

```

/*
 * Justin Mendes
 * December 23, 2016
 * Unit 4 Activity 3 Program/Question 1
 * This program will convert standard times into traditional time (also considering
inputs past 23:59:59)
 */
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class StandardTime
{
    static String amPM;//will hold am/pm for standard time
    static String traditionalTime;//will store traditional time from user
    static int hours, mins, secs;//will store hours, minutes and seconds
    public static void main (String args []) throws IOException
    {
        //Variable Declarations and Initializations
        String standardTime;
        int tryAgain = 1;
        BufferedReader br = new BufferedReader (new InputStreamReader
(System.in)); // user input
        while (tryAgain == 1)
        {
            System.out.println("Standard Time to Traditional Time
Converter");

            System.out.println("=====");
            System.out.println("\nInput a time in Standard Form
(HH:MM:SS):");

            standardTime = br.readLine();//user inputs time in standard form
            while (standardTime.length() != 8)
            {
                System.out.println("Invalid time entered.");
                System.out.println("Input a time in Standard Form that has
this form HH:MM:SS ...");
                standardTime = br.readLine();//user inputs time in
standard form
            } //end loop
            convertToTraditional(standardTime);
            System.out.println(standardTime + " is equivalent to "+
traditionalTime);

            System.out.println("\nEnter 1 to try again.");
            tryAgain = Integer.parseInt(br.readLine());//user decides to try
again

        } //end loop
    } //end main
    public static void convertToTraditional(String standardTime)
    {
        int days = 0;
        hours = Integer.parseInt(standardTime.substring(0 , 2));
        mins = Integer.parseInt(standardTime.substring(3, 5));
        secs = Integer.parseInt(standardTime.substring(6, 8));
        if(hours >= 12)
        {

```

```

        //for when times are past the value 12(not traditional to print)
        hours = hours - 12;
        //times past 12 are P.M (ex. 13:00 into 1:00 P.M)
        amPM = "P.M";
    }//end if
    else
    {
        amPM = "A.M";
    }//end else
    //for the inputs where the hours = "00", and it wouldn't be 00:00 A.M it
    should be 12:00 A.M
    if (hours == 0)
    {
        hours = 12;
    }//end if
    /*
    * Code past here deals with inputs with higher than normal inputs
    * ex. the minutes and seconds being over 60, 13:90:80
    * ex. hours going past 24, 25:20:45
    */
    //to make the amPM still accurate when input is past 24:00:00(REMOVABLE
    WITH PROPER INPUT)
    if(hours >= 24 || (mins / 60) + hours >= 24)
    {
        if (Math.floor((hours / 12) % 2) == 1)
        {
            amPM = "P.M";
        }//end if
        else
        {
            amPM = "A.M";
        }//end else
    }//end if
    //to make the minutes always below 60 and help with the output of a
    valid traditional time(REMOVABLE WITH PROPER INPUT)
    if (mins >= 60)
    {
        hours = hours + 1;
        mins = mins - 60;
    }//end if
    //to make sure that the seconds they input are accounted for(REMOVABLE
    WITH PROPER INPUT)
    if (secs >= 60)
    {
        mins = mins + 1;
        secs = secs - 60;
        //to re-check if hours are supposed to be made higher because of
        minutes being changed into 60, ex. 13:59:60
        if (mins >= 60)
        {
            hours = hours + 1;
            mins = mins - 60;
        }//end if
    }//end if
    if (mins < 10)

```

```

    {
        //mins = "0" + mins;
    } //end if
    //to add how many days have passed with higher standard time
inputs(REMOVABLE WITH PROPER INPUT)
    for (int i = hours; i >= 24; i = i - 24)
    {
        days++;
        if (i >= 24 && i < 48)
        {
            amPM = amPM + " after " + days + " day(s)";
        } //end if
    } //end loop
    //to make sure the hours are in standard time(REMOVABLE WITH PROPER
INPUT)
    while (hours > 12)
    {
        hours = hours - 12;
    } //end loop
    /*
     * Removable/special code ends here
     */
    traditionalTime = hours + ":" + mins + " " + amPM;
    //to make sure the MINUTES output is always viable
    if (mins < 10)
    {
        traditionalTime = hours + ":0" + mins + " " + amPM;
    } //end if
    } //end method convertToTraditional
} //end class

```

```

Standard Time to Traditional Time Converter
=====

Input a time in Standard Form (HH:MM:SS):
13:59:60
13:59:60 is equivalent to 2:00 P.M

Enter 1 to try again.
1
Standard Time to Traditional Time Converter
=====

Input a time in Standard Form (HH:MM:SS):
14:35:22
14:35:22 is equivalent to 2:35 P.M

Enter 1 to try again.
1
Standard Time to Traditional Time Converter
=====

Input a time in Standard Form (HH:MM:SS):
97:45:90
97:45:90 is equivalent to 1:46 P.M after 3 day(s)

Enter 1 to try again.
1

