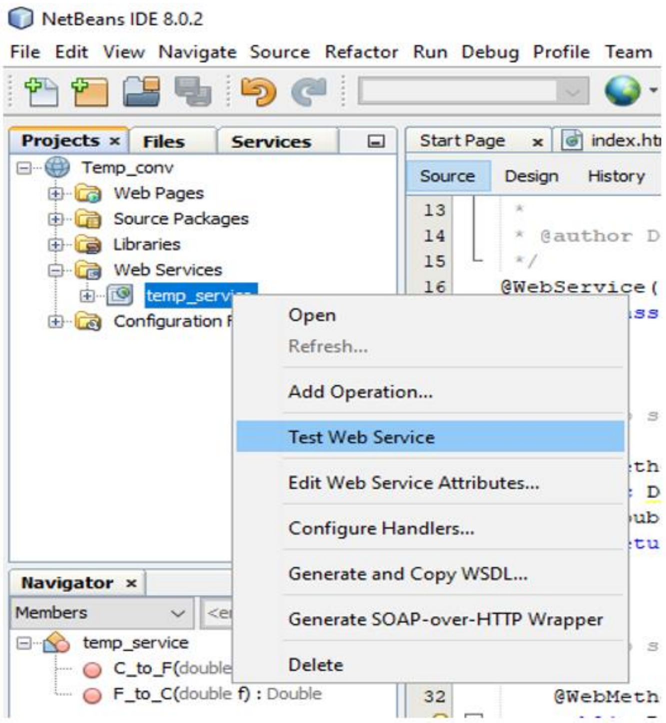


```
        /**
         * Web service operation
         */
        @WebMethod(operationName = "F_to_C")
        public Double F_to_C(@WebParam(name = "f") double f) {
            Double c = (f-32)*1.8;
            return c;
        }

        /**
         * Web service operation
         */
        @WebMethod(operationName = "C_to_F")
        public Double C_to_F(@WebParam(name = "c") double c) {
            //TODO write your implementation code here:
            return null;
        }
    }
```

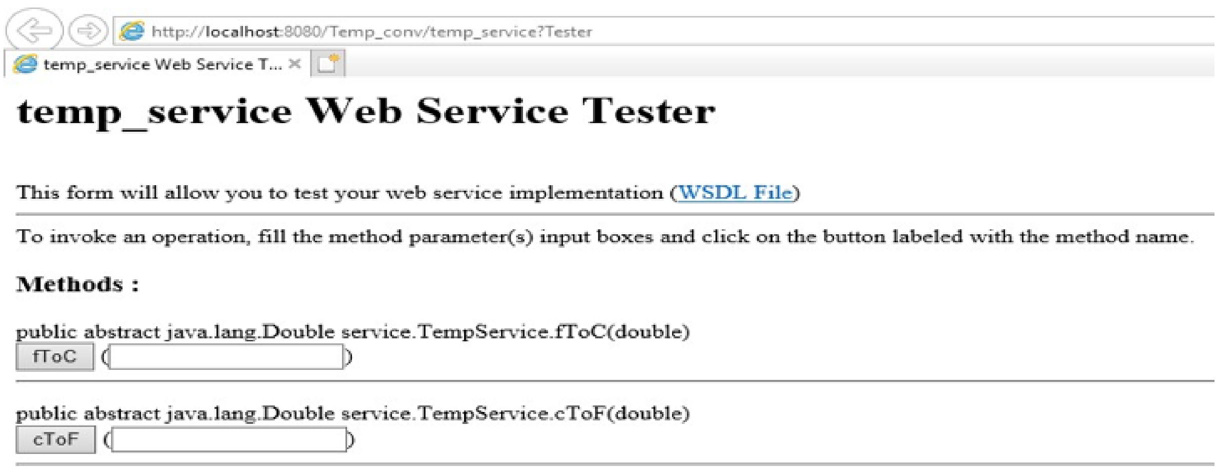
13. Now **right click on project name** and **click on Deploy** to deploy your project.



14. To test your web service follow the following process as in picture.



15. Following window will open in in browser. Now if you will enter a numeric data into first box and you will click on first button it will convert the entered data into Celsius



http://localhost:8080/Temp_conv/temp_service?Tester

temp_service Web Service T...

temp_service Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

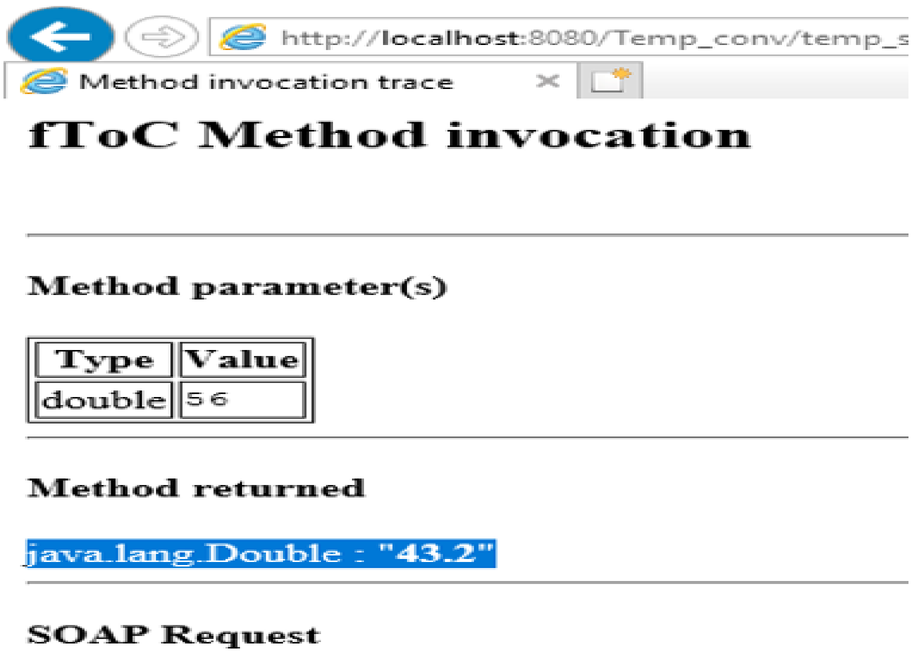
public abstract java.lang.Double service.TempService.fToC(double)

fToC ( )

public abstract java.lang.Double service.TempService.cToF(double)

cToF ( )

