Problem 1

```
package problem1;

import java.awt.*;
import java.text.DecimalFormat;

import javax.swing.*;
import javax.swing.border.EmptyBorder;

public class Solution1 extends JFrame {

	private JTextField txtMille;
	private JTextField txtPound;
	private JTextField txtGallon;
	private JTextField txtFhrenheit;
	private JTextField txtKilometer;

	private JTextField txtKilogram;
	private JTextField txtLitre;
	private JTextField textCentigrade;

	final String TITLE = "UNIT CONVERSOR";
	public static final int DEFAULT_WIDTH = 470;
	public static final int DEFAULT_HEIGHT = 350;

	public static void main(String[] args) {
		Solution1 frame = new Solution1();
		frame.setVisible(true);
	}

	public Solution1() {
		JButton btnConvert = new JButton("Convert");
		btnConvert.setBounds(170, 208, 100, 70);

		this.setSize(DEFAULT_WIDTH, DEFAULT_HEIGHT);
		this.getContentPane().setLayout(null);

		this.setLayout(null);

		JLabel lblMile = new JLabel("Mile:");
		JLabel lblPound = new JLabel("Pound:");
		JLabel lblGallon = new JLabel("Gallon:");
		JLabel lblFahrenheit = new JLabel("Fahrenheit:");

		txtPound = new JTextField();

		txtGallon = new JTextField();
		txtFhrenheit = new JTextField();
		JLabel lblKilometer = new JLabel("Kilometer");
		JLabel lblKilogram = new JLabel("Kilogram");
		JLabel lblLitre = new JLabel("Litre");
		JLabel lblCentigrade = new JLabel("Centigrade");
		txtKilometer = new JTextField();
		txtLitre = new JTextField();
```

```
textCentigrade = new JTextField();
txtKilogram = new JTextField();

lblMile.setBounds(10, 36, 69, 22);

lblPound.setBounds(10, 78, 69, 22);

lblGallon.setBounds(10, 120, 69, 22);

lblFahrenheit.setBounds(10, 162, 69, 22);

txtMille = new JTextField();
txtMille.setBounds(89, 36, 108, 22);

txtMille.setColumns(10);

txtPound.setBounds(87, 78, 110, 22);

txtPound.setColumns(10);

txtGallon.setBounds(89, 120, 108, 22);

txtGallon.setColumns(10);

txtFhrenheit.setBounds(89, 162, 108, 22);

txtFhrenheit.setColumns(10);

lblKilometer.setBounds(226, 36, 79, 22);

lblKilogram.setBounds(226, 78, 79, 22);

lblLitre.setBounds(226, 120, 79, 22);

lblCentigrade.setBounds(226, 162, 79, 22);

txtKilometer.setBounds(315, 36, 109, 22);
```

```
txtKilometer.setColumns(10);


txtKilogram.setBounds(315, 78, 109, 22);

txtKilogram.setColumns(10);


txtLitre.setBounds(315, 120, 109, 22);

txtLitre.setColumns(10);



textCentigrade.setBounds(315, 162, 109, 22);

textCentigrade.setColumns(10);
txtMille.setText("0");
txtPound.setText("0");
txtGallon.setText("0");
txtFhrenheit.setText("0");
txtKilometer.setText("0");
txtKilogram.setText("0");
txtLitre.setText("0");
textCentigrade.setText("0");

btnConvert.addActionListener(evt -> {









      double tmpMile = 0;
      double tempPound = 0;
      double tempGallon = 0;
      double tempFahrenheit = 0;

      double tempKilometer = 0;
      double tempKilogram = 0;
      double tempLitre = 0;
      double tempCentigrade = 0;

      if (!txtMille.getText().chars().allMatch(x ->
((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false) )  )
                        {
            tmpMile = 0;
            JOptionPane.showMessageDialog(null, "Upss  :(  Mile
must be number. Check it.  ");
      } else {
```

```
                        tmpMile = ConvertToNumber(txtMille.getText());
                    }

                    if (!txtPound.getText().chars().allMatch(x -
>((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false)) ) {
                        tempPound = 0;
                        JOptionPane.showMessageDialog(null, "Upss  :(  Pound
must be number. Check it.  ");
                    }

                    else {
                        tempPound = ConvertToNumber(txtPound.getText());
                    }

                    if (! txtGallon.getText().chars().allMatch(x -
>((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false)) ) {
                        tempGallon = 0;
