**CSC 3020 – Java Programming**

**Homework 2 – [your name]**

**25 points – Due February 7, 10am**

**Late deadline is February 9, 11:59pm, but 20% off**

**a)** Save this document with your name and the homework number somewhere in the file name.

**b)** Type/paste your answers into the document.

**c)** Submit this document and your .java file(s) to the Blackboard item where you downloaded this document. Do not submit a zip file but individually attach your files.

**1) [8 points]** Write regular expressions after the ► to validate the following strings:

**a)** An International Standard Book Number (ISBN). This is a unique 13-digit number assigned to each edition of a book. An ISBN contains hyphens. Here is a sample: 978-3-16-148410-0. Assume that each digit is in the range 0-9.

► [0-9]{3}-[0-9]-[0-9]{2}-[0-9]{6}-[0-9]

**b)** A one- or two-character chemical symbol of an element from the Periodic Table.

► [A-Z]|([A-Z][a-z])

**c)** A measurement including a whole number and units separated by one space. A sample measurement is 34 oz. Assume that units is lower case letters only.

► [1-9][0-9]\* [a-z]+

OR

► ([0]|[1-9][0-9]\*) [a-z]+

**d)** A two-character abbreviation for any of the eight US states or the Canadian province adjacent to the Great Lakes. Assume the abbreviation is upper case characters only.

► MN|WI|IL|IN|MI|OH|PA|NY|ON

OR

► (MN)|(WI)|(IL)|(IN)|(MI)|(OH)|(PA)|(NY)|(ON)

**2) [9 points]** You've been hired by *Money Monarchs* to write a Java console application that converts between currencies US Dollar ($), Euro (€), and Japanese Yen (¥). Prompt the user for a conversion code:

| Code | Conversion | Last run input |
| --- | --- | --- |
| a | US Dollar → Euro | 32 |
| b | US Dollar → Japanese Yen | 64 |
| c | Euro → US Dollar | 200 |
| d | Euro → Japanese Yen | 300 |
| e | Japanese Yen → US Dollar | 24 |
| f | Japanese Yen → Euro | 28 |
| x | Exit |  |

Then prompt and get from the user a currency value. Convert the currency value and show the two values, rounded to two decimal places, with appropriate units. Continue to prompt the user for conversion codes until the sentinel value of ‘x’. Research the current conversion rates, and represent and use them as double constants. Locate the UNICODE values for units € and ¥, and represent and use them as string constants. They will be in the form "\uHHHH". Use the inputs shown in the table for the last six inputs.

