Christopher Simon

Monday, 10/19

|  |  |  |
| --- | --- | --- |
| Time | Activity | Description/Details |
| 3:00 – 3:05 | Attendance and Announcements |  |
| 3:00 – 3:50 | Working on Assignment 1 | Today, as planned last week, I dedicated all of this week to helping students on their first assignment. I’m planning on going over some of the core methods, such as the validFormat() and tokenizer methods.  Helping the students today didn’t go as well as planned. The instructions for the assignment weren’t as clear as I hoped, and so there was some disagreement on how to approach writing the code. Since there are so many ways of going about it, I had a difficult time getting my class to understand the way that I decided to tackle the problem.  I think that for the next assignment, in order to prevent this from happening, I plan to leave the work of writing the code methods to the students, and will instead go over how to put these methods, that differ from student to student, together. |
|  |  |  |
|  |  |  |

1 class Token {  
 2 public static void main(String[] args) {  
 3 String s = "1 + 2;";  
 4 System.out.println(isValidFormat(s));  
 5 }  
 6   
 7 public static boolean isValidFormat(String exp) {  
 8 String operand1 = token(exp);  
 9 exp = removeToken(exp);  
 10 exp = removeToken(exp);  
 11 String operation = token(exp);  
 12 exp = removeToken(exp);  
 13 exp = removeToken(exp);  
 14 String operand2 = token(exp);  
 15 exp = removeToken(exp);  
 16 String semicolon = token(exp);  
 17   
 18 if(validNumber(operand1) && validNumber(operand2) && validOperation(operation) && semicolon.equals(";")) {  
 19 return true;  
 20 } else {  
 21 return false;  
 22 }  
 23 }  
 24   
 25 public static boolean validNumber(String operand) {  
 26 boolean isValid;  
 27 switch(operand) {  
 28 case "0":   
 29 isValid = true;  
 30 break;  
 31 case "1":   
 32 isValid = true;  
 33 break;  
 34 case "2":   
 35 isValid = true;  
 36 break;  
 37 case "3":   
 38 isValid = true;  
 39 break;  
 40 case "4":   
 41 isValid = true;  
 42 break;  
 43 case "5":   
 44 isValid = true;  
 45 break;  
 46 case "6":   
 47 isValid = true;  
 48 break;  
 49 case "7":   
 50 isValid = true;  
 51 break;  
 52 case "8":   
 53 isValid = true;  
 54 break;  
 55 case "9":   
 56 isValid = true;  
 57 break;  
 58 default:  
 59 isValid = false;  
 60 break;  
 61 }  
 62 return isValid;  
 63 }  
 64   
 65 public static boolean validOperation(String operation) {  
 66 // Keep the status of the validity in a variable  
 67 boolean isValid;  
 68   
 69 // Is operation one of the following symbols? (+ - \* / ^ %)  
 70 switch(operation) {  
 71 case "+":  
 72 isValid = true;  
 73 break;  
 74 case "-":  
 75 isValid = true;  
 76 break;  
 77 case "\*":  
 78 isValid = true;  
 79 break;  
 80 case "/":  
 81 isValid = true;  
 82 break;  
 83 case "^":  
 84 isValid = true;  
 85 break;  
 86 case "%":  
 87 isValid = true;  
 88 break;  
 89 default:  
 90 // If the symbol is any other character than (+ - \* / ^ %)  
 91 isValid = false;  
 92 break;  
 93 }  
 94 // Return the boolean, isValid, which will be true  
 95 // if the operand is valid, and false if it isn't valid  
 96 return isValid;  
 97 }  
 98   
 99 public static String token(String s) {  
100 // Get the first character from the string, s  
101 String firstCharacter = "" + s.charAt(0);  
102 // Return that first character  
103 return firstCharacter;  
104 }  
105   
106 public static String removeToken(String s) {  
107 // Give us the substring of s from indexes 1 through 5.  
108 s = s.substring(1, s.length()); //\_+\_5;  
109 return s;  
110 }  
111 }

1 public class Swap {  
 2 public static void main(String... args) {  
 3 String s = "hello world";  
 4 System.out.println(swap(s));  
 5 }  
 6   
 7 public static String swap(String s) {  
 8 String flippedString = "";  
 9 int size = s.length() / 2 \* 2;  
10 for(int i = 0; i < size; i += 2) {  
11 char c1 = s.charAt(i);  
12 char c2 = s.charAt(i+1);  
13 flippedString += "" + c2 + c1;  
14 }  
15 if(s.length() % 2 != 0) {  
16 flippedString += s.charAt(size);  
17 }  
18 return flippedString;  
19 }  
20 }