16. Selected value in Celsius of 56.



http://localhost:8080/Temp_conv/temp_s

Method invocation trace

fToC Method invocation

Method parameter(s)

Type	Value
double	56

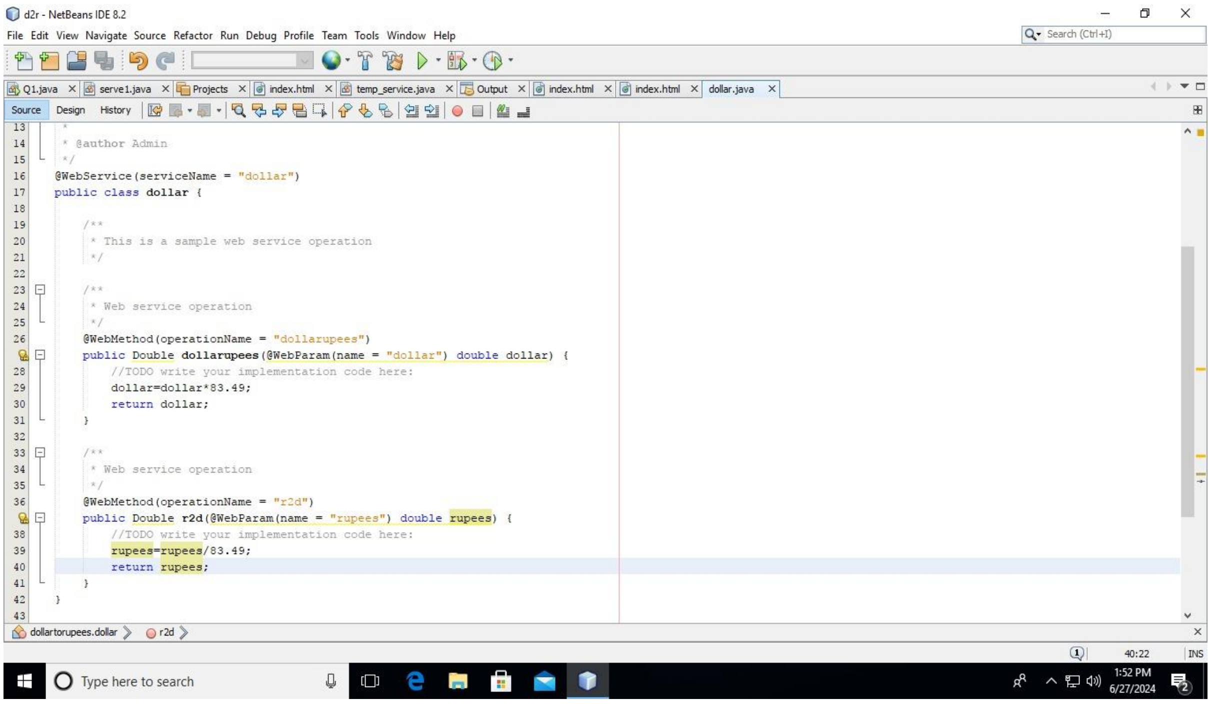
Method returned

java.lang.Double : "43.2"

SOAP Request

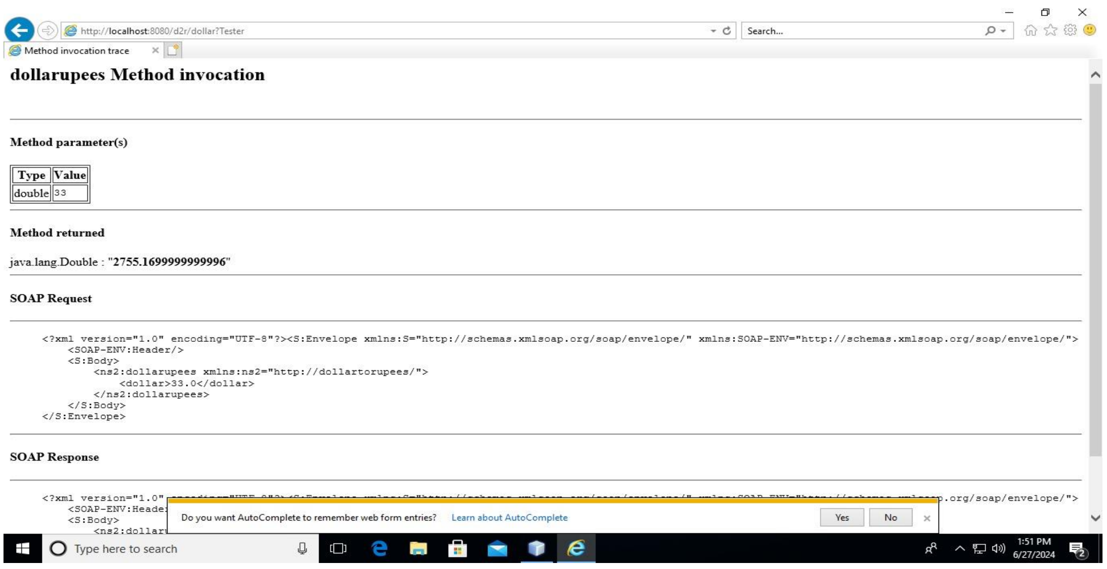
Q2. Write a program to implement to create a simple web service that converts the Revenue from Rupees to Doller and vice versa.