**//======================================================================**

**//**

**// Title: Money Monarchs**

**// Course: CSC 3020**

**// Homework: 2-2**

**// Author: Dan Ouellette**

**// Date: 7 February 2018**

**// Description:**

**// This Java console application converts between currencies US Dollar,**

**// Euro, and Japanese Yen. It prompts the user for one of the following**

**// conversion codes:**

**//**

**// Code Conversion**

**// a US Dollar -> Euro**

**// b US Dollar -> Japanese Yen**

**// c Euro -> US Dollar**

**// d Euro -> Japanese Yen**

**// e Japanese Yen -> US Dollar**

**// f Japanese Yen -> Euro**

**// x Exit**

**//**

**// It then uses a validation loop to prompt and get from the**

**// user a positive currency value. It converts the currency**

**// value and shows the two values, rounded to two decimal**

**// places. The application continues to prompt the user for**

**// conversion codes until the sentinel value of ‘x’.**

**//======================================================================**

**package wsu.HW02\_02;**

**// Import classes**

**import java.util.Scanner;**

**//======================================================================**

**// class HW02\_02**

**//======================================================================**

**public class HW02\_02**

**{**

**//------------------------------------------------------------------**

**// main**

**//------------------------------------------------------------------**

**public static void main(String[] args)**

**{**

**// Declare constants**

**final double USDtoEURO = 0.889452;**

**final double USDtoYEN = 111.021;**

**final double EUROtoUSD = 1.1246;**

**final double EUROtoYEN = 124.820;**

**final double YENtoUSD = 0.00900612;**

**final double YENtoEURO = 0.00801051;**

**final String EURO\_SYMBOL = "\u20ac";**

**final String YEN\_SYMBOL = "\u00a5";**

**final String COLFMT = "%,1.2f";**

**// Declare variables**

**char option;**

**double inValue;**

**double outValue;**

**Scanner keyboard = new Scanner(System.in);**

**// Show application header**

**System.out.println("Welcome to Money Monarchs");**

**System.out.println("-------------------------");**

**// Prompt for and get option**

**System.out.println("\nOption\tConversion");**

**System.out.println("a\tUS Dollar -> Euro");**

**System.out.println("b\tUS Dollar -> Japanese Yen");**

**System.out.println("c\tEuro -> US Dollar");**

**System.out.println("d\tEuro -> Japanese Yen");**

**System.out.println("e\tJapanese Yen -> US Dollar");**

**System.out.println("f\tJapanese Yen -> Euro");**

**System.out.println("x\tExit");**

**System.out.print("Enter an option: ");**

**option = keyboard.next().charAt(0);**

**// Loop to process options**

**while (Character.toLowerCase(option) != 'x')**

**{**

**// Test entered option**

**System.out.println();**

**switch (option)**

**{**

**// Convert from US Dollar -> Euro**

**case 'a':**

**System.out.println("US Dollar -> Euro");**

