```java
/*
 * Programmer:  Chad Philip Johnson
 * Date Created:  Wednesday, October 03rd, 2012
 * Date of Last Modification:  Thursday, October 04th, 2012
 *
 * Description:
 * Assistant.class provides a few features that an incredibly
 * basic assistant application would have.  It offers the
 * following text-based tools and games:  a gas mileage
 * calculator, a to-do list composer, a grade calculator, a
 * number guessing game, and the classic grade-school game of
 * Hangman.
 */

import java.util.*;

/**
 * Assistant.class provides a few features that an incredibly
 * basic assistant application would have.  It offers the
 * following text-based tools and games:  a gas mileage
 * calculator, a to-do list composer, a grade calculator, a
 * number guessing game, and the classic grade-school game of
 * Hangman.
 *
 * @author Chad Philip Johnson
 * @version 1.0
 */

public class Assistant  {

    CinReader driverKeyboard;
    Random rand;

    // Number of different joke categories; increase this value for each new joke category added
    static final int NUMBER_OF_JOKE_CATEGORIES  = 3;
    // Max number of test scores allowed for the grade calculator
    static final int MAX_NUMBER_OF_TESTS        = 20;
    // Max number of tries when playing the number guessing game
    static final int NUMBER_GUESS_MAX_TRIES     = 10;
    // Max number of tries when playing Hangman
    static final int HANGMAN_MAX_TRIES          = 10;

    /**
      * Default constructor:
      * Instantiate CinReader.class and Random.class and associate them with the current instance of Assistant.class
      */
    Assistant() {

            // Instantiate CinReader.class and associate it with the current instance of Assistant.class
            this.driverKeyboard = new CinReader();
```

```
51              // Instantiate Random.class and associate it with the current instance of Assistant.class
52              this.rand           = new Random();
53
54      }
55
56      /**
57        * Print a random joke from a number of categories
58        */
59      public void randomJoke()    {
60
61          // Select a joke category randomly
62          int intSelectJokeCategory    = ( rand.nextInt( 1000 ) % NUMBER_OF_JOKE_CATEGORIES );
63          String strCurrentJoke = "";
64
65          switch( intSelectJokeCategory ) {
66              // Random jokes
67              case 0:
68                  strCurrentJoke  = getARandomStupidJoke();
69                  break;
70
71              // Dennis Miller Monday Night Football quotes
72              case 1:
73                  strCurrentJoke  = getADennisMillerMNFQuote() + "\n--Dennis Miller, Monday Night Football";
74                  break;
75
76              // Blind jokes
77              case 2:
78                  strCurrentJoke  = getABlindJoke();
79                  break;
80
81              // Default is random jokes
82              default:
83                  strCurrentJoke  = getARandomStupidJoke();
84
85          }
86
87          System.out.print( strCurrentJoke + "\n\n" );
88
89      }
90
91      /**
92        * Calculate miles per gallon based on user input
93        */
94      public void mileageCalculator() {
95
96          // Continue subroutine set to true by default
97          boolean blnContinue = true;
98          double dblMilesDriven, dblGallonsUsed;
99
100         System.out.print( "\nWelcome to the Mileage Calculator!\n\n" );
```

```
101
102         // Continue subroutine until user decides to exit to the main menu
103         while( blnContinue )    {
104
105             // Continue user input until a valid number is received (must be zero or greater)
106             while( true )   {
107
108                 System.out.print( "How many miles did you drive?  " );
109                 dblMilesDriven  = driverKeyboard.readDouble();
110
111                 if( dblMilesDriven < 0 )    {
112                     // Print error message and restart the loop
113                     System.out.println( "This number cannot be negative.  Please re-enter.");
114                     continue;
115
116                 } else {
117                     // Exit loop when a valid entry has been received
118                     break;
119
120                 }
121
122             };
123
124             // Continue user input until a valid number is received (must be greater than zero)
125             while( true )   {
126
127                 System.out.print( "How many gallons of gas did your car use to go that distance?  " );
128                 dblGallonsUsed  = driverKeyboard.readDouble();
129
130                 if( dblGallonsUsed <= 0 )   {
131                     // Print error message and restart the loop
132                     System.out.println( "This number cannot be negative or zero.  Please re-enter.");
133                     continue;
134
135                 } else {
