1. Homework 3

Posted: May/10/2018
Due: September/16/2018 24.00 - This is a change

All homework solutions are due September/16/2018 24.00 - This is a change. I recommend to submit at least one version of all homework solutions long before due date.

Note the change of the deadline"

1.1. Homework 3.1 (10 Points)

Objective: Design, implementation, and testing of algorithm

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Implement from the following the following methods:::

```
static double   abs(double a)
static int      max(int a, int b)
static double   sqrt(double a)
```

and a test method for each method you implement. You can not use any method from Math class.

Explanation:
The following be useful how to calculate an approximation of *sqrt(x)*, for $x \geq 0$. Your aproximation of *sqrt(x)* for $x \geq 0$ will calculate

$$x_{n+1} \textit{ using } x_n$$

and the approximation will get closer to *sqrt(x)* which each step. In other words:

$$|{x_{n+1}}^2 - x| < |{x_n}^2 - x|$$

Assuming you calculate *sqrt(a)* using a converging series you can stop your calculation if the following is true:

$$|x_{n+1} - x_n| < \varepsilon \text{ for } \varepsilon > 0; \text{ for } \textit{example } \varepsilon = 0.001$$

This might also become useful. The following snippet shows an idea using the *information*.

```
static void   testSqrt()    {
    if ( 0 != sqrt(0) )
            System.out.println("Test 1: sqrt failed");
    if ( 0 != sqrt(-0) )
            System.out.println("Test 2: sqrt failed");
    if ( Double.NaN == sqrt(-1) )
            System.out.println("Test 3: sqrt failed");
    if ( Double.NaN == sqrt(Double.NaN) )
            System.out.println("Test 4: sqrt failed");
    double result;
    double aDouble;
    double theDoubles[] = {1, 2, 3, 4, 5 };
    for ( int index = 0; index < theDoubles.length; index ++ )      {
```

```
            result = sqrt(theDoubles[index]);
            if ( abs( result * result - theDoubles[index] ) > epsilon )
                    System.out.println("Test 5: sqrt failed: " + ( result * result - theDo
    }
}
```

An example output for *x_n, n = 5, epsilon = 0.000000001*:

```
x_n = 2.25
x_n = 2.236111111111111
x_n = 2.2360679779158037
x_n = 2.23606797749979
sqrt(5.0) ~= 2.23606797749979
    (2.23606797749979    ^2 = 5.000000000000001)
    (error  =             8.881784197001252E-16)
```

The implementation of *max(int a, int b)* should be straight forward. An idea for the test case for *max(int a, int b)*:

```
//              x y
//              - +
//              + -     -> Associativity
//              + +
//              - -
//              = =
//              0 -
//              0 +
```

Your Work:
You have to implement the 3 methods including test methods.

Requirements:
You have to name your class *TestMath.java*.

Example:

An example of a solution execution:

```
% java TestMath
```

Submission:

```
% ssh glados.cs.rit.edu # or use queeg.cs.rit.edu if glados is down
# password
# go to the directory where your solution is ...
% try hpb-grd lab3-1 'All files required'
# you can see if your submission was successful:
# try -q hpb-grd lab3-1
```

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1        class TestMath {
 2
 3          static double epsilon = 0.000000001;
 4
 5          static double   abs(double a)       {
 6              return ( a > 0.0 ? a : -1 * a );
 7          }
 8          static int      floorDiv(int x, int y)      {
 9              return 0;
10          }
11          static int      max(int a, int b)   {
12              return ( a > b ? a : b );
13          }
14          static double   sqrt(double a)      {
15              if ( a == Double.POSITIVE_INFINITY )
16                      return Double.POSITIVE_INFINITY;
17              if ( a == Double.NaN )
18                      return Double.NaN;
19              if ( a < 0.0 )
20                      return Double.NaN;
21              if ( ( a == 0.0 ) || ( a == -0.0 ) )
22                      return 0;
23
24              double xN = a / 2;
25              while ( abs(xN * xN - a )  > epsilon )  {
26                      xN = 0.5 * ( xN + a / xN );
27                      // System.out.println("x_n = " + xN); for hw creation
28              }
29              return xN;
30           }
31
32       //      test case
33       //              - +
34       //              + -     -> Associativity
35       //              + +
36       //              - -
37       //              = =
38       //              0 -
39       //              0 +
40          static void   testMax()     {
41                  int a = 1;
42                  int b = 2;
43              if ( b != max(a, b) )
44                      System.out.println("Test 1: max failed");
45              if ( b != max(-1 * a, b) )
46                      System.out.println("Test 2: max failed");
47              if ( a != max(a, -1 * b) )
48                      System.out.println("Test 3: max failed");
49              if ( -1 * a != max(-1 * a, -1 * b) )
50                      System.out.println("Test 4: max failed");
51              if ( a != max(a, a) )
52                      System.out.println("Test 5: max failed");
53              if ( 0 != max(0, -1 * a ) )
54                      System.out.println("Test 6: max failed");
```