**System.out.print("Enter a value in US Dollar: ");**

**inValue = keyboard.nextDouble();**

**outValue = inValue \* USDtoEURO;**

**System.out.printf(**

**"$" + COLFMT + " equals " +**

**EURO\_SYMBOL + COLFMT + "%n",**

**inValue, outValue);**

**break;**

**// Convert from US Dollar -> Japanese Yen**

**case 'b':**

**System.out.println("US Dollar -> Japanese Yen");**

**System.out.print("Enter a value in US Dollar: ");**

**inValue = keyboard.nextDouble();**

**outValue = inValue \* USDtoYEN;**

**System.out.printf(**

**"$" + COLFMT + " equals " +**

**YEN\_SYMBOL + COLFMT + "%n",**

**inValue, outValue);**

**break;**

**// Convert from Euro -> US Dollar**

**case 'c':**

**System.out.println("Euro -> US Dollar");**

**System.out.print("Enter a value in Euro: ");**

**inValue = keyboard.nextDouble();**

**outValue = inValue \* EUROtoUSD;**

**System.out.printf(**

**EURO\_SYMBOL + COLFMT + " equals " +**

**"$" + COLFMT + "%n",**

**inValue, outValue);**

**break;**

**// Convert from Euro -> Japanese Yen**

**case 'd':**

**System.out.println("Euro -> Japanese Yen");**

**System.out.print("Enter a value in Euro: ");**

**inValue = keyboard.nextDouble();**

**outValue = inValue \* EUROtoYEN;**

**System.out.printf(**

**EURO\_SYMBOL + COLFMT + " equals " +**

**YEN\_SYMBOL + COLFMT + "%n",**

**inValue, outValue);**

**break;**

**// Convert from Japanese Yen -> US Dollar**

**case 'e':**

**System.out.println("Japanese Yen -> US Dollar");**

**System.out.print("Enter a value