136                     // Exit loop when a valid entry has been received
137                     break;
138
139                 }
140
141             }
142
143             // Compute and print mpg
144             System.out.printf( "You car averaged %.2f miles per gallon.\n\n", (float) (dblMilesDriven / dblGallonsUsed) );
145
146             // Prompt user to do another calculation
147             System.out.println( "Would you like to do another gas mileage calculation?  (y/n)" );
148
149             // Restart subroutine if true; exit to main menu if false
150             blnContinue = continueSubroutine();
```

```
151
152         }
153
154     }
155
156     /**
157       * Create a simple to-do list from user input
158       */
159     public void toDoList()  {
160
161         // Continue subroutine set to true by default
162         boolean blnContinue = true;
163         int intCounter      = 0;
164         LinkedList<String> myToDoList   = new LinkedList<String>();
165
166         System.out.print( "\nWelcome to the To-Do List Organizer!\n\n" );
167
168         // Continue subroutine until user decides to exit to the main menu
169         while( blnContinue )    {
170
171             // Accept a string as input from the user
172             System.out.println( "Please add an entry to your To-Do List:" );
173             myToDoList.add( driverKeyboard.readString() );
174
175             // Display current to-do list
176             showToDoList( myToDoList );
177
178             // Prompt user to make more entires
179             System.out.println( "Would you like to make another entry?  (y/n)" );
180             blnContinue = continueSubroutine();
181
182         }
183
184         // Print final to-do list
185         showToDoList( myToDoList );
186
187         // Warn user that the data is about to be lost
188         System.out.println( "You might want to write this list down because it's about to disappear!" );
189         // Wait till user presses enter
190         System.out.println( "Please press [Enter] to continue..." );
191         Scanner keyboard = new Scanner( System.in );
192         keyboard.nextLine();
193
194     }
195
196     /**
197       * Calculate an average grade from a user specified number of test scores
198       */
199     public void gradeCalculator()   {
200
```

```
201         // Continue subroutine set to true by default
202         boolean blnContinue = true;
203         int intNumberOfTests;
204         int intNumberOfPointsEarned, intMaxExamPoints;
205
206         System.out.print( "\nWelcome to the Grade Calculator!\n\n" );
207
208         // Continue subroutine until user decides to exit to the main menu
209         while( blnContinue )    {
210
211             // Reset values each time user decides to calculate a new average score
212             intNumberOfPointsEarned = 0;
213             intMaxExamPoints        = 0;
214
215             // Continue user input until a valid number is received (must be greater than zero)
216             while( true )   {
217
218                 System.out.print( "How many tests would you like to input to find your current average?  " );
219                 intNumberOfTests    = driverKeyboard.readInt();
220
221                 if( intNumberOfTests > 0 && intNumberOfTests <= MAX_NUMBER_OF_TESTS )   {
222                     // Exit loop when a valid value has been received
223                     break;
224
225                 } else if( intNumberOfTests > MAX_NUMBER_OF_TESTS ) {
226                     // Display warning message and restart loop if the number of tests to be entered exceeds the boundaries of
                        the program
227                     System.out.printf( "I'm sorry, you can only input a maximum of %d different test scores.\n",
                        MAX_NUMBER_OF_TESTS );
228                     continue;
229
230                 } else if( intNumberOfTests <= 0 )  {
231                     // Display warning message and restart loop if an invalid value has been received
232                     System.out.println( "I'm sorry, the number of tests cannot be negative or zero." );
233                     continue;
234
235                 }
236
237             }
238
239             for( int i = 1; i <= intNumberOfTests; i++ )    {
240
241                 int intTempPointsEarned, intTempMaxExamPoints;
242
243                 // Continue user input until a valid number is received (must be greater than or equal to zero)
244                 while( true )   {
245                     System.out.printf( "Please input the points earned for exam #%d:  ", i );
246                     intTempPointsEarned = driverKeyboard.readInt();
247
248                     if( intTempPointsEarned >= 0 )  {
```

```
                        // Exit loop when a valid value has been received
                        break;

                    } else {
