**Name**

**Advanced Programming in Java**

**Lab Exercise 10/17/2024**

**StringTokenizer Class**

Refer to Lesson 23 in Blue Pelican Java

1. Create a *StringTokenizer* object called *st*. We wish to tokenize *String zulu* and specify

only a plus sign as a delimiter.

2. What are the “things” called that separate the “words” within a *String* that is to be

tokenized?

3. What are the individual parts or “words” called in a *String* that is to be tokenized?

4. What is the import we need in order to get the *StringTokenizer* to work?

5. What is the output of the following code?

StringTokenizer t = new StringTokenizer(“Hello there good buddy”);

String m = t.nextToken( );

System.out.println(m + “>>>” + t.countTokens( ) + “ tokens left.”);

6. Rewrite the following *if* statement using *countTokens( )* rather than *hasMoreTokens( )*.

if ( jj.hasMoreTokens( ) )

{ …. }

7. What is the output of the following code?

StringTokenizer g = new StringTokenizer(“Rumplestillskin”, “me”);

System.out.println( g.nextToken( ) );

System.out.println( g.nextToken(“s”) );

8. Write a class called *SpecialToken* that has a *static* method called *thirdToken*. This method should return as a *String*, the third token of a *String* that you pass as a parameter. You may assume that spaces will serve as delimiters.

9. Which constructor for the *StringTokenizer* class would be simplest to use if you wanted spaces and tabs as delimiters?

In problems 10 – 13 state what’s printed. Use the following code and assume for each question that the code in the previous questions has been executed.

StringTokenizer gt = new StringTokenizer(“Humpty Dumpty”, “ pu\n\t”);

10. System.out.println(gt.countTokens( ));

11. String radString = gt.nextToken( );

System.out.println( gt.nextToken( ) + radString );

12. System.out.println(gt.countTokens);

13. What should replace ??? below in order to insure that we don’t get an exception?

while ( ??? )

{

System.out.println( gt.nextToken( ) );

}

14. What is output by the following code?

StringTokenizer tux = new StringTokenizer(“Ignoramus”);

System.out.println( tux.countTokens( ) );

System.out.println( tux.nextToken( ) );

System.out.println( tux.nextToken( ) );

**Project… Military Censor**

You are in the Army and have been assigned the task of censoring soldiers’ outgoing mail for security reasons. Let’s assume that the troops all know about an upcoming offensive that will involve an assault on the Hermes bridge that crosses the Muddy River. Develop an algorithm that uses the *StringTokenizer* to examine each word of outgoing email. If any of the following words are found, print the word REJECTED. If none are found, then print OK. Taboo words are: Hermes, bridge, Muddy, River, assault, and offensive.

Call your class *Censor* and use the following sentences for testing.

“I hope I survive the assault tomorrow.”

“I want to talk to you about Bobby, but we’ll cross that bridge later.”

“Tell sis and Larry that I’ll be Ok and I will see them in 6 months”

“Your last letter was a little muddy on exactly what you meant.”

“I see no point in us trying to take the hermes crossing.”

Notice the last sentence uses “hermes” instead of “Hermes”. Your code should not be sensitive to case and should reject this sentence.

You should input these sentences via the keyboard using the *Scanner* class. Your output screen should look like the following after testing all the sentences:

Enter next sentence: I hope I survive the assault tomorrow.

I hope I survive the assault tomorrow.>>>REJECTED

Enter next sentence: I want to ask about Bobby, but we'll cross that

bridge later.

I want to ask about Bobby, but we'll cross that bridge

later.>>>REJECTED

Enter next sentence: Tell sis and Larry that I'll be ok and I will see

them in 6 months.

Tell sis and Larry that I'll be ok and I will see them in 6

months.>>>OK

Enter next sentence: Your last letter was a little muddy on exactly

what you meant.

Your last letter was a little muddy on exactly what you

meant.>>>REJECTED

Enter next sentence: I see no point in us trying to take the hermes

crossing.

I see no point in us trying to take the hermes crossing.>>>REJECTED

**Nuclear Energy Application**

Einstein’s famous equation  gives the amount of energy released by the complete conversion of matter of mass *m* into energy *e*. If m represents the mass in kilograms and c represents the speed of light in meters per second (3 \* 108 m/s), then the result is energy units joules. It takes 360000 joules to light a 100-watt light bulb for an hour. Create an *Energy* application that prompts the user for a mass in kilograms and then displays the energy and the number of light bulbs that could be powered for an hour. Output should look like:

Enter the mass in kilograms: 1

The energy produced in joules is: 9.0e16

The number of 100-watt light bulb powered for an hour: 2.5e11