Logic Code:



```
 *
 * @author Admin
 */
@WebService(serviceName = "dollar")
public class dollar {

    /**
     * This is a sample web service operation
     */

    /**
     * Web service operation
     */
    @WebMethod(operationName = "dollarupees")
    public Double dollarupees(@WebParam(name = "dollar") double dollar) {
        //TODO write your implementation code here:
        dollar=dollar*83.49;
        return dollar;
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "r2d")
    public Double r2d(@WebParam(name = "rupees") double rupees) {
        //TODO write your implementation code here:
        rupees=rupees/83.49;
        return rupees;
    }
}
```

Output:



http://localhost:8080/d2r/dollar?Tester

dollarupees Method invocation

Method parameter(s)

Type	Value
double	33

Method returned

java.lang.Double : "**2755.1699999999996**"

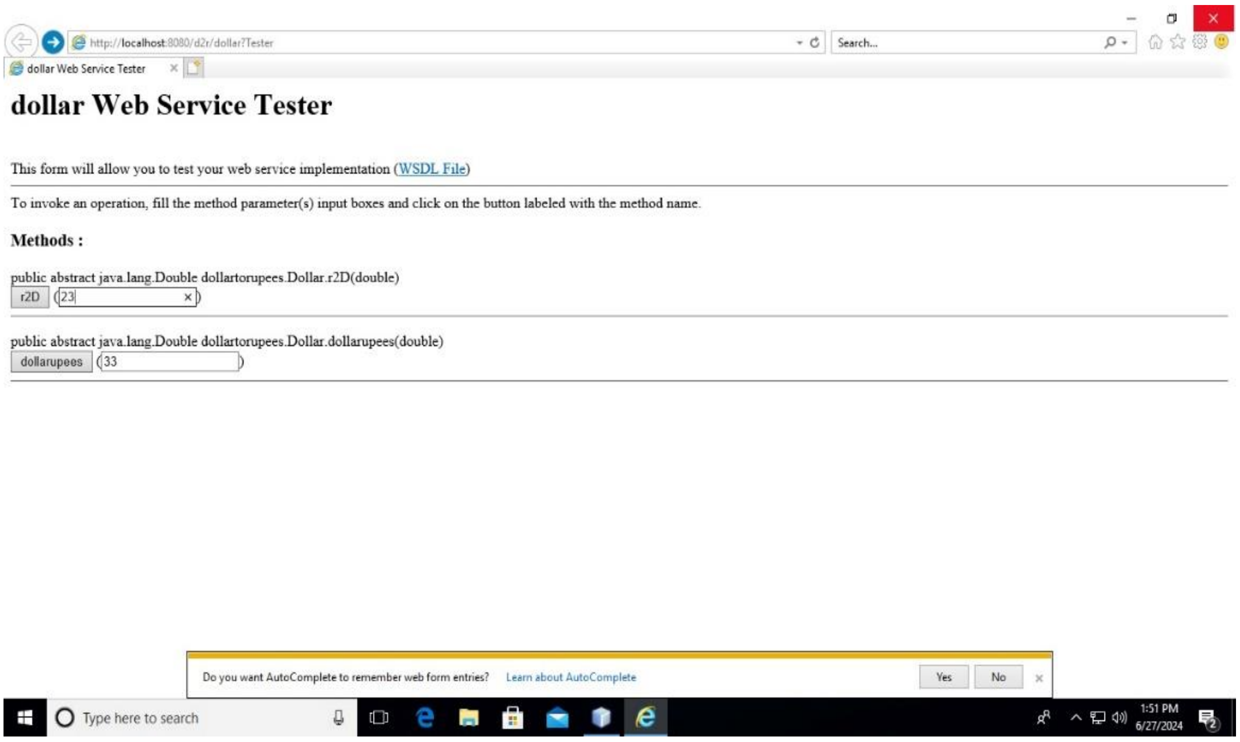
SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:dollarupees xmlns:ns2="http://dollartorupees/">
            <dollar>33.0</dollar>
        </ns2:dollarupees>
    </S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" .org/soap/envelope/">
    <SOAP-ENV:Heade
    <S:Body>
        <ns2:dollar
```

Do you want AutoComplete to remember web form entries?


http://localhost:8080/d2r/dollar?Tester

dollar Web Service Tester

dollar Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

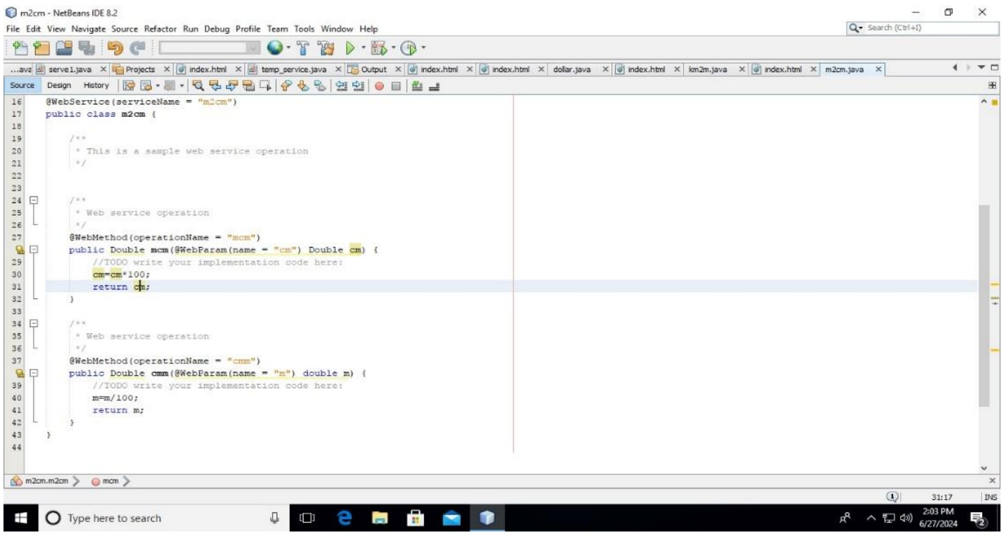
public abstract java.lang.Double dollartorupees.Dollar.r2D(double)
r2D (23)

public abstract java.lang.Double dollartorupees.Dollar.dollarupees(double)
dollarupees (33)

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No

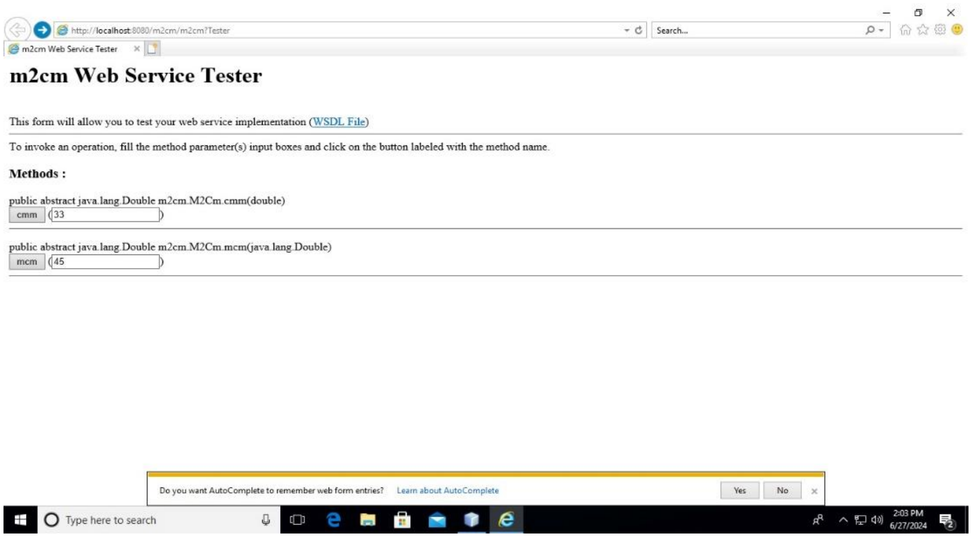
Q3. Write a program to convert the data from centimeter to meter.

Logic Code:



```
@WebService(serviceName = "m2cm")
public class m2cm {

    /**
     * This is a sample web service operation
     */


    /**
     * Web service operation
     */
    @WebMethod(operationName = "mcm")
    public Double mcm(@WebParam(name = "cm") Double cm) {
        //TODO write your implementation code here:
        cm=cm*100;
        return cm;
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "cmm")
    public Double cmm(@WebParam(name = "m") double m) {
        //TODO write your implementation code here:
        m=m/100;
        return m;
    }
}
```

Output:



m2cm Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

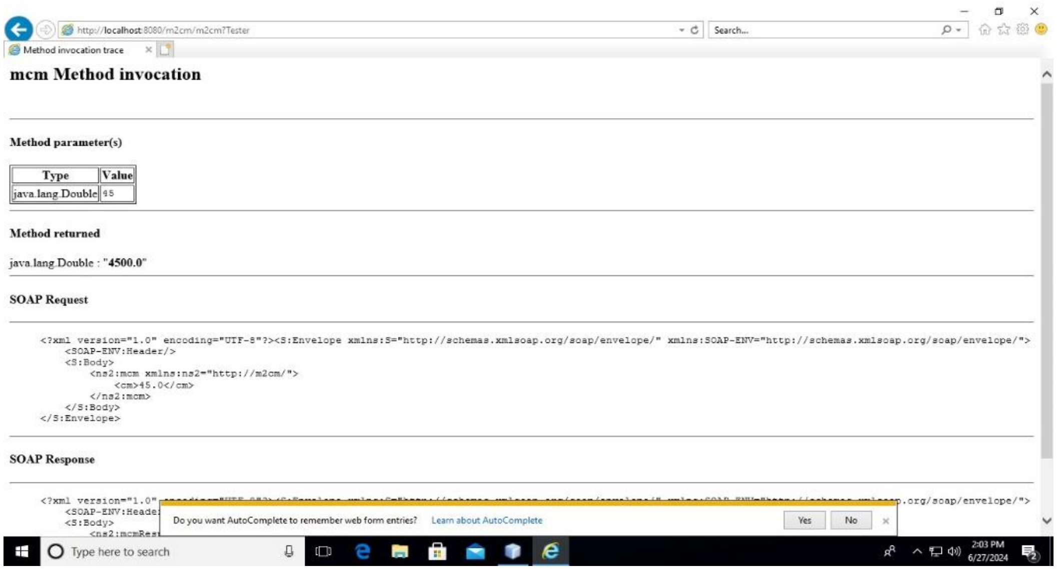
To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

public abstract java.lang.Double m2cm.M2Cm.cmm(double)