```
55              if ( 0 != max(-1 * a , 0 ) )
56                      System.out.println("Test 7: max failed");
57          }
58          static void   testSqrt()    {
59              if ( 0 != sqrt(0) )
60                      System.out.println("Test 1: sqrt failed");
61              if ( 0 != sqrt(-0) )
62                      System.out.println("Test 2: sqrt failed");
63              if ( Double.NaN == sqrt(-1) )
64                      System.out.println("Test 3: sqrt failed");
65              if ( Double.NaN == sqrt(Double.NaN) )
66                      System.out.println("Test 4: sqrt failed");
67              double result;
68              double aDouble;
69              double theDoubles[] = {1, 2, 3, 4, 5 };
70              // double theDoubles[] = { 5 };
71              for ( int index = 0; index < theDoubles.length; index ++ )      {
72                      result = sqrt(theDoubles[index]);
73                      if ( abs( result * result - theDoubles[index] ) > epsilon )
74                              System.out.println("Test 5: sqrt failed: " + ( result * re
75                      else
76                              System.out.println("sqrt(" + theDoubles[index] + ") ~= " +
77                              System.out.println("\t(" + result + "\t^2 = "  + result *
78                              System.out.println("\t(error  =\t\t     "  + abs ( result
79
80              }
81          }
82          static void   testAbs()     {
83                  double aDouble = 1;
84              if ( aDouble != abs(aDouble) )
85                      System.out.println("Test 1: abs failed");
86              if ( aDouble != abs(-1 * aDouble) )
87                      System.out.println("Test 2: abs failed");
88              if ( 0 != abs( 0) )
89                      System.out.println("Test 3: abs failed");
90          }
91          public static void main( String args[] ) {
92
93              testAbs();
94              testMax();
95              testSqrt();
96          }
97      }
```

Source Code: Src/23_sol/TestMath.java

1.2. Homework 3.2 (10 Points)

Objective: Understanding of Scanner and Pattern

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the

grading session

Homework Description:

Given is a file, each line of the file has a single word. A word consist of printable, visible characters. You have to read the file and print the word, if the word has specific properties, like being a palindrome. You have to the pattern class to determine if a word has a specific property, and you have to write your own method to determine if a word has a specific property.

You have to test each word, i.e. each line, for the following properties:

- ´a' is part of the word.
- Palindrome anchored at the beginning and end of line.
- Include a palindrome which is 2 characters long.
- Include a palindrome which is 3 characters long.
- The word has at least one 'a' in it.
- The word consist only of 'a's or 'b's.
- ´a's or 'b's can not be part of the word.
- The word is == '.'.
- The word includes a '.'.

Explanation:
Let assume the file is:

```
a
aba
abb
cd
Hello World.
```

- ´a' has an 'a' as part of the word.
- ´a' is a palindrome anchored at the beginning and end of line
- ´abb' includes a palindrome which is 2 characters long
- ´cd' is a word which does not include 'a's or 'b's.
- ´.' is == to '.'.

Your Work:
You have to use the class to determine if a word fulfills a particular requirement. You have to write your own method for each property in order to determine if a word fulfills a particular property.

Your own methods should use the should be efficient.

An 'mini' example of how to use the Pattern class:

```
 1
 2      import java.util.Scanner;
 3      import java.io.File;
 4      import java.util.regex.Pattern;
 5
 6
 7      public class RegExTest {
 8          public static void testPattern(String regEx, String aString, String comment )
 9              if ( Pattern.matches(regEx, aString) )  {
10                      System.out.println("RegEx Test:");
11                      System.out.println("\tInput line: " + aString);
```

```
12                      System.out.println("\t" + regEx );
13                      System.out.println("\t" + comment);
14              }
15          }
16          public static void testOwn1(String aString, String comment ) {
17              if ( aString.indexOf('a') >= 0 )        {
18                      System.out.println("Own Test:");
19                      System.out.println("\tInput line: " + aString);
20                      System.out.println("\t" + comment);
21              }
22          }
23          public static void main( String[] args ) {
24              String aString;
25              Pattern aPattern;
26
27              aString = "a";
28              testPattern("[a]+", aString, "string has at least one a ");
29              testOwn1(aString, "string has at least one a ");
30          }
31      }
```

Source Code: Src/23/RegExTest.java

```
% java RegExTest
RegEx Test:
    Input line: a
    [a]+
    string has at least one a
Own Test:
    Input line: a
    string has at least one a
```

Requirements:
You have to name your class *RegExTest.java*.

Submission:

```
% ssh glados.cs.rit.edu # or use queeg.cs.rit.edu if glados is down
# password
# go to the directory where your solution is ...
% try hpb-grd lab3-2 'All files required'
# you can see if your submission was successful:
# try -q hpb-grd lab3-2
```

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
1       import java.util.Scanner;
2       import java.io.File;
3       import java.util.regex.Pattern;
4       /*
5        * explain how the code was developed
6        */
```

```
 7
 8      public class RegExTest {
 9          public static void testPattern(String regEx, String aString, String comment )
10              // System.out.println(aString);
11              if ( Pattern.matches(regEx, aString) )  {
12                      System.out.println("Input line: " + aString);
13                      System.out.println("\t" + regEx );
14                      System.out.println("\t" + comment);
15              }
16          }
17          public static void specialTestPattern(boolean anchored, String aString, String
18              String pattern = aString.length() % 2 == 0 ? "" : ".";
19
20              for ( int index = 1; index < aString.length() / 2 + 1; index ++ )       {
21                      pattern = "(.)" + pattern + "\\" + index;
22              }
23              pattern = anchored ? "^" + pattern + "$" : pattern;
24              testPattern(pattern, aString, "Special Test Palindrom which is " + aString
25          }
26          public static void testWithPattern()        {
27              String aString;
28              Pattern aPattern;
29              try {
30                      Scanner sc  = new Scanner( new File("input.txt") );
31                      while ( sc.hasNextLine() )      {
32                              // aString = sc.nextLine();
33                              aString = "..";
34
35                              // \'a' is part of the word.
36                              // testPattern(".*a.*", aString, "string has at least one
37                                      // Palindrome anchored at the beginning and end of
38                              specialTestPattern(false, aString, "Palindrome anchored at
39                              specialTestPattern(true, aString, "Palindrome anchored at
40                                      // Include a palindrome which is 2 characters long
41                              testPattern(".*(.)\\1.*", aString, "Palindrom which is 2 c
42                                      // Include a palindrome which is 3 characters long
43                              testPattern(".*(.).\\1.*", aString, "a palindrome which is
44                                      // The word consist only of 'a's or 'b's.
45                              testPattern("[ab]+", aString, "\'a's or 'b's build the wor
46                                      // \'a's or 'b's can not be part of the word.
47                              testPattern("[^ab]+", aString, "\'a's or 'b's can not be p
48                                      // The word is == '.'.
49                              testPattern("^\\.$", aString, "the word is '.'");
50                                      // The word includes a '.'.
51                              testPattern(".*\\..*", aString, "the word includes '.'");
52                      }
53                      sc.close();
54              } catch ( Exception e ) {
55                      e.printStackTrace();
56              }
57          }
58          public static void test1(String aString, String comment ) {
59              if ( aString.indexOf("a") >= 0 )        {
60                      System.out.println("Input line: " + aString);
```

```
61                      System.out.println("\t" + comment);
62              }
63          }
64          public static void test2(String aString, String comment ) {
65              int max = aString.length() / 2;
66              int index = 0;
67              while (    ( aString.charAt(index) == aString.charAt(aString.length() - in
68                      && ( index < max ) )
69                      index ++;
70              if ( index == max  )    {
71                      System.out.println("Input line: " + aString);
72                      System.out.println("\t" + comment);
73              }
74          }
75          public static void test3(String aString, String comment ) {
76              int max = aString.length();
77              int index = 0;
78              while (     ( index < max  - 1 )
79                      &&  ( aString.charAt(index) != aString.charAt(index + 1) ) )    {
80                      index ++;
81              }
82              if ( (index < max - 1 ) && aString.charAt(index) == aString.charAt(index +
83                      System.out.println("Input line: " + aString);
84                      System.out.println("\t" + comment);
85              }
86          }
87          public static void test4(String aString, String comment ) {
88              int max = aString.length();
89              if (  max < 3 )
90                      return;
91              int index = 0;
92              while (     ( index < max  - 2 )
93                      &&  ( aString.charAt(index) != aString.charAt(index + 2) ) )    {
94                      index ++;
95              }
96              if (       (index < max - 2 )
97                      && (aString.charAt(index) == aString.charAt(index + 2) ) )      {
98                      System.out.println("Input line: " + aString);
99                      System.out.println("\t" + comment);
00              }
01          }
02          public static void test5(String aString, String comment ) {
03              int max = aString.length();
04              int index = 0;
05              while (     ( index < max  )
06                      &&  ( ( aString.charAt(index) == 'a' ) ||
07                            ( aString.charAt(index) == 'b' )
08                          )
09                   )
10                      index ++;
11              if ( index  == max )    {
12                      System.out.println("Input line: " + aString);
13                      System.out.println("\t" + comment);
14              }
```

```
15
16          }
17      /*
18          public static boolean OwnCheckCannotHaveAB(String word) {
19              if(word.indexOf('a') == -1 && word.indexOf('b') == -1)
20                  return true;
21              else
22                  return false;
23          }
24      1. should be written as:
25          public static boolean OwnCheckCannotHaveAB(String word) {
26              return (word.indexOf('a') == -1 && word.indexOf('b') == -1)
27          }
28      2.  is false why?
29      */
30          public static void test6(String aString, String comment ) {
31               if (  ! (      ( aString.indexOf("a") >= 0 )
32                        ||     ( aString.indexOf("b") >= 0 )
33                        )
34                  )   {
35                      System.out.println("Input line: " + aString);
36                      System.out.println("\t" + comment);
37              }
38
39          }
40          public static void test7(String aString, String comment ) {
41               if (  aString.equals(".") ) {
42                      System.out.println("Input line: " + aString);
43                      System.out.println("\t" + comment);
44              }
45
46          }
47          public static void test8(String aString, String comment ) {
48               if (  aString.indexOf(".") >= 0 ) {
49                      System.out.println("Input line: " + aString);
50                      System.out.println("\t" + comment);
51              }
52
53          }
54          public static void testWithOwn()    {
55              String aString;
56              Pattern aPattern;
57              try {
58                      Scanner sc  = new Scanner( new File("input.txt") );
59                      while ( sc.hasNextLine() )      {
60                              aString = sc.nextLine();
61                              aString = ".a.";
62
63                                      // \'a' is part of the word.
64                              test1(aString, "string has at least one a ");
65                                      // Palindrome anchored at the beginning and end of
66                              test2(aString, "Palindrome anchored at the beginning and e
67                                      // Include a palindrome which is 2 characters long
68                              test3(aString, "Palindrom which is 2 characters long");
```

```
69                                      // Include a palindrome which is 3 characters long
70                                  test4(aString, "a palindrome which is 3 characters long.")
71                                      // The word consist only of 'a's or 'b's.
72                                  test5( aString, "\'a's or 'b's build the word");
73                                      // \'a's or 'b's can not be part of the word.
74                                  test6(aString, "\'a's or 'b's can not be part of the word"
75                                      // The word is == '.'.
76                                  test7(aString, "the word is '.'");
77                                      // The word includes a '.'.
78                                  test8(".*\\..*", aString, "the word includes '.'");
79                          }
80                          sc.close();
81                  } catch ( Exception e ) {
82                          e.printStackTrace();
83                  }
84          }
85          public static void main( String[] args ) {
86              testWithPattern();
87              testWithOwn();
88          }
89      }
```