                        JOptionPane.showMessageDialog(null, "Upss  :(  Gallon
must be number. Check it.  ");
                    }

                    else {
                        tempGallon = ConvertToNumber( txtGallon.getText());
                    }

                    if (!txtFhrenheit.getText().chars().allMatch(x -
>((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false)) ) {
                        tempFahrenheit = 0;
                        JOptionPane.showMessageDialog(null, "Upss  :(
Fahrenheit must be number. Check it.  ");
                    }

                    else {
                        tempFahrenheit =
ConvertToNumber(txtFhrenheit.getText());
                    }

                    if (!txtKilometer.getText().chars().allMatch(x -
>((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false)) ) {
                        tempKilometer = 0;
                        JOptionPane.showMessageDialog(null, "Upss  :(
Kilometer must be number. Check it.  ");
                    }

                    else {
                        tempKilometer =
ConvertToNumber(txtKilometer.getText());
                    }

                    if (!txtKilogram.getText().chars().allMatch(x -
>((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false)) ) {
                        tempKilogram = 0;
                        JOptionPane.showMessageDialog(null, "Upss  :(
Kilogram must be number. Check it.  ");
                    }
```

```
                    else {
                          tempKilogram = ConvertToNumber(txtKilogram.getText());
                    }

                    if (!txtLitre.getText().chars().allMatch(x -
>((Character.isDigit(x)  || x==',' || x=='.' ) ? true : false)) ) {
                          tempLitre = 0;
                          JOptionPane.showMessageDialog(null, "Upss  :(  Litre
must be number. Check it.  ");
                    }

                    else {
                          tempLitre = ConvertToNumber(txtLitre.getText());
                    }

                    if (!textCentigrade.getText().matches("\\d+(\\.\\d+)?")) {
                          tempCentigrade = 0;
                          JOptionPane.showMessageDialog(null, "Upss  :(
Centigrade must be number. Check it.  ");
                    }

                    else {
                          tempCentigrade =
ConvertToNumber(textCentigrade.getText());
                    }



                    if (tmpMile == 0 && tempKilometer > 0)
                    {
                          tmpMile = 0.62 * tempKilometer;
                    }
                    else if (tmpMile > 0 && tempKilometer == 0)
                    {
                          tempKilometer = 1.6 * tmpMile;
                    }
                    else
                    {
                          tmpMile = 0.62 * tempKilometer;
                          tempKilometer = 1.6 * tmpMile;
                    }

                    if (tempPound == 0 && tempKilogram > 0)
                    {
                          tempPound = 2.2 * tempKilogram;
                    } else if (tempPound > 0 && tempKilogram == 0) {
                          tempKilogram = 0.45 * tempPound;
                    }

                    if (tempGallon == 0 && tempLitre > 0)
                    {
                          tempGallon = 0.264 * tempLitre;
                    }
```

```
            else if (tempGallon > 0 && tempLitre == 0)
            {
                  tempLitre = 3.785 * tempGallon;
            }

            if (tempFahrenheit == 0 && tempCentigrade > 0) {
                  tempFahrenheit = (tempCentigrade * 1.8) + 32;
            }

            else if (tempFahrenheit > 0 && tempCentigrade == 0)
            {
                  tempCentigrade = (tempFahrenheit - 32) / 1.8;
            }
            else {
                  tempFahrenheit = (tempCentigrade * 1.8) + 32;