cmm (33)

public abstract java.lang.Double m2cm.M2Cm.mcm(java.lang.Double)
mcm (45)

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No



mcm Method invocation

Method parameter(s)

Type	Value
java.lang.Double	45

Method returned

java.lang.Double : "**4500.0**"

SOAP Request

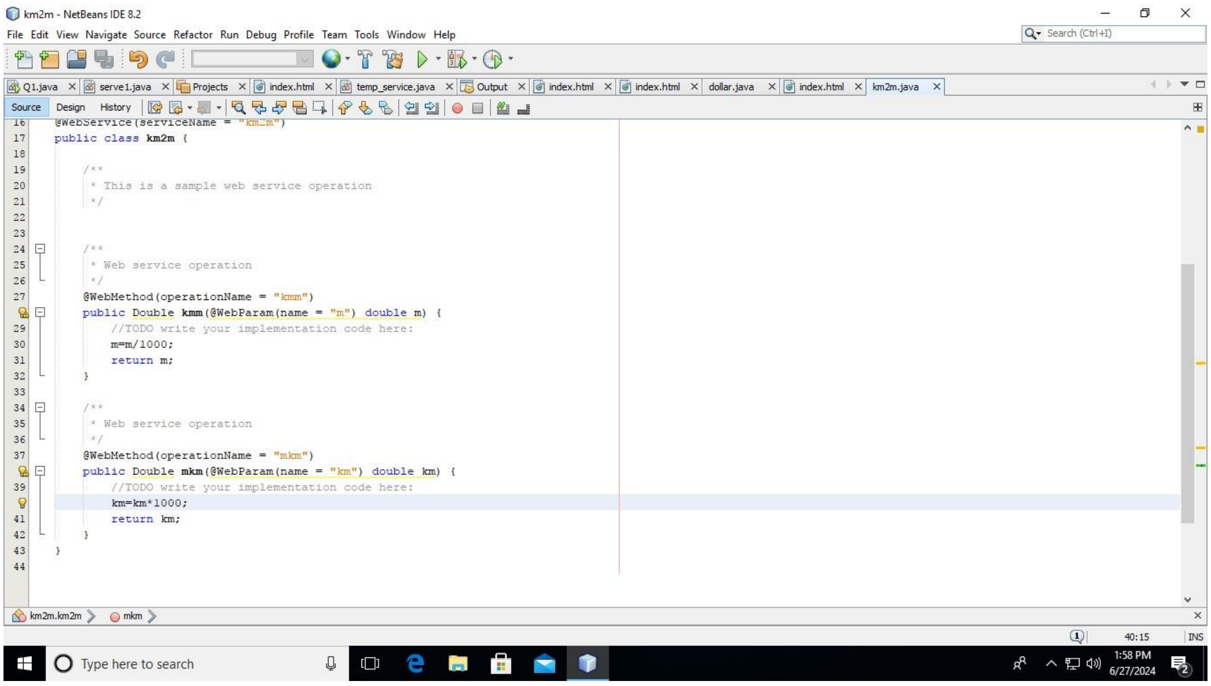
```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:mcm xmlns:ns2="http://m2cm/">
            <cm>45.0</cm>
        </ns2:mcm>
    </S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0"                                                                            .org/soap/envelope/">
    <SOAP-ENV:Heade
    <S:Body>
        <ns2:mcmRes
```

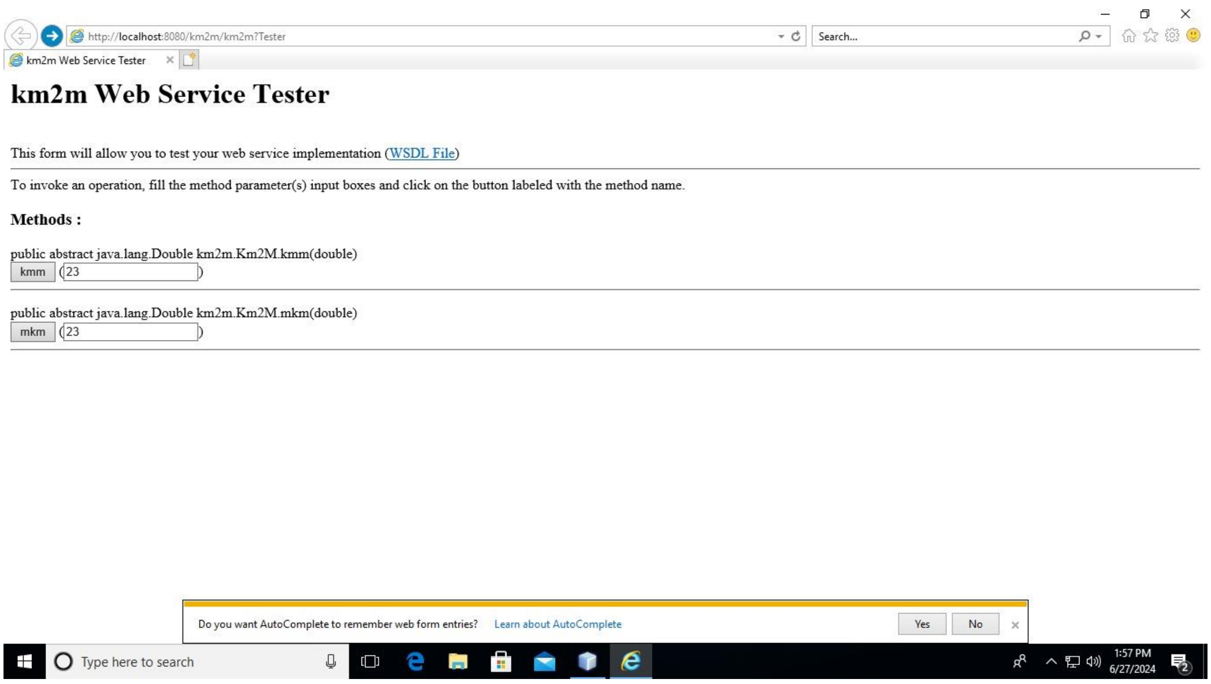
Q4. Write a program to convert the data from kilometer to meter.

Logic Code:



```
@WebService(serviceName = "km2m")
public class km2m {

    /**
     * This is a sample web service operation
     */


    /**
     * Web service operation
     */
    @WebMethod(operationName = "kmm")
    public Double kmm(@WebParam(name = "m") double m) {
        //TODO write your implementation code here:
        m=m/1000;
        return m;
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "mkm")
    public Double mkm(@WebParam(name = "km") double km) {
        //TODO write your implementation code here:
        km=km*1000;
        return km;
    }
}
```

Output:



http://localhost:8080/km2m/km2m?Tester

km2m Web Service Tester

km2m Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

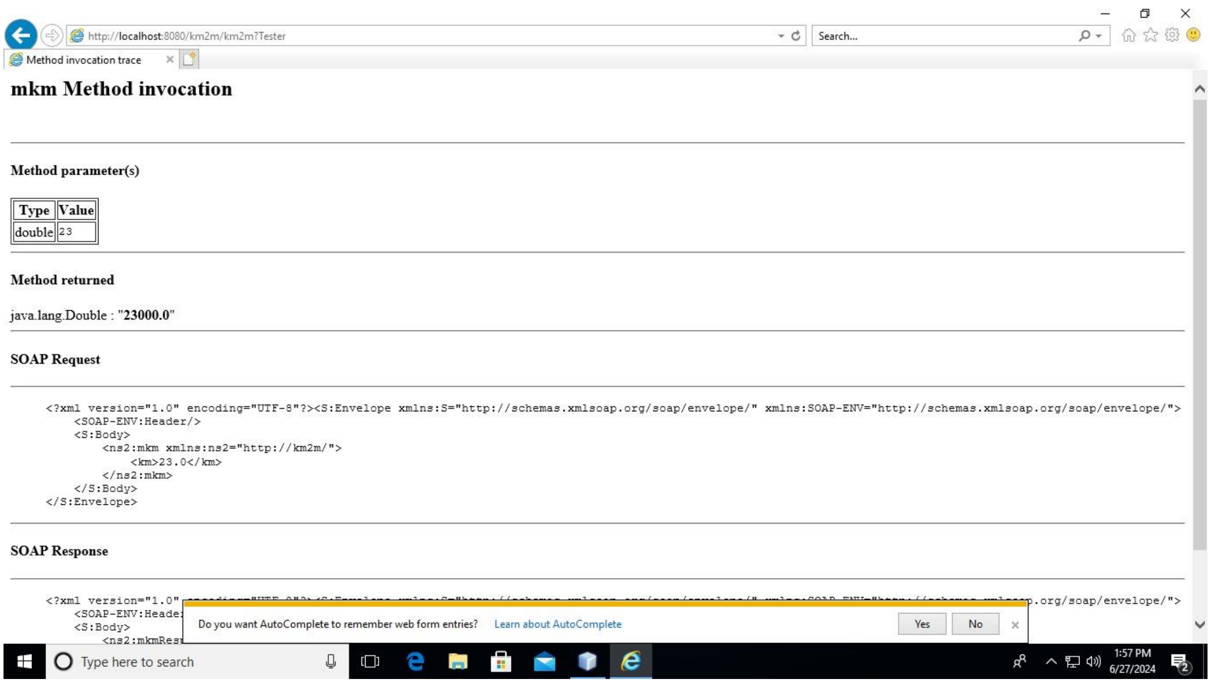
To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

public abstract java.lang.Double km2m.Km2M.kmm(double)
kmm (23)

public abstract java.lang.Double km2m.Km2M.mkm(double)
mkm (23)

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No



http://localhost:8080/km2m/km2m?Tester

Method invocation trace

mkm Method invocation

Method parameter(s)

Type	Value
double	23

Method returned

java.lang.Double : "**23000.0**"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:mkm xmlns:ns2="http://km2m/">
            <km>23.0</km>
        </ns2:mkm>
    </S:Body>
</S:Envelope>
```

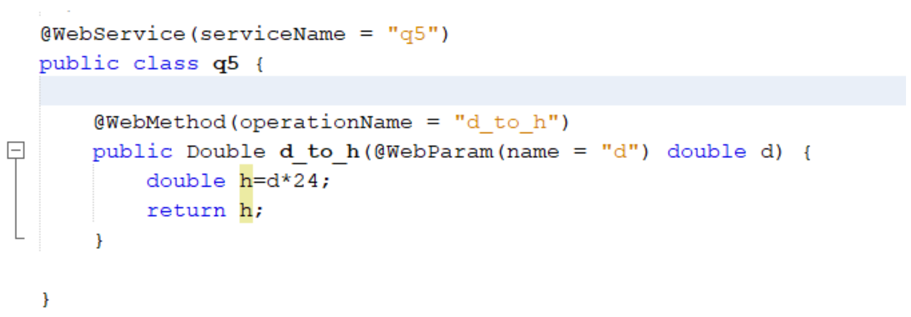
SOAP Response

```
<?xml version="1.0"                                                              .org/soap/envelope/">
    <SOAP-ENV:Heade
    <S:Body>
        <ns2:mkmRes
```

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No

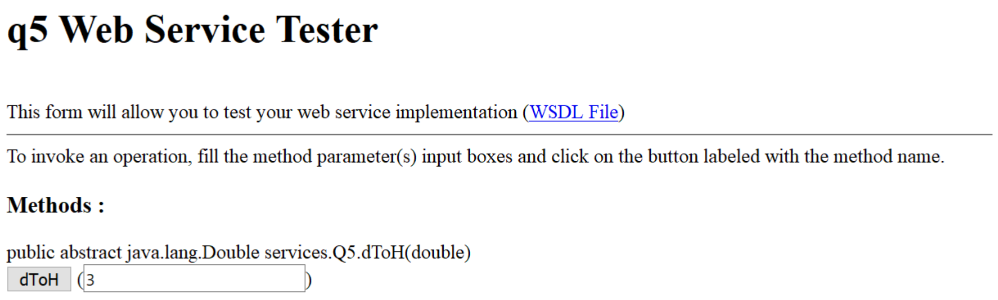
Q5. Write a program to convert the data from days to hours.