Source Code: Src/23_sol/RegExTest.java

1.3. Homework 3.3 (10 Points)

Objective: Getting familiar with the Scanner class.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Implement a program which allows to play the game 'Let's see the art'.

Explanation:
This game is normally played as a two player game, but you have to implement a single person version of the game. The program will show the player a word, but each character of the word is show as '_'. The player has to guess one character at a time. If the player guesses a correct character of the word, all characters will be displayed. Every time the player guesses wrong, more of a piece of 'character art' will be chosen and shown. It is up to you how much of the art work will be shown, but the complete artwork has to be visible. The game is over after 9 guesses.

Your Work:
You program has to read a list of words from a file, from which a random word will be shown to the player. The program then has to show a representation of the word and read the guesses of the player. All correct guessed characters will be shown. The artowrk will niot be displayed for correct guesses.

For every incorrect gues more of the artowrk will be shown. The complete artwork is shown after 9 incorrect guesses, which also ends a round. You may end the game.

The artwork can be hard coded in your program.

It must be possible to specify the ’word.txt’ file on the command line.

```
% java ArtWork words.txt
```

Requirements:

You have to name your file ArtWork.java

Example:

An example of a solution execution:

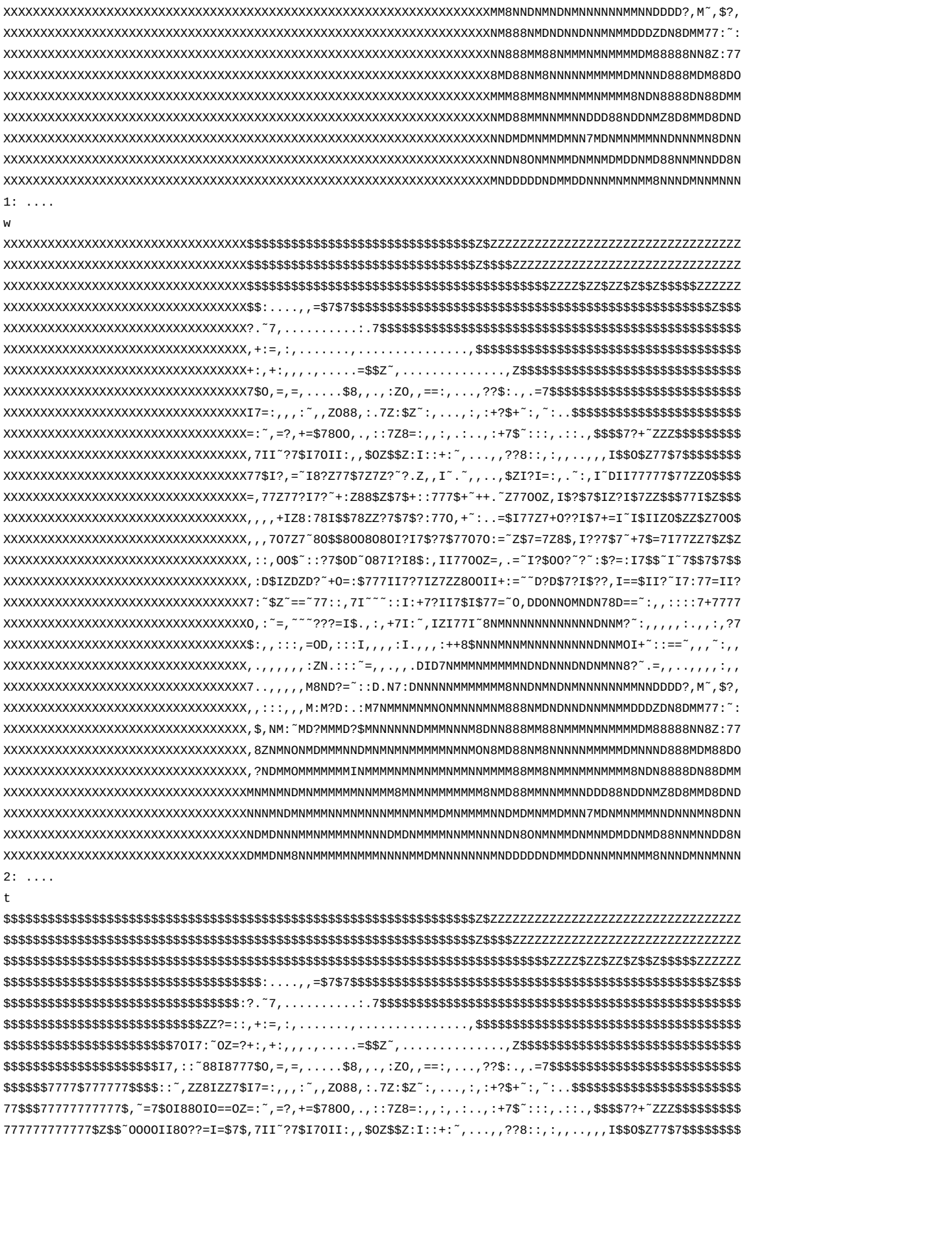
For these examples I have set the maximum number of wrong guesses to 3 and the word to guess is ’abca’.