```

```

/*
 * Justin Mendes
 * December 23, 2016
 * Unit 4 Activity 3 Program/Question 2
 * This program will calculate the gross wages of a person given their number of
hours and the hourly rate of pay.
 */
import java.io.*;
import java.text.DecimalFormat;
public class GrossWages
{
    static double grossWage;//gross wage that will be calculated in the
calculateGrossWage method, and printed in main method
    public static void main (String args []) throws IOException
    {
        //Variable Declarations and Initializations
        double hours, rate;
        BufferedReader br = new BufferedReader (new InputStreamReader
(System.in));
        DecimalFormat twoDigit = new DecimalFormat ("#,##0.00");
        System.out.println("Gross Wage Calculator\n=====\\n");
        System.out.println("Enter the number of hours worked:");
        hours = Double.parseDouble(br.readLine());
        System.out.println("\\nEnter the hourly rate of pay (in $):");
        rate = Double.parseDouble(br.readLine());
        calculateGrossWages(rate, hours);
        System.out.println("\\nGross wage: $" + twoDigit.format(grossWage));
    } //end main
}

```

```

static void calculateGrossWages(double rate, double hours)
{
    //checking overtime pay
    if(hours > 40)
    {
        grossWage = (rate * 40) + (hours - 40) * rate * 1.5;
    } //end if
    else
    {
        grossWage = rate * hours;
    } //end else
} //end method calculateGrossWages
} //end class

Gross Wage Calculator
=====

Enter the number of hours worked:
41

Enter the hourly rate of pay (in $):
2

Gross wage: $83.00

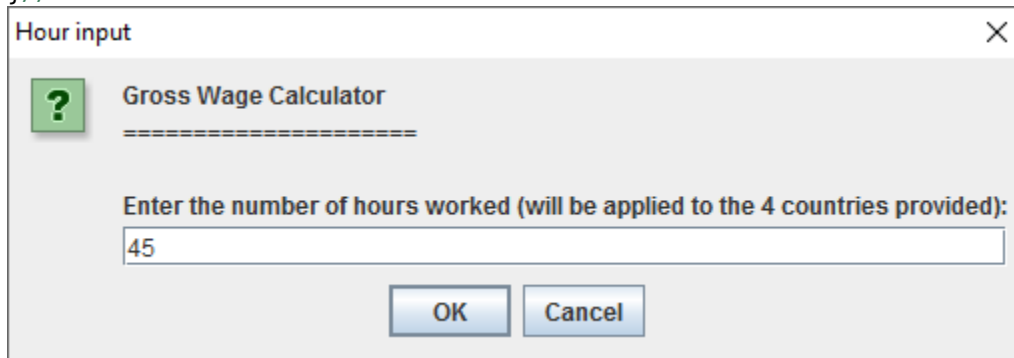
/*
 * Justin Mendes
 * December 23, 2016
 * Unit 4 Activity 3 Program/Question 2
 * This program will calculate the gross wages of a person given their number of
hours and the hourly rate of pay.
 */
import java.io.*;
import java.text.DecimalFormat;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
public class GrossWages2
{
    private static final Object[][] rowData = {};
    private static final Object[] columnNames = {"Country", "Hourly Wage in $
Canadian", "Hours", "Gross Wage"};
    static double bangladesh = 0.15, china = 0.48, dominicanRepublic = 1.60, haiti
= 0.5;
    public static void main (String args []) throws IOException
    {
        //Variable Declarations and Initializations
        int hours;
        DefaultTableModel listTableModel = new DefaultTableModel(rowData,
columnNames);
        DecimalFormat twoDigit = new DecimalFormat("#,##0.00");
        hours = Integer.parseInt(JOptionPane.showInputDialog(null, "Gross Wage
Calculator\n=====
\nEnter the number of hours worked (will be
applied to the 4 countries provided):", "Hour input", JOptionPane.QUESTION_MESSAGE));

```


```

        listTableModel.addRow(columnNames);
        calculateGrossWages(hours);
        listTableModel.addRow(new Object[]{"Bangladesh", "$0.15", hours,
twoDigit.format(bangladesh)});
        listTableModel.addRow(new Object[]{"China", "$0.48", hours,
twoDigit.format(china)});
        listTableModel.addRow(new Object[]{"Dominican Republic", "$1.60", hours,
twoDigit.format(dominicanRepublic)});
        listTableModel.addRow(new Object[]{"Haiti", "$0.55", hours,
twoDigit.format(haiti)});
        JTable listTable;
        listTable = new JTable(listTableModel);
        listTable.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
        listTable.setCellEditor(null);
        listTable.setBounds(50, 200, 500, 200);
        JFrame frame = new JFrame();
        frame.add(listTable);
        frame.setVisible(true);
        frame.pack();
    } //end main
    static void calculateGrossWages(int hours)
    {
        bangladesh = bangladesh * hours;
        china = china * hours;
        dominicanRepublic = dominicanRepublic * hours;
        haiti = haiti * hours;
    } //end method calculatedGrossWages
} //end class

```



Hour input

 Gross Wage Calculator

=====

Enter the number of hours worked (will be applied to the 4 countries provided):

45

OK Cancel

Country	Hourly Wage in \$ Canadian	Hours	Gross Wage
Bangladesh	\$0.15	45	6.75
China	\$0.48	45	21.60
Dominican Republic	\$1.60	45	72.00
Haiti	\$0.55	45	22.50