in Japanese Yen: ");**

**inValue = keyboard.nextDouble();**

**outValue = inValue \* YENtoUSD;**

**System.out.printf(**

**YEN\_SYMBOL + COLFMT + " equals " +**

**"$" + COLFMT + "%n",**

**inValue, outValue);**

**break;**

**// Convert from Japanese Yen -> Euro**

**case 'f':**

**System.out.println("Japanese Yen -> Euro");**

**System.out.print("Enter a value in Japanese Yen: ");**

**inValue = keyboard.nextDouble();**

**outValue = inValue \* YENtoEURO;**

**System.out.printf(**

**YEN\_SYMBOL + COLFMT + " equals " +**

**EURO\_SYMBOL + COLFMT + "%n",**

**inValue, outValue);**

**break;**

**// Handle invalid option**

**default:**

**System.out.println("Error: invalid option '" +**

**option + "'.");**

**}**

**// Prompt for and get option**

**System.out.println("\nOption\tConversion");**

**System.out.println("a\tUS Dollar -> Euro");**

**System.out.println("b\tUS Dollar -> Japanese Yen");**

**System.out.println("c\tEuro -> US Dollar");**

**System.out.println("d\tEuro -> Japanese Yen");**

**System.out.println("e\tJapanese Yen -> US Dollar");**

**System.out.println("f\tJapanese Yen -> Euro");**

**System.out.println("x\tExit");**

**System.out.print("Enter an option: ");**

**option = keyboard.next().charAt(0);**

**}**

**// Close keyboard**

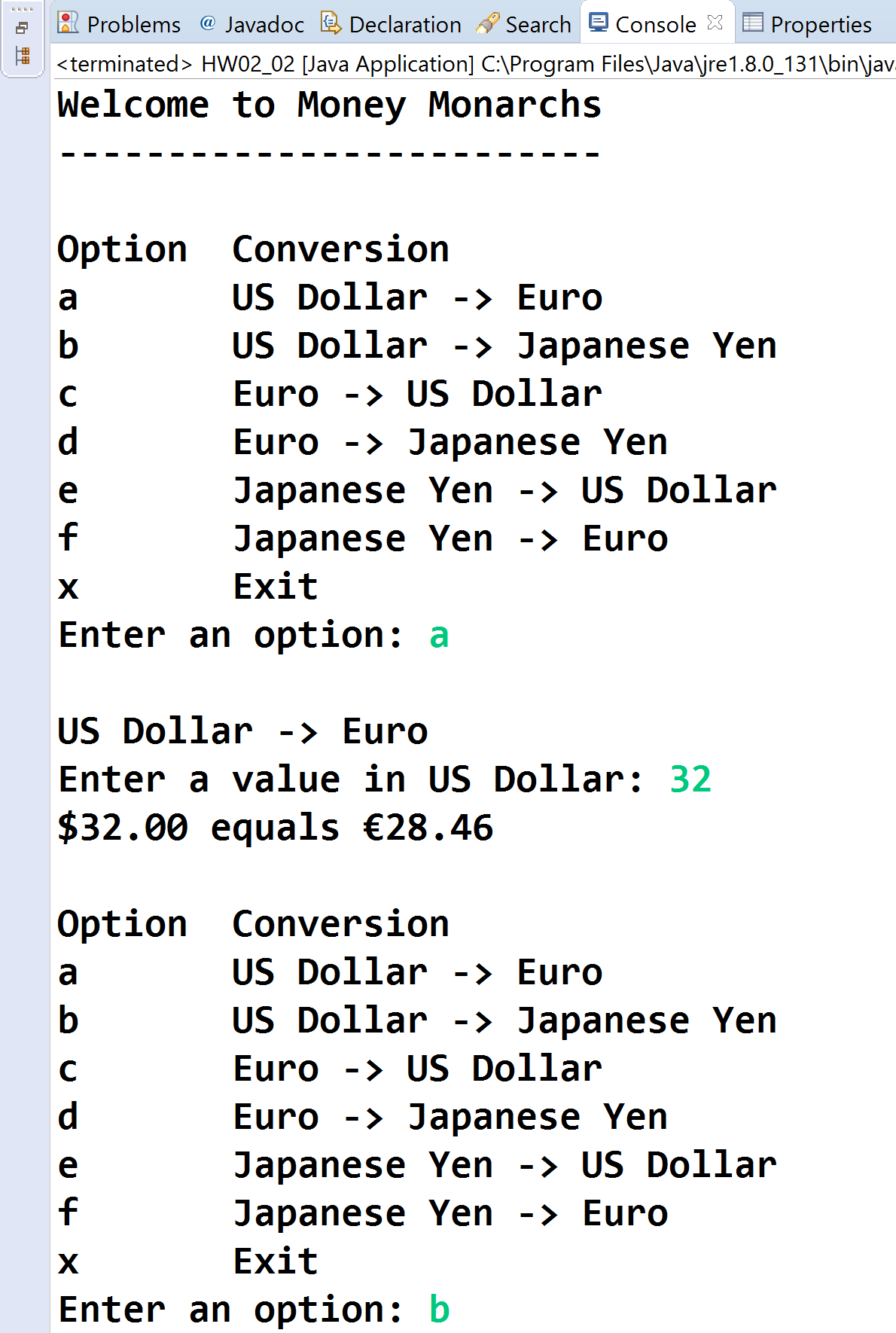
**keyboard.close();**

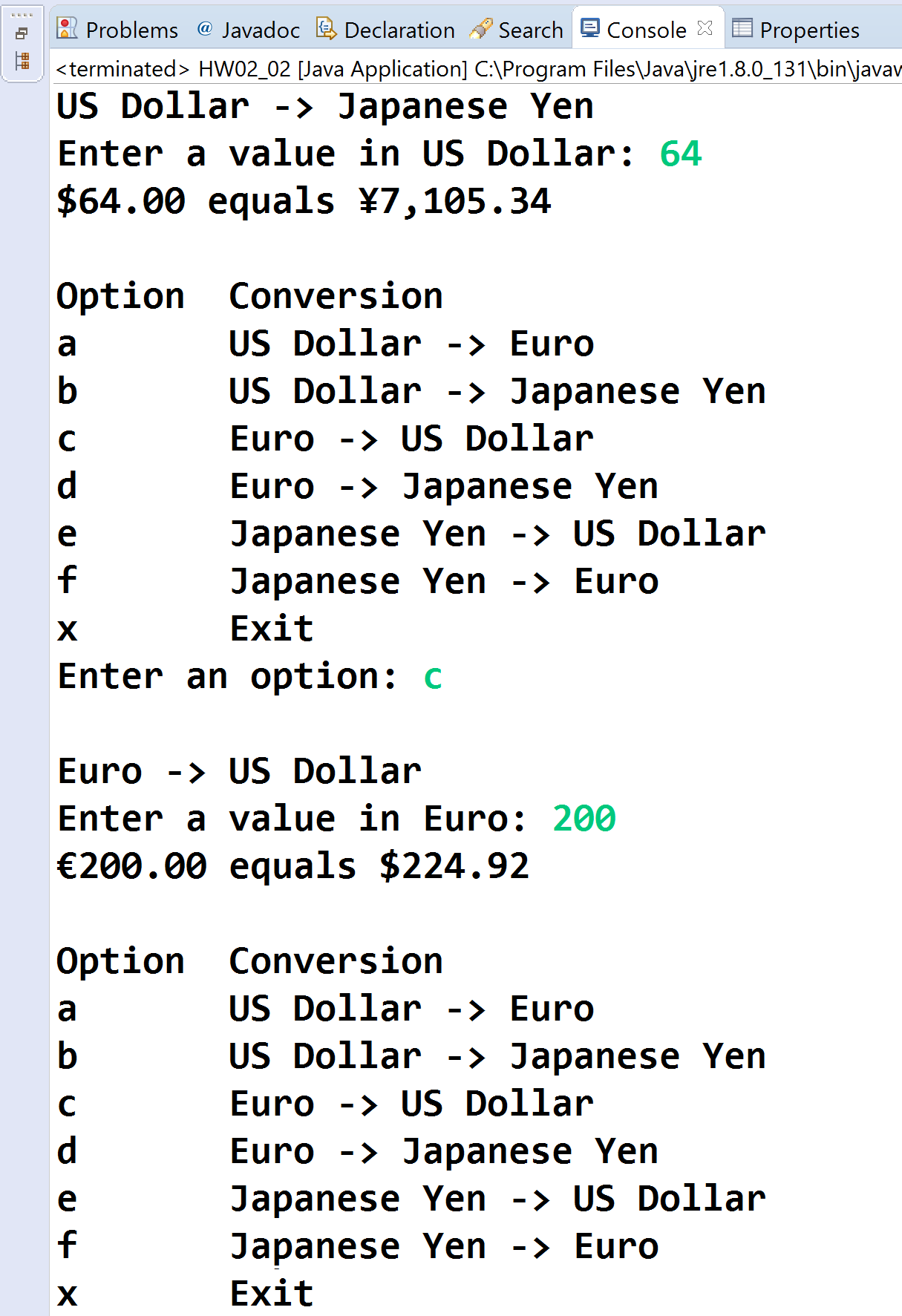
**// Show application close**

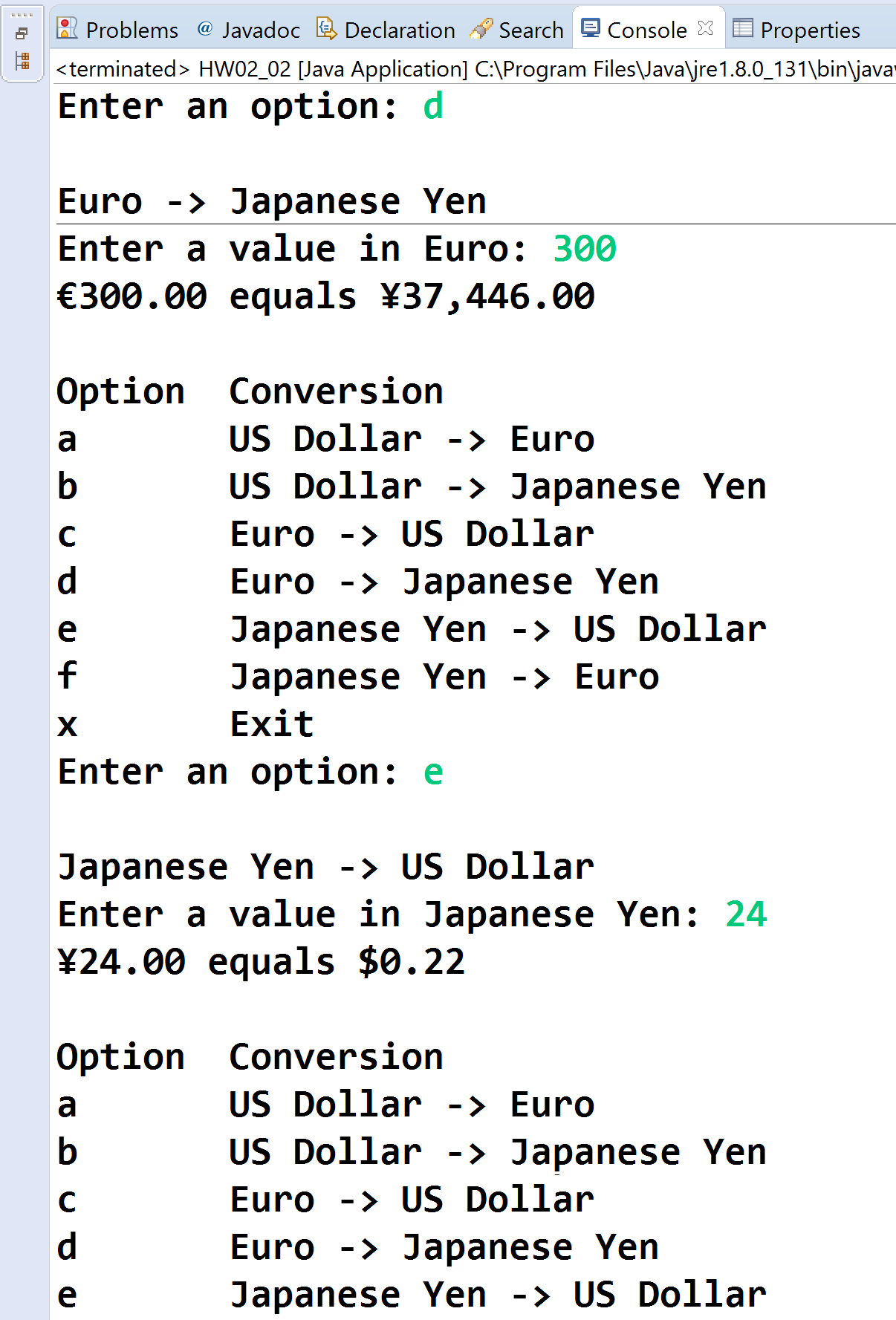
**System.out.println("\nEnd of Money Monarchs");**