All guesses are correct for the this execution.

```
java ArtWork words.txt
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
0: ....
b
0: .b..
a
0: ab.a
c
The word was: abca
```

All guesses are wrong for the this execution.

```
java ArtWork words.txt
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0: ....
t
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$ZZZZ$ZZ$ZZ$Z$$Z$$$$$ZZZZZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$Z$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX..,Z$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX?$:.,.=7$$$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX+?$+~:,~:..$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:+7$~:::,.::.,$$$$7?+~ZZZ$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,,??8::,:,,..,,,I$$O$Z77$7$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.,$ZI?I=:,.~:,I~DII77777$77ZZO$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.~Z77OOZ,I$?$7$IZ?I$7ZZ$$$77I$Z$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.=$I77Z7+O??I$7+=I~I$IIZO$ZZ$Z7OO$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:=~Z$7=7Z8$,I??7$7~+7$=7I77ZZ7$Z$Z
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX=,.=~I?$OO?~?~:$?=:I7$$~I~7$$7$7$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXI+:=~~D?D$7?I$??,I==$II?~I7:77=II?
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX7=~O,DDONNOMNDN78D==~:,,::::7+7777
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNMNNNNNNNNNNNNNDNNM?~:,,,,,:.,,:,?7
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNMNNMNNNNNNNNNDNNMOI+~::==~,,,~:,,
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXMMMMNDNDNNNDNDNMNN8?~.=,,..,,,,:,,
```


XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXMM8NNDNMNDNMNNNNNNMMNNDDDD?,M~,$?,
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNM888NMDNDNNDNNMNMMDDDZDN8DMM77:~:
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNN888MM88NMMMNMNMMMMDM88888NN8Z:77
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX8MD88NM8NNNNNMMMMMDMNNND888MDM88DO
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXMMM88MM8NMMNMMNMMMM8NDN8888DN88DMM
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNMD88MMNNMMNNDDD88NDDNMZ8D8MMD8DND
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNNDMDMNMMDMNN7MDNMNMMMNNDNNNMN8DNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNNDN8ONMNMMDNMNMDMDDNMD88NNMNNDD8N
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXMNDDDDDNDMMDDNNNMNMNMM8NNNDMNNMNNN
1: ....
w
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$Z$ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$Z$$$$ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ZZZZZZZZ$Z$$Z$$$$$ZZZZZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$$:....,,=$7$7$$Z$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX?.~7,.........:.7$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,+:=,:,.......,.............,$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX+:,+:,,,.,.....=$$Z~,.............,Z$$$$$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX7$O,=,=,.....$8,,.,:ZO,,==:,...,??$:.,.=7$$$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXI7=:,,,:~,,ZO88,:.7Z:$Z~:,...,:,:+?$+~:,~:..$$$$$$$$$$$$$$$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX=:~,=?,+=$78OO,.,::7Z8=:,,:,.:..,:+7$~:::,.::.,$$$$7?+~ZZZ$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,7II~?7$I7OII:,,$OZ$$Z:I::+:~,...,,??8::,:,,..,,,I$$O$Z77$7$$$$$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX77$I?,=~I8?Z77$7Z7Z?~?.Z,,I~.~,,..,$ZI?I=:,.~:,I~DII77777$77ZZO$$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX=,77Z77?I7?~+:Z88Z7$+::777$+~++.~Z77OOZ,I$?$7$IZ?I$7ZZ$$$77I$Z$$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,,,,+IZ8:78I$$78ZZ?7$7$?:77O,+~:..=$I77Z7+O??I$7+=I~I$IIZO$ZZ$Z7OO$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,,,7O7Z7~8O$$8OO8O8OI?I7$?7$77O7O:=~Z$7=7Z8$,I??7$7~+7$=7I77ZZ7ZZ
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,::,OO$~::?7$OD~O87I?I8$:,II77OOZ=,.=~I?$OO?~?~:$?=:I7$$~I~7$$7$7$$
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,:D$IZDZD?~+O=:$777II7?7IZ7ZZ8OOII+:=~~D?D$7?I$??,I==$II?~I7:77=II?
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX7:~$Z~==~77::,7I~~~::I:+7?II7I77=~O,DDONNOMNDN78D==~:,,::::7+7777
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXO,:~=,~~~???=I$.,:,+7I:~,IZI77I~8NMNNNNNNNNNNNNDNNM?~:,,,,,:.,,:,?7
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX$:,,:::,=OD,:::I,,,,:I.,,,:++8$NNNMNNMNNNNNNNNNDNNMOI+~::==~,,,~:,,
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,.,,,,,,:ZN.:::~=,,.,,.DID7NMMMNMMMMMNDNDNNNDNDNMNN8?~.=,,.,,,,,:,,
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX7..,,,,,M8ND?=~::D.N7:DNNNNNMMMMMMM8NNDNMNDNMNNNNNNMMNNDDDD?,M~,$?,
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,,:::,,,M:M?D:.:M7NMMNMNMONMNNNMNM888NMDNDNNDNNMNMMDDDZDN8DMM77:~:
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,$,NM:~MD?MMMD?$MNNNNNNDMMMNNNM8DNN888MM88NMMMNMNMMMDM88888NN8Z:77
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,8ZNMNONMDMMMNNDMNMNMNMMMMMNMNMON8MD88NM8NNNNNMMMMDMNNND888MDM88DO
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX,?NDMMOMMMMMMMINMMMMNMNMNMMNMNMNMMM88MM8NMMNMMNMMMM8NDN8888DN88DMM
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXMNMNMNDMNMMMMMNNMMM8MNMNMMMMMMM8NMD88MMNNMMNNDDD88NDDNMZ8D8MMD8DND
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNNNMNDMNMMMNNMNMNNNMMNMNMMDMNMMMNNDMDMNMMDMNN7MDNMNMMMNNDNNNMN8DNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXNDMDNNNMMNMMMMNMNNNDMDNMMMMNNMMNNNNDN8ONMNMMDNMNMDMDDNMD88NNMNNDD8N
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXDMMDNM8NNMMMMMNMMMNNNNMMDMNNNNNNNMNDDDDDNDMMDDNNNMNMNMM8NNNDMNNMNNN
2: ....
t
$$$Z$ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
$$Z$$$$ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
$$ZZZZZZZZ$Z$$Z$$$$$ZZZZZZ
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:....,,=$7$7$$Z$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:?.~7,.........:.7$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ZZ?=::,+:=,:,.......,.............,$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$7OI7:~OZ=?+:,+:,,,.,.....=$$Z~,.............,Z$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$I7,::~88I8777$O,=,=,.....$8,,.,:ZO,,==:,...,??$:.,.=7$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$7777$777777$$$$::~,ZZ8IZZ7$I7=:,,,:~,,ZO88,:.7Z:$Z~:,...,:,:+?$+~:,~:..$$$$$$$$$$$$$$$$$$$$$$
77$$$77777777777$,~=7$OI88OIO==OZ=:~,=?,+=$78OO,.,::7Z8=:,,:,.:..,:+7$~:::,.::.,$$$$7?+~ZZZ$$$$$$$$
777777777777$Z$$~OOOOII8O??=I=7,7II~?7$I7OII:,,$OZ$$Z:I::+:~,...,,??8::,:,,..,,,I$$O$Z77$7$$$$$$$$

```
777777777OZZO7,$O7ZO77=$++~:$IZI777$I?,=~I8?Z77$7Z7Z?~?.Z,,I~.~,,..,$ZI?I=:,.~:,I~DII77777$77ZZO$$$$
7I7777$OZ$:+$O7$$8I$7ZO?::~$I77~:=,77Z77?I7?~+:Z88$Z$7$+::777$+~++.~Z77OOZ,I$?$7$IZ?I$7ZZ$$$77I$Z$$$
?I~?7?Z~:$~~$$:$7I7$I=~~~?$Z?:,::,,,,+IZ8:78I$$78ZZ?7$7$?:77O,+~:..=$I77Z7+O??I$7+=I~I$IIZO$ZZ$Z7OO$
I+7OOZ:.=,:~$87?ZZ~:::$7,$$Z,:,,:,,,7O7Z7~8O$$8OO8O8OI?I7$?7$77O7O:=~Z$7=7Z8$,I??7$7~+7$=7I77ZZ7$Z$Z
Z$Z7ZZ$0$Z7II7,,,,::,,:Z$$$7:::,:,::,OO$~::?7$OD~O87I?I8$:,II77OOZ=,.=~I?$OO?~?~:$?=:I7$$~I~7$$7$7$$
$$I$$$:=I7~:::,:,,I.,,.7$I,,,=++,,:D$IZDZD?~+O=:$777II7??IZ7ZZ8OOII+:=~~D?D$7?I$??,I==$II?~I7:77=II?
Z$$7I:I7I=~:::,,....,$$$7.,$7$$$O7:~$Z~==~77::,7I~~~::I:+7?II7$I$77=~O,DDONNOMNDN78D==~:,,::::7+7777
7777777~:::,::,,:,=?77~.,,78$O7IZO,:~=,~~~???=I$.,:,+7I:~,IZI77I~8NMNNNNNNNNNNNNDNNM?~:,,,,,:.,,:,?7
I.+=77+,:::,?=:,:==,,.,Z.~77777$7$:,,:::,=OD,:::I,,,,:I.,,,:++8$NNNMNNMNNNNNNNNNDNNMOI+~::==~,,,~:,,
?===~,$7::,,.+:,,.,+Z7$IIII$~,?I$,.,,,,,,:ZN.:::~=,,.,,.DID7NMMMNMMMMMNDNDNNNDNDNMNN8?~.=,,..,,,,:,,
N~=,:,,,I?7:..~+$$77,.?I+?.,7I,,77..,,,,,M8ND?=~::D.N7:DNNNNNMMMMMMM8NNDNMNDNMNNNNNNMMNNDDDD?,M~,$?,
8Z=..$7I?~Z7,,...~7$7D:,,,:~:::,,,,:::,,,M:M?D:.:M7NMMNMNMNONMNNNMNM888NMDNDNNDNNMNMMDDDZDN8DMM77:~:
DDN:,D,.=7=.........77$$.,$,,,,,,,$,NM:~MD?MMMD?$MNNNNNNDMMMNNNM8DNN888MM88NMMMNMNMMMDM88888NN8Z:77
MMMD7D7$777I:.......IOOOZOM?N7,+,,8ZNMNONMDMMMNNDMNMNMNMMMMMNMNMON8MD88NM8NNNNNMMMMDMNNND888MDM88DO
MMN?~N$?O+.......:$$88D77$M$ND$8~,?NDMMOMMMMMMMINMMMNMNMNMNMMNMNMNMMM88MM8NMMNMMNMMM8NDN8888DN88DMM
MMN:MNN$M$OZ$.==::8$DDMN$NNNDN~NMMNMNMNDMNMMMMMNMNMM8MNMNMMMMMM8NMD88MMNNMMNNDDD88NDDNMZ8D8MMD8DND
NMMMMMNNN$MZNZ$Z88:,?:NMMNNNDMMNMNNNMNDMNMMMNNMNMNNNMMNMNMMDMNMMMNNDMDMNMMDMN7MDNMNMMMNNDNNNMN8DNN
NNNNMNNMNDNNNN8DNDMN7NMNNMMMNNNMMNDMDNNNMMNMMMMNMNNNDMDNMMMMNNMMNNNNDN8ONMNMMDNMNMDMDDNMD88NNMNNDD8N
DMNDMMMDNNNDMNNMDNDNNMDNMNMMMMMMNDMMDNM8NNMMMMMNMMMNNNNMMDMNNNNNNNMNDDDDDNDMMDDNNNMNMNMM8NNNDMNNMNNN
The word was: abca
Do you want to continue (yes/no)?
```