Logic Code:



```
@WebService(serviceName = "q5")
public class q5 {

    @WebMethod(operationName = "d_to_h")
    public Double d_to_h(@WebParam(name = "d") double d) {
        double h=d*24;
        return h;
    }

}
```

Output:



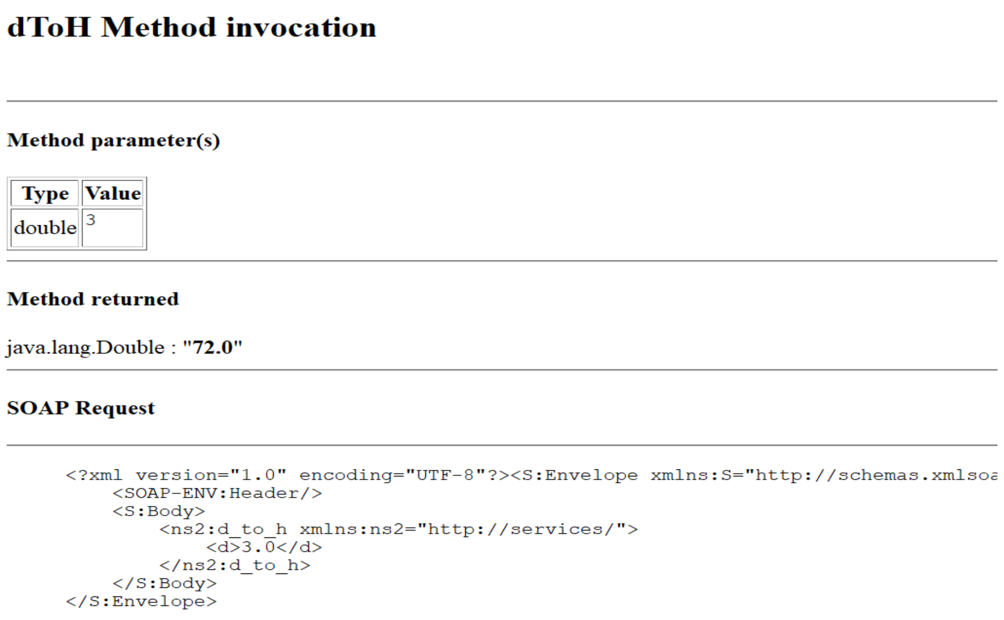
q5 Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

public abstract java.lang.Double services.Q5.dToH(double)



dToH Method invocation

Method parameter(s)

Type	Value
double	3

Method returned

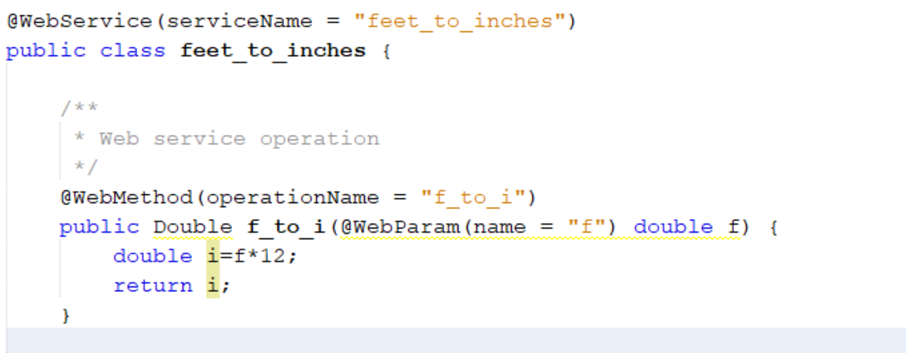
java.lang.Double : "**72.0**"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoa
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:d_to_h xmlns:ns2="http://services/">
            <d>3.0</d>
        </ns2:d_to_h>
    </S:Body>
</S:Envelope>
```

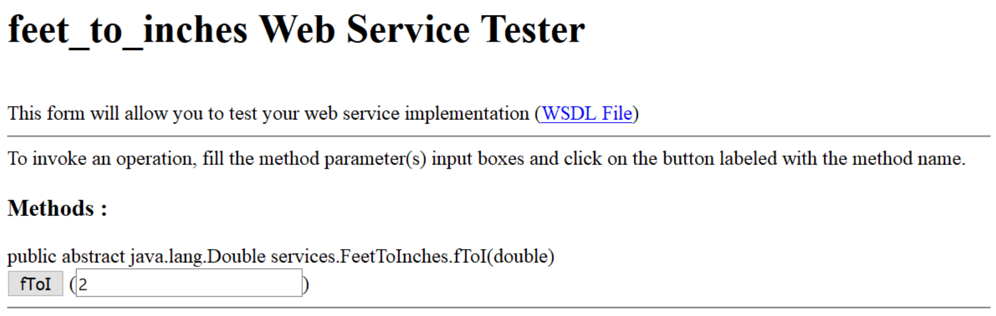
Q6.Write a program to convert the data from feet to inches.

Logic Code:



```java
@WebService(serviceName = "feet_to_inches")
public class feet_to_inches {