                        // Display warning message and restart loop if an invalid value has been received
                        System.out.println( "I'm sorry (not really, you moron), the points earned cannot be negative." );
                        continue;

                    }

                }

                // Tally the total points earned with each pass
                intNumberOfPointsEarned += intTempPointsEarned;

                // Continue user input until a valid number is received (must be greater than or equal to zero)
                while( true )   {
                    System.out.printf( "Please input the maximum points for exam #%d:  ", i );
                    intTempMaxExamPoints    = driverKeyboard.readInt();

                    if( intTempMaxExamPoints >= 0 ) {
                        // Exit loop when a valid value has been received
                        break;

                    } else {
                        // Display warning message and restart loop if an invalid value has been received
                        System.out.println( "You idiot, the points earned cannot be negative.");

                    }

                }

                // Tally the total points available with each pass
                intMaxExamPoints    += intTempMaxExamPoints;

            }

            // Perform average calculation, print result and offer an encouraging message
            System.out.printf( "\nYou earned %d out of %d total points which represents an average of %.2f%%.  Good job!\n\n",
            intNumberOfPointsEarned, intMaxExamPoints, (((float) intNumberOfPointsEarned / (float) intMaxExamPoints ) * 100.0f) );

            // Prompt user to perform another grade calculation
            System.out.println( "Would you like to calculate another average?  (y/n)" );
            blnContinue = continueSubroutine();

        }

        System.out.println( "Thanks for using the Grade Calculator!" );

    }
```

```java

    /**
      * Play a number guessing game
      */
    public void numberGame()    {

        // Continue subroutine set to true by default
        boolean blnContinue = true;
        int intUserInput;
        int intSecretNumber;

        System.out.print("\n\nWelcome to the Number Guessing Game!\n\n");
        System.out.println("You must guess the correct number between 1 and 100 in ten tries.");

        // Continue subroutine until user decides to exit to the main menu
        while( blnContinue )    {

            // Find a random number between 1 and 100
            intSecretNumber = rand.nextInt(100) + 1;

            System.out.print( "\nThe secret number is between 1 and 100.  What do you think it is?  " );

            // Continue game until max tries have all been used up
            for( int i = 1; i <= NUMBER_GUESS_MAX_TRIES; i++ )  {

                // Continue user input until a valid number is received (must be between 1 and 100)
                while( true )   {
                    intUserInput    = driverKeyboard.readInt();

                    if( intUserInput > 100 || intUserInput <= 0 )   {
                        System.out.println( "Oops!  The number must be between 1 and 100.  Try again!" );
                        continue;

                    } else {
                        // Exit loop when a valid value has been received
                        break;

                    }

                }

                // If guessed number matches secret number, print victory message and exit game loop
                if( intUserInput == intSecretNumber )   {
                    System.out.printf( "\nCongratulations!  You win!  You found the secret number %d in %d tries!\n\n",
                    intSecretNumber, i );
                    break;

                } else {

                    // If guessed number is too high, print "too high" message
```

```
347                     if( intUserInput > intSecretNumber )    {
348                         System.out.printf( "\nThe number %d is too high!  Guess lower!\n", intUserInput );
349
350                     // If guessed number is too low, print "too low" message
351                     } else if( intUserInput < intSecretNumber ) {
352                         System.out.printf( "\nThe number %d is too low!  Guess higher!", intUserInput );
353
354                     }
355
356                     // Print the number of tries remaining
357                     if( (NUMBER_GUESS_MAX_TRIES - i) > 1 )  {
358                         System.out.printf( "\nYou have only %d tries left!\n", (NUMBER_GUESS_MAX_TRIES - i) );
359
360                     } else if( (NUMBER_GUESS_MAX_TRIES - i) == 1 ) {
361                         System.out.println( "\nYou have only 1 try left!\n");
362
363                     } else {
364                         System.out.printf( "\nOh no!  You're all out of tries!  The secret number was %d.\n", (