Submission:

```
% ssh glados.cs.rit.edu # or use queeg.cs.rit.edu if glados is down
# password
# go to the directory where your solution is ...
% try hpb-grd lab3-3 'All files required'
# you can see if your submission was successful:
# try -q hpb-grd lab3-3
```

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      /**
 2       * This class implements a ArtWork Game.
 3       */
 4
 5      import java.io.File;
 6      import java.util.Scanner;
 7      import java.util.Random;
 8
 9
10      public class ArtWork {
11              private String fileName         = "";;
12              private Scanner theWords        = null;;
13              private Scanner theInput        = new Scanner(System.in);
14              private String theWordToGuess;
15              private String theWordShownToTheUser;
```

```
16              private int soManyGuesses       = 0;
17              final private Random aRandom    = new Random(System.currentTimeMillis());
18              final private int MAX_GUESSES   = 3;    // only numbers from 1 - 9 can be
19              final private char HIDE         = 'X';
20              private String[] pickAWordFromHere ;
21              boolean endTheGame = false;
22              final String theScene[] =      {
23                              "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
24                              "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
25                              "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
26                              "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:....,,=$7$7$$$$$$$$$$
27                              "$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:?.~7,.........:.7$$$$$$
28                              "$$$$$$$$$$$$$$$$$$$$$$$$$$$$ZZ?=::,+:=,:,.......,.........
29                              "$$$$$$$$$$$$$$$$$$$$$$$7OI7:~OZ=?+:,+:,,,.,.....=$$Z~,...
30                              "$$$$$$$$$$$$$$$$$$$$$I7,::~88I8777$O,=,=,.....$8,,.,:ZO,,
31                              "$$$$$7777$777777$$$$::~,ZZ8IZZ7$I7=:,,,:~,,ZO88,:.7Z:$Z~
32                              "77$$$77777777777$,~=7$OI88OIO==OZ=:~,=?,+=$78OO,.,::7Z8=:
33                              "777777777777$Z$$~OOOOII8O??=I=$7$,7II~?7$I7OII:,,$OZ$$Z:I
34                              "77777777OZZO7,$O7ZO77=$++~:$IZI777$I?,=~I8?Z77$7Z7Z?~?.Z
35                              "7I7777$OZ$:+$O7$$8I$7ZO?::~$I77~:=,77Z77?I7?~+:Z88$Z$7$+:
36                              "?I~?7?Z~:$~~$$:$7I7$I=~~~?$Z?:,::,,,,+IZ8:78I$$78ZZ?7$7$?
37                              "I+7OOZ:.=,:~$87?ZZ~:::$7,$$Z,:,,:,,,7O7Z7~8O$$8OO8O8OI?I7
38                              "Z$Z7ZZ$O$Z7II7,,,,::,,:Z$$$7:::,:,::,OO$~::?7$OD~O87I?I8$
39                              "$$I$$$:=I7~:::,:,,I.,,.7$I,,,=++,,:D$IZDZD?~+O=:$777II7?7
40                              "Z$$7I:I7I=~:::,,....,$$$7.,$7$$$O7:~$Z~==~77::,7I~~~::I:+
41                              "7777777~:::,::,,:,=?77~.,,78$O7IZO,:~=,~~~???=I$.,:,+7I:~
42                              "I.+=77+,:::,?=:,:==,,.,Z.~77777$7$:,,:::,=OD,:::I,,,,:I.,
43                              "?===~,$7::,,.+:,,.,+Z7$IIII$~,?I$,.,,,,,,:ZN.:::~=,,.,,.D
44                              "N~=,:,,,I?7:..~+$$77,.?I+?.,7I,,77..,,,,,M8ND?=~::D.N7:DN
45                              "8Z=..$7I?~Z7,,...~7$7D:,,,:~:::,,,,:::,,,M:M?D:.:M7NMMNMN
46                              "DDN:,D,.=7=.........77$$.,$,,,,,,,$,NM:~MD?MMMD?$MNNNNNND
47                              "MMMD7D7$777I:.......IOOOZOM?N7,+,,8ZNMNONMDMMMNNDMNMNMNMM
48                              "MMN?~N$?O+.......:$$88D77$M$ND$8~,?NDMMOMMMMMMMINMMMMNMNM
49                              "MMN:MNN$M$OZ$.==::8$DDMN$NNNDN~NMMNMNMNDMNMMMMMMNNMMM8MNM
50                              "NMMMMMNNN$MZNZ$Z88:,?:NMMNNNDMMNMNNNMNDMNMMMNNMNMNNNMMNMN
51                              "NNNNMNNMNDNNNN8DNDMN7NMNNMMMNNNMMNDMDNNNMMNMMMMNMNNNDMDNM
52                              "DMNDMMMDNNNDMNNMDNDNNMDNMNMMMMMNDMMDNM8NNMMMMNMMMNNNNMM
53                                              };
54
55              private void printTheWords()    {       // for testing only
56                      for ( int index = 0; index < pickAWordFromHere.length; index ++ )
57                              System.out.println(index  + "/" + pickAWordFromHere.length
58                      }
59              }
60              private void fillTheWords(Scanner sc)   {
61                      try {
62                              String theWords = "";
63                              while ( sc.hasNext() )          // no word at postion 0;
64                                      theWords = theWords +":" + sc.next();
65                              sc.close();
66                              pickAWordFromHere = theWords.split(":");
67                      } catch ( Exception e ) {}      // nothing checked if file does no
68              }
69              private void parseArgs (String[] args)  {
```