    /**
     * Web service operation
     */
    @WebMethod(operationName = "f_to_i")
    public Double f_to_i(@WebParam(name = "f") double f) {
        double i=f*12;
        return i;
    }
```

Output:



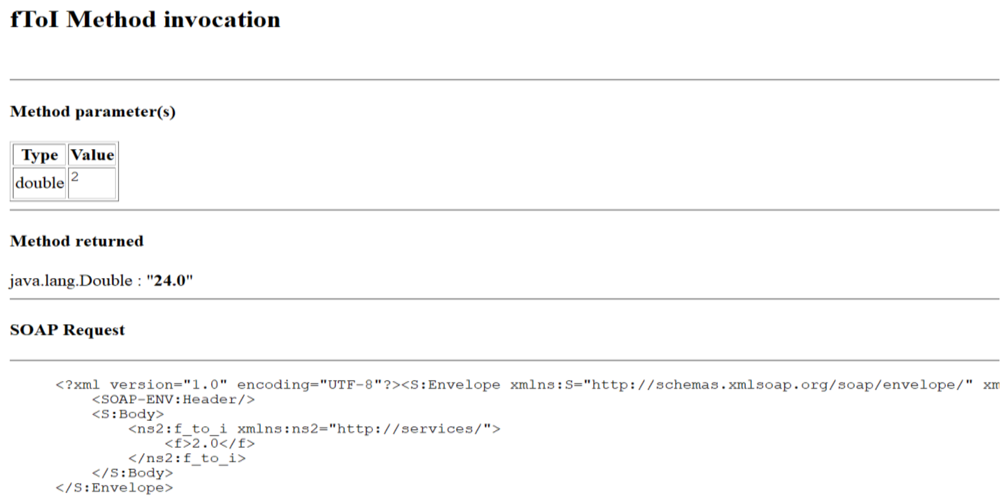
feet_to_inches Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

public abstract java.lang.Double services.FeetToInches.fToI(double)



fToI Method invocation

Method parameter(s)

Type	Value
double	2

Method returned

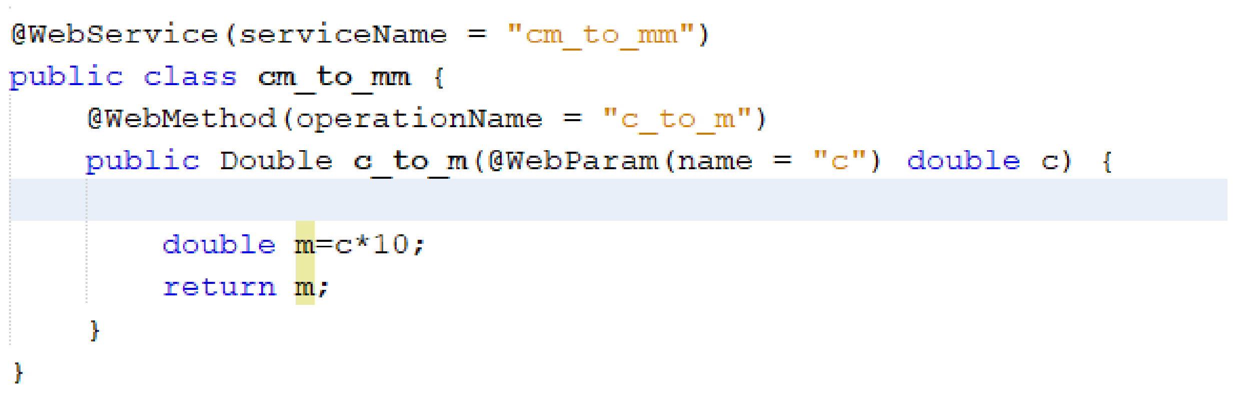
java.lang.Double : "**24.0**"

SOAP Request

```
    <?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xm
        <SOAP-ENV:Header/>
        <S:Body>
            <ns2:f_to_i xmlns:ns2="http://services/">
                <f>2.0</f>
            </ns2:f_to_i>
        </S:Body>
    </S:Envelope>
```

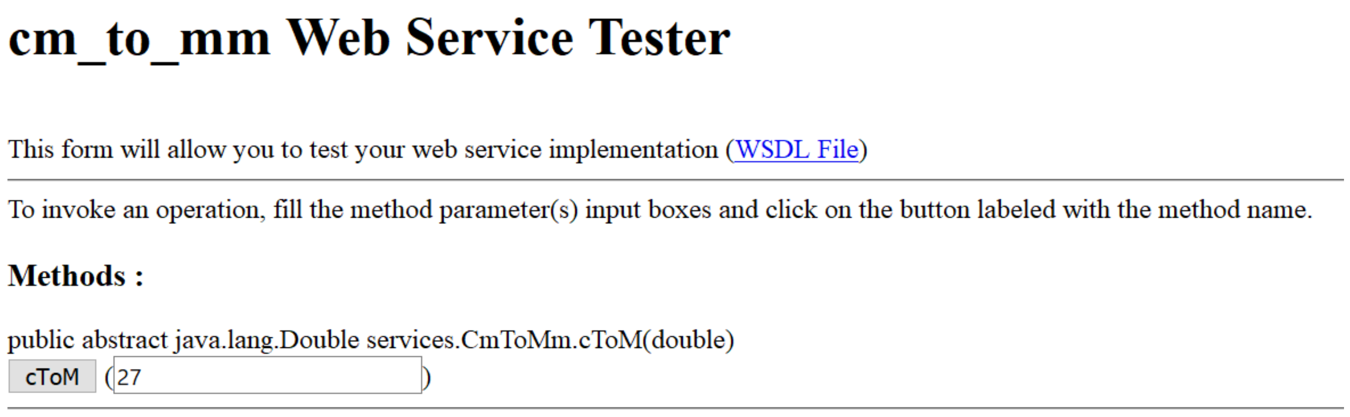
Q7. Write a program to convert the data from centimeter to millimeter.

Logic Code:



```
@WebService(serviceName = "cm_to_mm")
public class cm_to_mm {
    @WebMethod(operationName = "c_to_m")
    public Double c_to_m(@WebParam(name = "c") double c) {

        double m=c*10;
        return m;
    }
}
```

Output



cm_to_mm Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

public abstract java.lang.Double services.CmToMm.cToM(double)

cToM Method invocation

Method parameter(s)

Type	Value
double	27

Method returned

java.lang.Double : "**270.0**"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:c_to_m xmlns:ns2="http://services/">
            <c>27.0</c>
        </ns2:c_to_m>
    </S:Body>
</S:Envelope>
```