                         NUMBER_GUESS_MAX_TRIES - i), intSecretNumber );
365
366                     }
367
368                 }
369
370             }
371
372             // Print game over message and prompt user to play again
373             System.out.println("Game Over.  Would you like to play again?  (y/n)");
374             blnContinue = continueSubroutine();
375
376         }
377
378         System.out.println( "Thanks for playing the number guessing game!" );
379
380     }
381
382     /**
383       * Play a game of Hangman
384       */
385     public void hangmanGame()   {
386
387         // Continue subroutine set to true by default
388         boolean blnContinue = true;
389         char chrUserInput;
390
391         System.out.print("\n\nWelcome to Hangman!\n");
392
393         // Continue subroutine until user decides to exit to the main menu
394         while( blnContinue )    {
395
```

```
396             // Retrieve random word for the current game
397             char[] chrCurrentWord   = getAWordForHangman().toCharArray();
398             // Create a "blank" word of the same length as the secret word
399             char[] chrEmptyWord     = new char[chrCurrentWord.length];
400
401             // Blank out the "blank" word with underscores
402             for( int i = 0; i < chrCurrentWord.length; i++ )    {
403                 chrEmptyWord[i] = '_';
404
405             }
406
407             System.out.print("\nGuess the secret word in ten tries or less!\n");
408
409             int intNumberOfTries = 0;
410             boolean blnLetterExists;
411             while( true )   {
412
413                 // Guessed letter does not exist, by default
414                 blnLetterExists = false;
415
416                 // Show player's current progress with each pass
417                 hangmanShowPlayerProgress( chrEmptyWord, chrCurrentWord.length, intNumberOfTries );
418
419                 // Continue user input until a valid number is received (must be between a-z or A-Z)
420                 while( true )   {
421
422                     System.out.print( "What letter would you like to guess?  ");
423                     chrUserInput    = driverKeyboard.readChar();
424
425                     // Valid input of ASCII set a-z
426                     if( chrUserInput >= 97 && chrUserInput <= 122 ) {
427                         break;
428
429                     // Valid input of ASCII set A-Z, convert to lowercase
430                     } else if( chrUserInput >= 65 && chrUserInput <= 90 )   {
431                         chrUserInput    += 32;
432                         break;
433
434                     // Invalid character:  must be a letter
435                     } else {
436                         System.out.printf( "The character %c is invalid.  Please retry.\n", chrUserInput );
437                         continue;
438
439                     }
440
441                 }
442
443                 // Compare guessed letter with all letters in secret word
444                 for( int i = 0; i < chrCurrentWord.length; i++ )    {
445
```

```
446                     // If the guessed letter matches any letters of the secret word, update the "blank" word with that character
                        at the same array position
447                     // (test fails if the same correct letter is used more than once)
448                     if( chrCurrentWord[i] == chrUserInput && chrEmptyWord[i] != chrUserInput )  {
449                         // Assign character value to current position of "blank" word
450                         chrEmptyWord[i] = chrUserInput;
451                         // Set flag that the user has guessed a correct letter
452                         blnLetterExists = true;
453
454                     }
455
456                 }
457
458                 // If the user has not guessed correctly increment the total number of attempts by one
459                 if( blnLetterExists == false )  {
460                     intNumberOfTries++;
461
462                 }
463
464                 // Print victory message if the user has successfully guessed the word
465                 if( Arrays.equals( chrCurrentWord, chrEmptyWord ) ) {
466                     System.out.printf( "Congratulations, you guessed the correct word and had %d tries remaining!\n\n", (
                        HANGMAN_MAX_TRIES - intNumberOfTries) );
467                     hangmanShowPlayerProgress( chrEmptyWord, chrCurrentWord.length, intNumberOfTries );
468                     System.out.printf( "The secret word was \"%s\".\n\n", new String( chrCurrentWord ) );
469                     break;
470
471                 }
472
473                 // Print "successful" message when a letter has been guessed; print the number of tries left
474                 if( blnLetterExists == true )   {
475
476                     System.out.printf( "Way to go!  The letter \"%c\" appears in the secret word.\n", chrUserInput );