                  tempCentigrade = (tempFahrenheit - 32) / 1.8;
            }

            txtMille.setText(simpleFormat(tmpMile));
            txtGallon.setText(simpleFormat(tempGallon));
            txtFhrenheit.setText(simpleFormat(tempFahrenheit));
            txtLitre.setText(simpleFormat(tempLitre));
            textCentigrade.setText(simpleFormat(tempCentigrade));
            txtKilometer.setText(simpleFormat(tempKilometer));
            txtKilogram.setText(simpleFormat(tempKilogram));
            txtPound.setText(simpleFormat(tempPound));

      }

      );

      this.add(btnConvert);
      this.setTitle(TITLE);
      this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      this.add(textCentigrade);
      this.add(txtLitre);
      this.add(txtKilogram);
      this.add(txtKilometer);
      this.add(lblCentigrade);
      this.add(lblLitre);
      this.add(lblKilogram);
      this.add(lblKilometer);
      this.add(txtFhrenheit);
      this.add(txtGallon);
      this.add(txtPound);
      this.add(txtMille);
      this.add(lblFahrenheit);
      this.add(lblGallon);
      this.add(lblPound);
      this.add(lblMile);

}

public Double ConvertToNumber(String number)
```

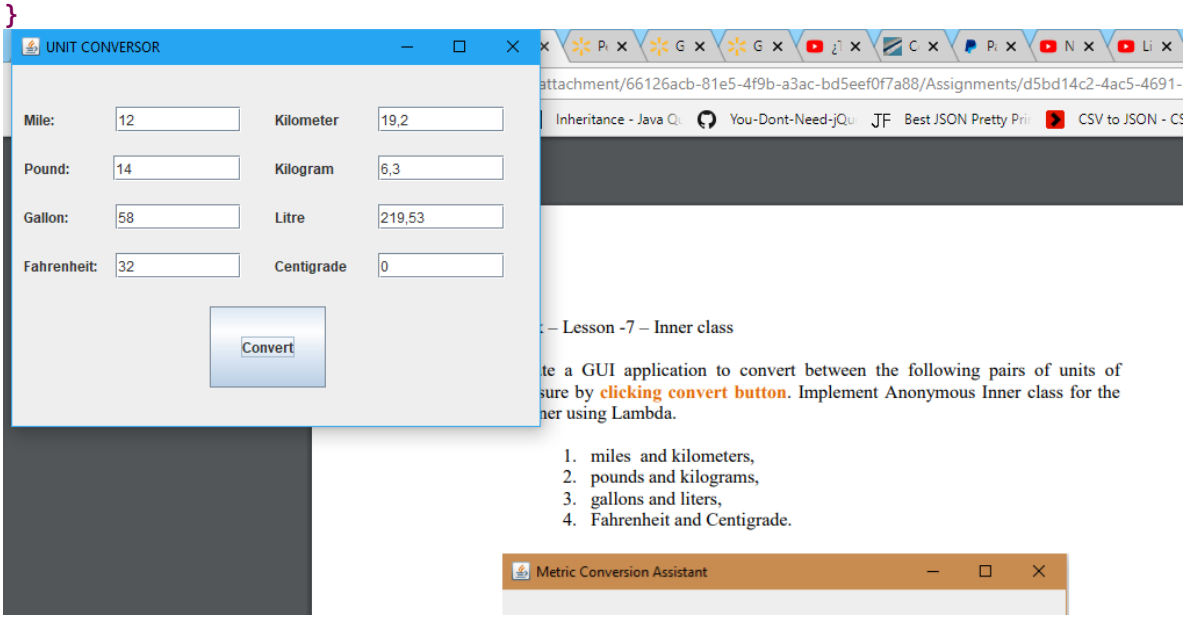
```
        {
                number=number.replace(",", ".");
                number=number.replace(" ", "");
                try {
                        return Double.parseDouble(number);

                }
                catch (Exception e)
                {

                        try {
                                Float f = Float.parseFloat(number);
                                Double d = new Double(f.toString());
                                return  d ;
                        } catch (Exception e2)
                        {
                                return Double.parseDouble("-999999999999999");
                        }
                }
        }

        public String simpleFormat(double number) {
                DecimalFormat df = new DecimalFormat("#.##");
                return df.format(number);

        }
}
```



Problem 2

```
package problem2;

public class MyTable {
        private Entry[] entries= new Entry[26];
        private final  String alphabet="abcdefghijklmnopqrstuvwxyz";
```

```java
// returns the String that is matched with char c in the table
public String get(char c)
{
	int index=this.alphabet.indexOf(c);
	Entry result = this.entries[index];

	return result.str;
}

// adds to the table a pair (c, s) so that s can be looked up using c
public void add(char c, String s) {
	int index=this.alphabet.indexOf(c);
	this.entries[index]=new Entry(c, s);

}

// returns a String consisting of nicely formatted display
// of the contents of the table
public String toString() {
	StringBuilder builder = new StringBuilder();
	for (Entry entry : entries)
	{

		if (entry!=null)
		{
			builder.append(entry.toString()+"\n");
		}


	}
	return builder.toString();
}

private class Entry {
	char ch; String str;

	Entry(char ch, String str)
	{


		this.ch=ch;
		this.str=str;
	}


	public String toString() {
		alphabet.equals("3");
		StringBuilder builder = new StringBuilder();
		builder.append(this.ch);
		builder.append("->");
		builder.append(this.str);
		return builder.toString();
	}
```

```
        }

        public static void main(String[] args) {
                MyTable t = new MyTable();
                t.add('a', "Andrew");
                t.add('b',"Billy");
                t.add('c',"Charlie");
                String s = t.get('b');
                System.out.println(s);


                System.out.println(t);
                //output

        }

}
```

Problems @ Javadoc Declaration Search Console Coverage Rebase Interactive

```
<terminated> MyTable (1) [Java Application] C:\Program Files\Java\jdk1.8.0_60\jre\bin\javaw.exe (10 jun. 2018 23:04
Billy
a->Andrew
b->Billy
c->Charlie
```