```
70                      try {
71                              if ( args.length == 0 ) {
72                                      theWords = new Scanner(new File( "words.txt") );
73                              } else {
74                                      theWords = new Scanner(new File( args[0]) );
75                              }
76                              fillTheWords(theWords);
77                              if ( theWords == null )
78                                      throw new Exception("scanner is not initialized");
79                      } catch (Exception e )  {
80                              System.err.println("Arguments could not be parsed.");
81                              e.printStackTrace();
82                              System.exit(1);
83                      }
84              }
85              private int createRandomNumber(int last )       {
86                      int  nextRandom = aRandom.nextInt();
87                      return Math.abs( nextRandom % last );
88              }
89              private String extractWord()    {
90                      int index = createRandomNumber(pickAWordFromHere.length);
91                      index = index == 0 ?  1 : index;        // no image at position 0
92
93                      String randomWord =  pickAWordFromHere[index];
94                      theWordShownToTheUser = randomWord.replaceAll(".", ".");
95                      return randomWord;
96              }
97              private boolean guessedCorrectly(String theGuess)       {
98                      boolean correctGuess  = ( theWordToGuess.indexOf(theGuess) >= 0 );
99                      if ( correctGuess )     {
00                              char[] theWordAsArray = theWordShownToTheUser.toCharArray(
01                              for ( int index = 0; index < theWordToGuess.length(); inde
02                                      if ( theWordToGuess.charAt(index) == theGuess.char
03                                              theWordAsArray[index] = theGuess.charAt(0)
04                              }
05                              theWordShownToTheUser = new String(theWordAsArray);
06                      }
07
08                      return  correctGuess;
09              }
10              private boolean notDone()       {
11                      return   ( (soManyGuesses >= MAX_GUESSES ) || ( theWordShownToTheU
12              }
13              private void printScene()       {
14
15                      for ( int line = 0; line < theScene.length; line ++ )   {
16                              String toPrint = theScene[line];
17                              double multiplier = ( (double) toPrint.length()/(double)MA
18                              int toHide = (int)( multiplier  * (MAX_GUESSES - soManyGue
19                              if ( MAX_GUESSES == soManyGuesses )
20                                      toHide = 0;
21                              String prefix = "";
22                              char[] theStringAsArray = toPrint.toCharArray();
23                              for ( int index = 0; index < toHide; index ++  )        {
```

```
24                                      theStringAsArray[index] = HIDE;
25                              }
26                              toPrint = new String(theStringAsArray);
27
28                              System.out.println(toPrint);
29                      }
30              }
31              private void printWord()        {
32                      System.out.println(soManyGuesses + ": " + theWordShownToTheUser );
33              }
34              private void playOneGame() {
35                      String guess = null;
36                      soManyGuesses = 0;
37                      printScene();
38                      do {
39                              printWord();
40                              if ( theInput.hasNext() )
41                                      guess = theInput.next();
42                              else {
43                                      System.err.println("Can not read input. Bye ");
44                                      System.exit(2);
45                              }
46                              if ( ! guessedCorrectly(guess) )        {
47                                      soManyGuesses ++;
48                                      printScene();
49                              }
50                      } while ( ! notDone() );
51
52                      System.out.println("The word was: " + theWordToGuess);
53              }
54              private void startTheGame(String[] args)        {
55                      parseArgs(args);
56                      do      {
57                              theWordToGuess = extractWord();
58                              playOneGame();
59                              System.out.println("Do you want to continue (yes/no)?");
60                              if ( theInput.hasNext() )
61                                      endTheGame = (theInput.next()).equals("yes");
62                      } while ( endTheGame );
63
64              }
65
66              public static void main( String[] args ) {
67                      new ArtWork().startTheGame(args);
68              }
69
70
71      }
```

Source Code: Src/23_sol/ArtWork.java