477
478                     if( (HANGMAN_MAX_TRIES - intNumberOfTries) > 1 )    {
479                         System.out.printf( "You still have %d tries left!\n", (HANGMAN_MAX_TRIES - intNumberOfTries) );
480
481                     } else {
482                         System.out.println( "Oh no!  You have only one try left!" );
483
484                     }
485
486                 }
487
488                 // Print "unsuccessful" message when a letter has not been guessed; increment the number of attempts and print
                    the number of tries left
489                 if( blnLetterExists == false ) {
490
491                     if( (HANGMAN_MAX_TRIES - intNumberOfTries) > 1 ) {
492                         System.out.printf( "No luck on that one.  You have %d tries left!\n", (HANGMAN_MAX_TRIES -
```

```
                        intNumberOfTries) );
493
494                     } else if( (HANGMAN_MAX_TRIES - intNumberOfTries) == 1 )    {
495                         System.out.println( "Now you've done it... You have only one try left!" );
496
497                     // Exit game loop when the number of tries reaches zero
498                     } else {
499                         System.out.println( "You lose!  Game Over!");
500                         hangmanShowPlayerProgress( chrEmptyWord, chrCurrentWord.length, intNumberOfTries );
501                         System.out.printf( "The secret word was \"%s\".\n\n", new String( chrCurrentWord ) );
502                         break;
503
504                     }
505
506                 }
507
508             }
509
510             // Prompt user to play again
511             System.out.println( "Would you like to play again?  (y/n)" );
512             blnContinue = continueSubroutine();
513
514         }
515
516     }
517
518     /**
519       * Return a joke from the list of random stupid jokes
520       */
521     public String getARandomStupidJoke()    {
522
523         String[] randomStupidJokes  = new String[]  {
524             "I have the power to channel my imagination into ever-soaring levels of\nsuspicion and paranoia." ,
525             "I assume full responsibility for my actions, except the ones that are someone\nelse's fault." ,
526             "I no longer need to punish, deceive, or compromise myself. Unless, of course,\nI want to stay employed." ,
527             "Having control over myself is nearly as good as having control over others." ,
528             "My intuition nearly makes up for my lack of good judgment." ,
529             "I honor my personality flaws, for without them I would have no personality at\nall." ,
530             "I am grateful that I am not as judgmental as all those censorious,\nself-righteous people around me." ,
531             "I need not suffer in silence while I can still moan, whimper, and complain." ,
532             "As I learn the innermost secrets of the people around me, they reward me in\nmany ways to keep me quiet." ,
533             "When someone hurts me, forgiveness is cheaper than a lawsuit. But not nearly\nas gratifying." ,
534             "The first step is to say nice things about myself. The second, to do nice\nthings for myself. The third, to find
                someone to buy me nice things.",
535             "As I learn to trust the universe, I no longer need to carry a gun." ,
536             "I am at one with my duality.",
537             "Blessed are the flexible, for they can tie themselves into knots." ,
538             "Only a lack of imagination saves me from immobilizing myself with imaginary\nfears." ,
539             "Does my quiet self-pity get to you or should I move up to incessant nagging?" ,
540             "Today I will gladly share my experience and advice, for there are no sweeter\nwords than \"I told you so.\"" ,
```

```
            "False hope is nicer than no hope at all.",
            "A good scapegoat is nearly as welcome as a solution to the problem.",
            "Just for today, I will not sit in my living room all day watching TV. Instead\nI will move my TV into the bedroom.",
            "Who can I blame for my own problems? Give me just a minute... I'll find someone.",
            "The complete lack of evidence is the surest sign that the conspiracy is working.",
            "I am learning that criticism is not nearly as effective as sabotage.",
            "Becoming aware of my character defects leads me to the next step - blaming my\nparents.",
            "I will find humor in my everyday life by looking for people I can laugh at.",
            "The next time the universe knocks on my door, I will pretend I am not home.",
            "To have a successful relationship I must learn to make it look like I'm giving\nas much as I'm getting.",
            "I am willing to make the mistakes if someone else is willing to learn from them."