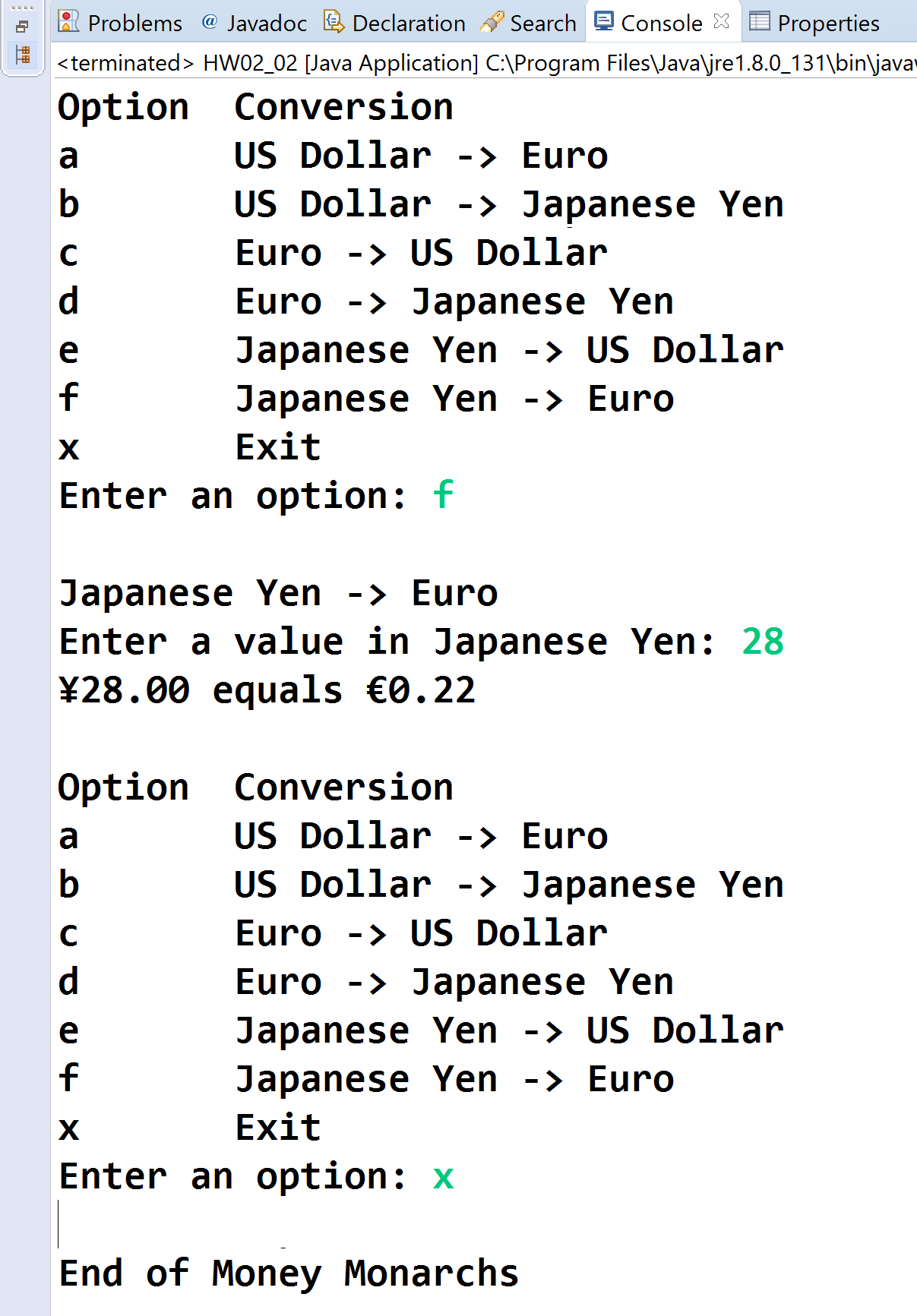
**}**

**}**









**3) [8 points]** You've been hired by *Readability Ravens* to write a Java console application that analyzes a paragraph for readability using the Coleman–Liau index. Prompt the user for a paragraph, and then calculate and show the following:

● Number of characters.

● Number of whitespace characters.

● Number of non-whitespace characters.

● Number of sentences – these are delimited by periods (.).

● Number of words – these are delimited by one or more spaces, commas, and periods.

● Number of word letters.

● Coleman–Liau index – this is calculated as follows:

1) Count the number of sentences.

2) Count the number of words.

3) Count the number of letters in words.

4) Calculate L = Letters ÷ Words × 100.

5) Calculate S = Sentences ÷ Words × 100.

6) Calculate CLI = 0.0588L - 0.296S - 15.8.

Format the output in two columns with the first column containing a label and the second column containing a value. Format the CLI to two decimal places. To test your program, use paragraph:

Existing computer programs that measure readability are based largely upon subroutines which estimate number of syllables, usually by counting vowels. The shortcoming in estimating syllables is that it necessitates keypunching the prose into the computer. There is no need to estimate syllables since word length in letters is a better predictor of readability than word length in syllables. Therefore, a new readability formula was computed that has for its predictors letters per 100 words and sentences per 100 words. Both predictors can be counted by an optical scanning device, and thus the formula makes it economically feasible for an organization such as the US Office of Education to calibrate the readability of all textbooks for the public school system.

