1. You are testing the following method: **[6+6 =12]**

public double celciusToFahrenheit(double tempInCelcius);

Devise two executable test cases for this method in the JUnit notation. The test case specifications are described for you –

/*case-1 when the input tempInCelcius < 0. The method should work properly for normal negative input temperatures.*/

Case no. 01

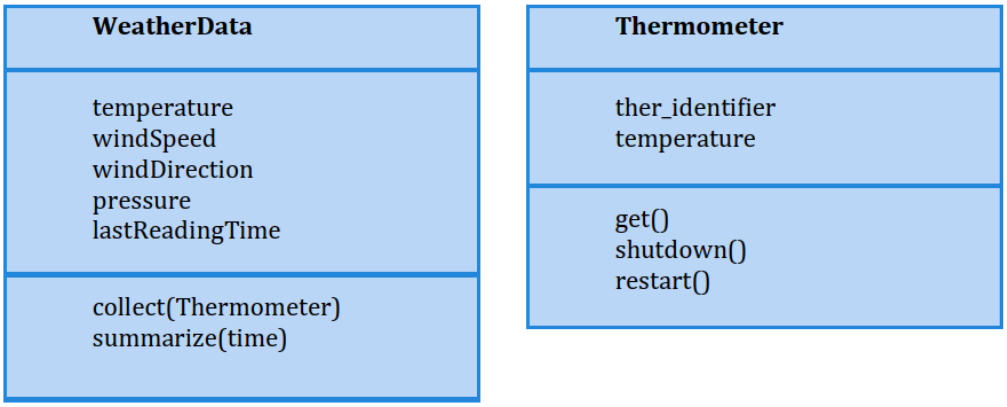
```
@Test

Public void testCelciusToFahrenheit_NormalNegative(){

    Temperature temp = new Temperature("No Name");

    double tempInCelcius = -10;

    String tempInFahrenheit = temp.celciusToFahrenheit(tempInCelcius);

    String expectedValue = 14;

    AssertEquals(tempInFahrenheit, expectedValue);

}
```

/*case-2 When the input parameter tempInCelcius < -273.16, the method should throw an "ImpossibleValueException" with the following message-'Input temperature below absolute zero.' */

Case no. 02

```
@Test

Public void testCelciusToFahrenheit_ImpossibleValue(){

    Temperature temp = new Temperature("No Name");

    double tempInCelcius = -275;

    Throwable exception = AssertThrows(ImpossibleValue.class, () ->
{temp.celciusToFahrenheit(tempInCelcius);});

    AssertEquals = ("Input temperature below absolute zero.",
exception.getMessage());

}
```

2.



Suppose you want to test the '***collect(Thermometer)***' method of **WeatherData** class. Which requires you to pass an instance of **Thermometer** class as parameter. But the **Thermometer** class is yet to be implemented. So, you need to use a stub. Write a Junit Test case to test the collect() method with the help of '**Mockito**' framework. [Hint : You need to mock the behavior of the **get()** method in the stub class to achieve correct weatherdata] **[8]**

```
@Test

Public void testCollect_temperatureCheck(){



    WeatherData wd = new WeatherData();

    Thermometer mocktherm = mock(Thermometer.class);

    When(mocktherm.get()).thenReturn(98);

    wd.collect(mocktherm);

    AssertEquals(wd.tempareture,98);

}
```