        };

        // Return a random string from the String array
        return randomStupidJokes[ rand.nextInt( randomStupidJokes.length ) ];

    }

    /**
      * Return a quote from the list of Dennis Miller Monday Night Football sayings
      */
    public String getADennisMillerMNFQuote()    {

        String[] dennisMillerMNFQuotes  = new String[]  {
            "Of *course* he needs to renegotiate his salary -- the guy buys more snow than\nSeward did when he bought Alaska
            from the Russians.",
            "I haven't seen anyone rely on the ground game this much since the battle of\nVerdun.",
            "The quarterback's spending so much time behind the center that he may\njeopardize his right to lead a Boy Scout
            troop.",
            "I've seen women pee standing up with better aim.",
            "Somebody call Janet Reno -- I think I just saw Donato dragging Doug Flutie\ninto a locker room closet!",
            "That field goal attempt was so far to the left it nearly decapitated Lyndon\nLaRouche.",
            "I haven't seen someone so overmatched since Mike Tyson tried to recite the\nalphabet.",
            "Hey, Cunningham -- Andy Warhol called. You're at 14:55 and we're tickin'\nbig-time here, Chachi.",
            "He lasted about as long as the dessert tray at Rosie O'Donnell's house.",
            "Hey Deion, Bubbelah -- maybe you'd better pay a little less attention to\nthose unfairly Draconian salary caps that
            only allowed you to acquire four of\nthe five remaining 1932 Aston Martins still in road-worthy condition
            after\nyou'd paid for life's little necessities like hookers and weed, get your\nmedulla oblongata out of your
            duodenum for a few milliseconds, and make a\ntackle or two, okay, Babe?",
            "When the hell is Warren Moon going to retire? I mean, this guy is older than\nthe cuneiform in Nebuchadnezzar's
            tomb.",
            "That punt was higher than Marion Berry on a fact-finding tour of Cartagena.",
            "Nervous? He's tighter than Pat Buchanan's sphincter muscle at a 4th of July\nsoiree on Fire Island.",
            "Warner had more hands in his face than an OB-GYN delivering Vishnu's\ntriplets!",
        };

        // Return a random string from the String array
        return dennisMillerMNFQuotes[ rand.nextInt( dennisMillerMNFQuotes.length ) ];

    }
```

```
585
586         /**
587           * Return a joke from the list of blind jokes
588           */
589         public String getABlindJoke()   {
590
591             String[] blindJokes = new String[]  {
592                 "How do you discipline a blind kid?  You move the furniture around.",
593                 "A blind man walks into a store with his seeing eye dog. All of a sudden, he\npicks up the leash and begins swinging
                    the dog over his head. The manager runs\nup to the man and asks, \"What are you doing?!!\" The blind man
                    replies,\n\"Just looking around.\"",
594                 "There are 2 blonds sitting on a porch in Kansas looking at the moon.  One\nblond says to the other, \"which do you
                    think is closer? The moon or Texas?\"\nThe other blond says \"Duh!  Can you see Texas?\"",
595                 "Why don't blind people skydive?  It scares the heck out of the dog.",
596                 "Marriage is love. Love is blind. Therefore, marriage is an institution for\nthe blind.",
597                 "What do you call a blind rabbit sitting on your face?  An unsightly facial\nhare!",
598                 "Remember:  Pirates with two eye patches are not twice as deadly.",
599                 "Why don't blind people ever watch where they're going?",
600                 "Why is it that the blind leading the blind always have so many places to go?",
601                 "Blind people are fun to trip.",
602                 "Blind people make wonderful moving targets, especially for paintball practice.",
603                 "What did one blind man say to the other blind man?  \"It sure is dark\ntoday.\"  To which, the other blind man
                    replied, \"Yep... sure is...\"",
604                 "Don't say \"It's such a beautiful day today!\" to a blind person.  It is\ninconsiderate and cruel.  Instead do the
                    right thing and say, \"It isn't a\nvery nice day today.\"",
605                 "Blind people know that life isn't fair... much more than most.",
606                 "Blind people are cowards.  I've never met a blind man that could look me\nin the eyes.",
607                 "Yes, sunglasses are an acceptable gift to give a blind person on his\nbirthday--the darker the shades the better.",
608                 "Blind people are allowed to run with scissors.",
609                 "Yes, blind people like blind jokes too, but only when they're wearing\nsunglasses.",
610                 "Blind people can't read these jokes because they aren't written in\nbraille.",
611             };
612
613             // Return a random string from the String array
614             return blindJokes[ rand.nextInt( blindJokes.length ) ];
615