You should get the following stats:

Characters: 765

Whitespace characters: 118

Non-whitespace characters: 647

Sentences: 5

Words: 119

Word letters: 639

L: 536.97

S: 4.20

CLI: 14.53

Use this paragraph for the last run:

Apollo 11 was the spaceflight that landed the first two humans on the Moon. Mission commander Neil Armstrong and pilot Buzz Aldrin, both American, landed the lunar module Eagle on July 20, 1969. Armstrong became the first to step onto the lunar surface six hours later on July 21. Aldrin joined him about 20 minutes later. They spent about two and a quarter hours together outside the spacecraft, and collected lunar material to bring back to Earth. Michael Collins piloted the command module Columbia alone in lunar orbit while they were on the Moon's surface. Armstrong and Aldrin spent just under a day on the lunar surface before rendezvousing with Columbia in lunar orbit.

**//======================================================================**

**//**

**// Title: Readability Ravens**

**// Course: CSC 3020**

**// Homework: 2-3**

**// Author: Dan Ouellette**

**// Date: 7 February 2018**

**// Description:**

**// This Java console application analyzes a paragraph for**

**// readability using the Coleman–Liau index. It prompts the**

**// user for a paragraph, and then calculates and shows the**

**// following:**

**// -Number of characters.**

**// -Number of whitespace characters.**

**// -Number of non-whitespace characters.**

**// -Number of sentences – these are delimited by periods (.).**

**// -Number of words – these are delimited by one or more**

**// spaces, commas, and periods.**

**// -Number of word letters.**

**// -Coleman–Liau index**

**//**

**//======================================================================**

**package wsu.HW02\_03;**

**// Import classes**

**import java.util.Scanner;**

**import java.util.StringTokenizer;**

**//======================================================================**

**// class HW02\_03**

**//======================================================================**

**public class HW02\_03**

**{**

**//------------------------------------------------------------------**

**// main**

**//------------------------------------------------------------------**

**public static void main(String[] args)**

**{**

**// Declare constants**

**final String WORD\_REGEXP = "[ ,.]