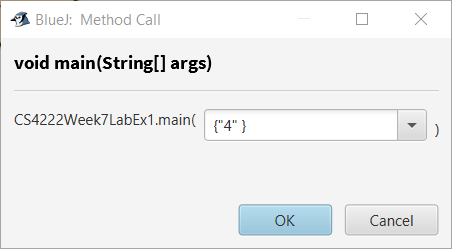
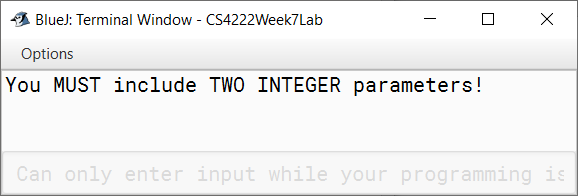
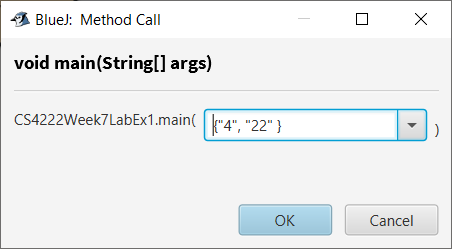
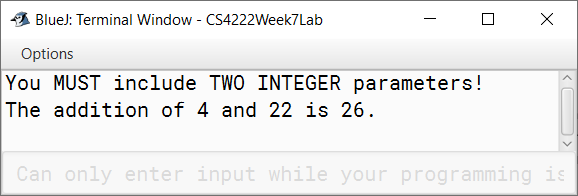
1. Write the code for a main method that is passed two integer parameters and outputs the sum/addition of the two values in the terminal window. If more, or less than two parameter values are passed the program should output the message “You MUST include TWO INTEGER parameters!” should be displayed in the terminal window.

For example

**// Select this line**

public class CS4222Week7LabEx1 {

public static void main(String[] args) {

if(args.length == 2) {

int x = Integer.parseInt(args[0]);

int y = Integer.parseInt(args[1]);

System.out.printf("The addition of %d and %d is %d.%n",x,y,x+y);

} else {

System.out.println("You MUST include TWO INTEGER parameters!");

}

}

}

**// and this one to unhide a sample solution**

1. Write the code for a main method that is passed a single string as a parameter and checks if the string is a palindrome. If no parameter is included, or if more than one parameter is included, the program should output the message

Invalid command usage. Use...

java CommandLinePalindrome text”

For example

java CommandLinePalindrome kayak

'kayak' is a palindrome

java CommandLinePalindrome racecar

'racecar' is a palindrome

**// Select this line**

public class CommandLinePalindromeV0 {

// Is the contents of a string a palindrome?

public static void main(String[] args) {

if(args.length != 1) {

System.out.println("Invalid command usage. Use...\n java CommandLinePalindrome text");

} else {

String text = args[0]; // Kayak Radar Racecar

int left = 0;

int right = text.length()-1;

while(left < right && text.charAt(left) == text.charAt(right)) {

left++;

right--;

}

if(left < right) {

System.out.printf("'%s' is NOT a palindrome\n",text);

} else {

System.out.printf("'%s' is a palindrome\n",text);

}

}

}

}**// and this one to unhide a sample solution**

}

1. Modify the code in Q2 so that the command can have a minimum of one word passed as parameter but any number of words can be passed and each one is checked.

**// Select this line**

public class CommandLinePalindromeV1 {

public static void main(String[] args) {

if(args.length == 0) {

System.out.println("Invalid command usage. Use...\n java CommandLinePalindrome text [...]");

} else {

for(String text : args) { // Kayak Radar Racecar

int left = 0;

int right = text.length()-1;

while(left < right && text.charAt(left) == text.charAt(right)) {

left++;

right--;

}

if(left < right) {

System.out.printf("'%s' is NOT a palindrome\n",text);

} else {

System.out.printf("'%s' is a palindrome\n",text);

}

}

}

}

}

**// and this one to unhide a sample solution**

1. Write the code for a main method that is passed a single string as a parameter and outputs the Phonetic Alphabet equivalent of the string. If no parameter is included, or if more than one parameter is included, the program should output the message

Invalid command usage. Use...

java CommandLinePhoneticAlphabet text”

For example

java CommandLinePhoneticAlphabet MUNSTER

Mike Uniform November Sierra Tango Echo Romeo

java CommandLinePhoneticAlphabet SW1 Y48G

Sierra Whiskey One Yankee Four Eight Golf

Remember, the Phonetic Alphabet uses the following letter to word mappings

| **Letter** | **phonetic letter** | **Letter** | **phonetic letter** | **Letter** | **phonetic letter** |
| --- | --- | --- | --- | --- | --- |
| A | Alpha | J | Juliet | S | Sierra |
| B | Bravo | K | Kilo | T | Tango |
| C | Charlie | L | Lima | U | Uniform |
| D | Delta | M | Mike | V | Victor |
| E | Echo | N | November | W | Whiskey |
| F | Foxtrot | O | Oscar | X | X-ray |
| G | Golf | P | Papa | Y | Yankee |
| H | Hotel | Q | Quebec | Z | Zulu |
| I | India | R | Romeo |  |  |