616         }
617
618         /**
619           * Return a random word for a game of Hangman
620           */
621         private String getAWordForHangman() {
622
623             String[] wordsForHangman    = new String[]  {
624                 "kindergarten",
625                 "physics",
626                 "calculus",
627                 "computer",
628                 "chemistry",
629                 "biology",
```

```
            "headache",
            "programming",
            "compost",
            "recycle",
            "prius",
            "solar",
            "environmentalist",
            "battery",
            "lead",
            "yuppy",
            "object",
            "orient",
            "beer",
            "confession",
            "elementary",
            "electricity",
            "magnetism",
            "potential",
            "energy",
            "field",
            "organic",
            "potato",
            "tomato",
            "tortilla",
            "quesadilla",
            "enchilada",
            "burrito",
            "salsa",
            "frijoles",
            "spanish",
            "tequila",
            "wine",
            "differential",
            "equation",
            "architecture",
            "agression",
            "foreclosure",
            "dream",
            "tiger",
            "zebra",
            "africa",
            "tostada",
            "finish",
        };

        // Return a random string from the String array
        return wordsForHangman[ rand.nextInt( wordsForHangman.length ) ];

    }

```

```
    /**
      * Prompt user whether he/she would like to continue executing a subroutine
      */
    private boolean continueSubroutine()    {

        // Prompt user for character input
        char charUserInput  = driverKeyboard.readChar();

        // Resume subroutine if input does not equal 'n' or 'N'
        if( !( charUserInput == 'n' || charUserInput =='N' ) )  {
            return true;

        // Quit subroutine if input equals 'n' or 'N'
        } else {
            return false;

        }

    }

    /**
      * Display the current to-do list to the user
      */
    private void showToDoList( LinkedList<String> myToDoList )  {

        System.out.print( "\nHere is your current To-Do List:\n" );
        int i   = 1;
        // Display current to-do list with a leading number
        for( String readThrough : myToDoList )  {
            System.out.printf( "[%d] %s\n", i, readThrough );
            i++;

        }

        System.out.println();

    }

    /**
      * Show player progress (the correct guesses) in a game of hangman
      */
    private void hangmanShowPlayerProgress( char[] chrEmptyWord, int intCurrentWordLength, int intCurrentTry )  {

        if( HANGMAN_MAX_TRIES == 10 )   {

            switch( intCurrentTry ) {
                case 0:
                    System.out.println( "  *------------*    " );
                    System.out.println( "  |                 " );
                    System.out.println( "  |                 " );
```

```
730                     System.out.println( "  |                " );
731                     System.out.println( "  |                " );
732                     System.out.println( "  |                " );
733                     System.out.println( "  |                " );
734                     System.out.println( "--------------------" );
735                     System.out.print("\n\t");
736                     break;
737 
738                 case 1:
739                     System.out.println( "  *------------*    " );
740                     System.out.println( "  |            |    " );
741                     System.out.println( "  |                 " );
742                     System.out.println( "  |                 " );
743                     System.out.println( "  |                 " );
744                     System.out.println( "  |                 " );
745                     System.out.println( "  |                 " );
746                     System.out.println( "--------------------" );
747                     System.out.print("\n\t");
748                     break;
749 
750                 case 2:
751                     System.out.println( "  *------------*    " );
752                     System.out.println( "  |            |    " );
753                     System.out.println( "  |            O    " );
754                     System.out.println( "  |                 " );
755                     System.out.println( "  |                 " );
756                     System.out.println( "  |                 " );
757                     System.out.println( "  |                 " );
758                     System.out.println( "--------------------" );
759                     System.out.print("\n\t");
760                     break;
761 
762                 case 3:
763                     System.out.println( "  *------------*    " );
764                     System.out.println( "  |            |    " );
765                     System.out.println( "  |            O    " );
766                     System.out.println( "  |            |    " );
