CS 312 – Exam 2 – Spring 2023 SOLUTIONS

Boolean Expressions - 2 points each

1. true 2. false 3. true 4. yes

Code Tracing - 2 points each

1. `Keeley`
2. -6, -3, 0, 3, 6, 9, 12, 15, 18, 21
3. smallest output: 100.0
   all numbers are less than: 100.5
4. `***karma***`
5. `6`
6. `infinite loop`
7. `8 8 3`
8. `0 7.2 1`
9. `116.12ErasMean45`
10. `5 13`
11. `[-2, 0, 1, 5, 19, 47]`
12. `[10, 5, 7, -6, 24]`
13. `[-14, 56, 1, 0, 6][0, 0, 0, -1, -2]`
14. `[6, 4, 4]`
15. `[3, 7, 7, 7][4, 5, 5, 7][5, 7, 7, 8]`
16.

```
Name~~~~Number
Saul~~~~~81.90
Sweta~~~411.30
Jordan~~~59.12
Minerva~~~3.78
```

Assertions - 0.7 points each, max of 10

	`next == 0`	`prev == 0`	`next == prev`
Point A	S	A	S
Point B	N	S	S
Point C	N	N	A
Point D	S	N	S
Point E	A	S	S

Program - largerDigits() - 15 points

```
17public static int largerDigits1(int a, int b) {
18    int digits = 0;
19    int result = 0;
20    while (a != 0 && b != 0) {
21        if (a % 10 > b % 10) {
22            result += a % 10 * (int) Math.pow(10, digits);
23        } else {
24            result += b % 10 * (int) Math.pow(10, digits);
25        }
26        a = a / 10;
27        b = b / 10;
28        digits++;
29    }
30    return result;
31}
```

Item	Line #	Item	Points
A	17	Method declaration	+1 int return type +1 two int parameters
B	20	While loop	+3 check a != 0 and b !-=0
C	21 - 28	Results	+1 initialize result and digits/multiplier +3 select larger digit between a and b +3 adjust for place value (with pow or multiplier) +2 reduce a & b, increment digits/multiplier +1 correct return
X	N/A	Instructions	MAX of -5 -2 Unnecessary code -2 Redundant or inefficient code -2 Seriously incorrect style guide issue -2 Java syntax not allowed -2 Not following instructions (if not reflected above already)

Program - printDuplicates() - 23 points

```
16 public static void printDuplicates(Scanner input) {
17     while (input.hasNextLine()) {
18         String line = input.nextLine();
19         Scanner lineScan = new Scanner(line);
20         String token = lineScan.next();
21         int count = 1;
22         while (lineScan.hasNext()) {
23             String token2 = lineScan.next();
24             if (token2.equals(token)) {
25                 count++;
26             } else {
27                 if (count > 1) {
28                     System.out.print(token + "*" + count + " ");
29                 }
30                 token = token2;
31                 count = 1;
32             }
33         }
34
35         if (count > 1) {
36             System.out.print(token + "*" + count);
37         }
38         System.out.println();
39     }
40 }
```

Item	Line #	Item	Points
A	16	header	+1 scanner input +1 void return
B	17	while loop per line	+2 while loop for hasNextLine()
C	18-19	create Scanner for line	+2 read line from file +2 create scanner for the string
D	22	while loop per token	+2 while loop for hasNext()
E	24-25	count repetitions	+1 initialize counter and previous +3 increment if current token equal to previous token +3 reset counter if consecutive repetitions end +1 update previous
F	28, 36	print word and count	+3 when consecutive repetition ends, print word and count +1 handle consecutive repetition at end of line +1 println() in-between lines
X	N/A	Instructions	MAX of -5 -2 Unnecessary code -2 Redundant or inefficient code -2 Seriously incorrect style guide issue -2 Java syntax not allowed -2 Not following instructions (if not reflected above already)

Program - evenBeforeOdd - 12 points

```
15public static void evenBeforeOdd(int[] a) {
16    int fixed = 0;
17    for (int i = 0; i < a.length; i++) {
18        if (a[i] % 2 == 0) {
19            int temp = a[fixed];  // swap
20            a[fixed] = a[i];
21            a[i] = temp;
22            fixed++;
23        }
24    }
25}
```

Item	Line #	Item	Points
A	15	Method declaration	+1 int array parameter +1 void return type
B	17	Loop to traverse array	+3 traversal for loop (can vary - forward, backward, nested loops, etc.)
C	18-21	Odd swap	+2 check for odd number (or even) +2 swap if needed
D	28	Track index	+3 track index to indicate “already fixed” portion of array
X	N/A	Instructions	MAX of -5 -2 Unnecessary code -2 Redundant or inefficient code -2 Seriously incorrect style guide issue -2 Java syntax not allowed (for example, Arrays.sort()) -2 Not following instructions (if not reflected above already)