**// Select this line**

public class CommandLinePhoneticText {

// Is a word a palindrome

public static void main(String[] args) {

if(args.length != 1) {

System.out.println("Invalid command usage. Use...\n java CommandLinePhoneticText text");

} else {

String text = args[0];

int alphaPos = 0;

String alphabet = "abcdefghijklmnopqrstuvwxyz0123456789";

String[] phoneticAlphabet = {

"Alpha", "Bravo", "Charlie", "Delta", "Echo", "Foxtort", "Golf",

"Hotel", "India", "Juliet", "Kilo", "Lima", "Mike", "November",

"Oscar", "Papa", "Quebec", "Romeo", "Sierra", "Tango",

"Uniform", "Victor", " Whiskey ", "X-ray", "Yankee", "Zulu",

"Zero", "One", "Two", "Three", "Four", "Five", "Six", "Seven",

"Eight", "Nine"};

for(int i = 0; i < text.length(); i++) {

alphaPos = alphabet.indexOf(Character.toLowerCase(text.charAt(i)));

if(alphaPos == -1) {

// Not in the alphabet so just print the character as it is

System.out.print(text.charAt(i));

} else {

// In the alphabet so print the Phonetic Equivalent of the character

System.out.print(phoneticAlphabet[alphaPos]);

System.out.print(" ");

}

}

System.out.println();

}

}

} **// and this one to unhide a sample solution**

1. Write the code for a main method that uses the following command line format

java CommandLinePhoneticAlphabet cardNumber [segmentSize]

The program is passed a card number as a string and an optional segment size for formatting the display. The card number should be displayed in segments of the required size. If no segment size is included in the command line the segment size should default to a size of 4.

For example

java CommandLineFormatCardNumber 1234567890123456

1234 5678 9012 3456

java CommandLinePhoneticAlphabet 1234567890123456 3

123 456 789 012 345 6

java CommandLinePhoneticAlphabet 21786304 2

21 78 63 04

java CommandLinePhoneticAlphabet 21786304 1

2 1 7 8 6 3 0 4

**// Select this line**

public class CommandLineFormatCardNumber {

public static void main(String[] args) {

int paramCount = args.length;

if(paramCount == 0 || paramCount > 2) {

System.out.println("Invalid command usage. Use...\n java CommandLineFormatCardNumber number [segmentSize]");

} else {

String number = args[0];

int segmentSize = 4; // default setting;

if(paramCount == 2) {

segmentSize = Integer.parseInt(args[1]);

}

// From assignment last semester!!!!

int startPos = 0;

while(startPos < number.length()) {

int endPos = Math.min(startPos + segmentSize,number.length());

System.out.print(number.substring(startPos, endPos) + " ");

startPos = startPos + segmentSize;

}

System.out.println();

}

}

}

**// and this one to unhide a sample solution**

1. Write the code for a main method that uses the following command line format

java CommandLineCountLinesInFile filepath

The program is passed a filepath string. If the file exists the program should count the number of lines in the files and display it on the screen.

**// Select this line**

import java.io.\*;

import java.util.\*;

/\*

\* This example lists the files in a specified directory. The command line format is a follows

\*

\* java CommandLineCountLinesInFile directory-path

\*

\* The program reads each LINE in the file and displays it on the screen.

\*/

public class CommandLineCountLinesInFile {

public static void main(String[] args) {

if(args.length == 1) {

// Use the parameter value as the filename

String fileName = args[0];

// Create a file handle to get access to the file

File fileHandle = new File(fileName);

// Check if the file handle creation was successful

if (fileHandle != null) {

int lineCount = 0;

try {

// Use the fileHandle to associate file with Scanner

Scanner dataSource = new Scanner(fileHandle);

// Create a String variable to 'receive' each input from the file

String lineFromFile = "";

// Check that ther file has more data to be processed

while(dataSource.hasNext()) {

// If there is more data then read the next line from the file

lineFromFile = dataSource.nextLine();

// Count the line

lineCount++;

}

System.out.printf(“’%s’ contains %d lines\n”,fileName, lineCount);

} catch(IOException e) {

System.out.printf("%s caused an error!",fileName);

}

} else {

System.out.printf("%s is not a valid file!",fileName);

}

} else {

System.out.println("Invalid command usage. Use...\n java DisplayFileContents filepath\n");

}

}

}

**// and this one to unhide a sample solution**