767                     System.out.println( "  |                 " );
768                     System.out.println( "  |                 " );
769                     System.out.println( "  |                 " );
770                     System.out.println( "--------------------" );
771                     System.out.print("\n\t");
772                     break;
773 
774                 case 4:
775                     System.out.println( "  *------------*    " );
776                     System.out.println( "  |            |    " );
777                     System.out.println( "  |            O    " );
778                     System.out.println( "  |           -|    " );
779                     System.out.println( "  |                 " );
```

```
780                     System.out.println( "  |                " );
781                     System.out.println( "  |                " );
782                     System.out.println( "--------------------" );
783                     System.out.print("\n\t");
784                     break;
785 
786                 case 5:
787                     System.out.println( "  *------------*    " );
788                     System.out.println( "  |            |    " );
789                     System.out.println( "  |            O    " );
790                     System.out.println( "  |           -|-   " );
791                     System.out.println( "  |                " );
792                     System.out.println( "  |                " );
793                     System.out.println( "  |                " );
794                     System.out.println( "--------------------" );
795                     System.out.print("\n\t");
796                     break;
797 
798                 case 6:
799                     System.out.println( "  *------------*    " );
800                     System.out.println( "  |            |    " );
801                     System.out.println( "  |            O    " );
802                     System.out.println( "  |           -|-   " );
803                     System.out.println( "  |            |    " );
804                     System.out.println( "  |                " );
805                     System.out.println( "  |                " );
806                     System.out.println( "--------------------" );
807                     System.out.print("\n\t");
808                     break;
809 
810                 case 7:
811                     System.out.println( "  *------------*    " );
812                     System.out.println( "  |            |    " );
813                     System.out.println( "  |            O    " );
814                     System.out.println( "  |           -|-   " );
815                     System.out.println( "  |            |    " );
816                     System.out.println( "  |           /    " );
817                     System.out.println( "  |                " );
818                     System.out.println( "--------------------" );
819                     System.out.print("\n\t");
820                     break;
821 
822                 case 8:
823                     System.out.println( "  *------------*    " );
824                     System.out.println( "  |            |    " );
825                     System.out.println( "  |            O    " );
826                     System.out.println( "  |           -|-   " );
827                     System.out.println( "  |            |    " );
828                     System.out.println( "  |          _/    " );
829                     System.out.println( "  |                " );
```

```
                    System.out.println( "--------------------" );
                    System.out.print("\n\t");
                    break;

                case 9:
                    System.out.println( "  *------------*    " );
                    System.out.println( "  |            |    " );
                    System.out.println( "  |            O    " );
                    System.out.println( "  |           -|-   " );
                    System.out.println( "  |            |    " );
                    System.out.println( "  |          _/ \\   " );
                    System.out.println( "  |                 " );
                    System.out.println( "--------------------" );
                    System.out.print("\n\t");
                    break;

                case 10:
                    System.out.println( "  *------------*    " );
                    System.out.println( "  |            |    " );
                    System.out.println( "  |            O    " );
                    System.out.println( "  |           -|-   " );
                    System.out.println( "  |            |    " );
                    System.out.println( "  |          _/ \\_  " );
                    System.out.println( "  |                 " );
                    System.out.println( "--------------------" );
                    System.out.print("\n\t");
                    break;

            }

        }

        for( int i = 0; i < intCurrentWordLength; i++ ) {
            System.out.printf( "%c ", chrEmptyWord[i] );

        }

        System.out.print( "\n\n" );

    }

}
```