+";**

**final String SENTENCE\_REGEXP = "[.]";**

**final String COLFMTS = "%-28s";**

**final String COLFMTD = "%,10d";**

**final String COLFMTF = "%,10.2f";**

**// Declare variables**

**Scanner keyboard = new Scanner(System.in);**

**String paragraph;**

**char c;**

**double CLI;**

**int characters;**

**double L;**

**int letters; // Count of letters in words**

**int whitespaceCharacters;**

**int nonWhitespaceCharacters;**

**double S;**

**String[] sentences;**

**String[] words;**

**// Show application header**

**System.out.println("Welcome to Readability Ravens");**

**System.out.println("-----------------------------\n");**

**// Prompt for and get paragraph**

**System.out.print("Enter a paragraph to analyze: ");**

**paragraph = keyboard.nextLine();**

**// Loop to count characters**

**characters = 0;**

**whitespaceCharacters = 0;**

**nonWhitespaceCharacters = 0;**

**for (int i = 0; i < paragraph.length(); i++)**

**{**

**c = paragraph.charAt(i);**

**if (Character.isWhitespace(c))**

**whitespaceCharacters = whitespaceCharacters + 1;**

**else**

**nonWhitespaceCharacters = nonWhitespaceCharacters + 1;**

**characters = characters + 1;**

**}**

**// Tokenize paragraph into sentences and words**

**sentences = paragraph.split(SENTENCE\_REGEXP);**

**words = paragraph.split(WORD\_REGEXP);**

**// Loop to count word Letters**

**letters = 0;**

**for (int i = 0; i < words.length; i++)**

**letters = letters + words[i].length();**

**// Calculate CLI**

**L = letters / (double) words.length \* 100;**

**S = sentences.length / (double) words.length \* 100;**

**CLI = 0.0588 \* L - 0.296 \* S - 15.8;**

**// Show paragraph and counts**

**System.out.printf(COLFMTS + COLFMTD + "%n", "Characters:",**

**characters);**

**System.out.printf(COLFMTS + COLFMTD + "%n",**

**"Whitespace characters:", whitespaceCharacters);**

**System.out.printf(COLFMTS + COLFMTD + "%n",**

**"Non-whitespace characters:", nonWhitespaceCharacters);**

**System.out.printf(COLFMTS + COLFMTD + "%n",**

**"Sentences:", sentences.length);**

**System.out.printf(COLFMTS + COLFMTD + "%n",**

**"Words:", words.length);**

**System.out.printf(COLFMTS + COLFMTD + "%n",**

**"Word letters:", letters);**

**System.out.printf(COLFMTS + COLFMTF + "%n",**

**"CLI:", CLI);**

**// Close keyboard**

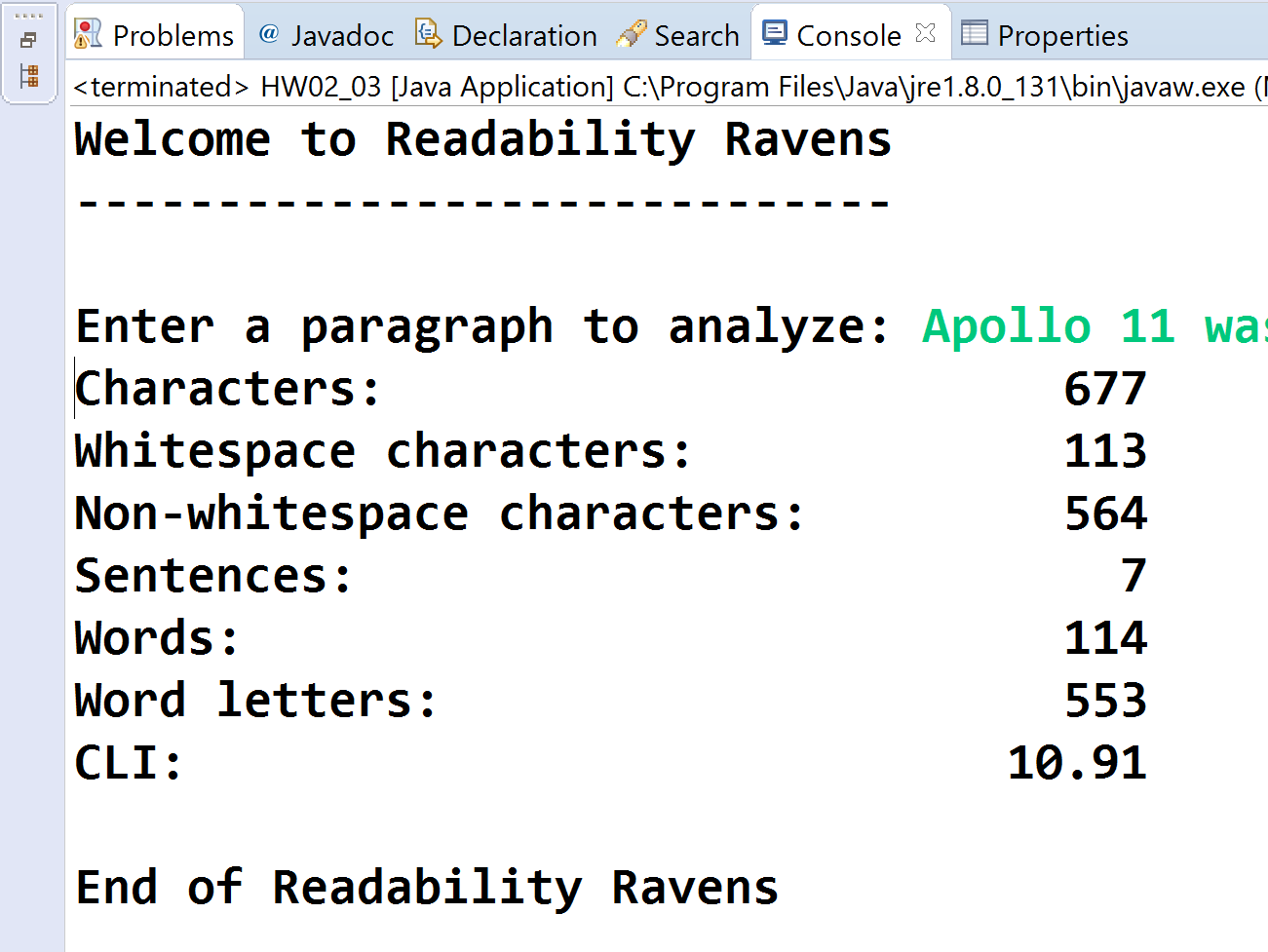
**keyboard.close();**

**// Show application close**

**System.out.println("\nEnd of Readability Ravens");**

**}**

**}**



\* **Copying-and-pasting application code to a Word document**

1) From the program editor window, press **CTRL-A** and press **CTRL-C**.

2) From within the Word document, press **CTRL-V**.

\*\* **Copying-and-pasting application output to a Word document**

1) From the Eclipse main screen, maximize the Console window.

2) From the Console window, press **ALT-PrintScreen**.

3) From within the Word document, press **CTRL-V**.