Solution to CSE143 Section #7 Practice Midterm

1.

Expression	Value
`5 * 6 - (4 + 3) * 2 - 2 * 3`	10
`208 / 20 / 4 + 12 / 10.0 + 0.4 * 2`	4.0
`8 - 2 + "8 - 2" + 8 * 2 + 8`	"68 - 2168"
`4 * 5 % 6 + 297 % 10 + 4 % 8`	13
`13 / 2 * 3.0 + 5.5 * 3 / 2`	26.25

2. Parameter Mystery. The program produces the following output.

```
picnic and lemonade like ants
ants and ants like y
lemonadey and antspicnic like antsy
lemonade and sunny like today
```

3.

Method Call	Output Produced
`ifElseMystery(5, 20);`	7 5
`ifElseMystery(42, 42);`	43 41
`ifElseMystery(6, 1);`	9 7
`ifElseMystery(2, 0);`	3 -1
`ifElseMystery(7, 10);`	9 7
`ifElseMystery(4, 4);`	5 3

4.

Method Call	Output Produced
`mystery(8);`	1 8
`mystery(32);`	2 5
`mystery(184);`	3 13
`mystery(8239);`	4 22

5.

	y == 0	y % 2 == 0	z == 0
Point A	never	always	always
Point B	never	always	sometimes
Point C	sometimes	always	never
Point D	never	never	sometimes
Point E	always	always	sometimes

6. One possible solution appears below.

```
public static int printSequenceTo(double value) {
    double sum = 0.5;
    System.out.print("1/2");
    int n = 1;
    while (sum < value) {
        n++;
        System.out.print(" + " + n + "/" + (n + 1));
        sum = sum + (double) n / (n + 1);
    }
    System.out.println(" = " + sum);
    return n;
}
```

7. One possible solution appears below.

```
public static void switchData(Scanner input) {
    while (input.hasNextLine()) {
        String line = input.nextLine();
        Scanner tokens = new Scanner(line);
        String label = tokens.next();
        System.out.print(label);
        while (tokens.hasNextInt()) {
            int n1 = tokens.nextInt();
            if (tokens.hasNextInt()) {
                int n2 = tokens.nextInt();
                System.out.print(" " + n2);
            }
            System.out.print(" " + n1);
        }
        System.out.println();
    }
}
```

8. Arrays. One possible solution appears below.

```
public static void minToFront(int[] list) {
    if (list.length > 0) {
        int min = 0;
        for (int i = 0; i < list.length; i++) {
            if (list[i] < list[min]) {
                min = i;
            }
        }
        int temp = list[0];
        list[0] = list[min];
        list[min] = temp;
    }
}
```

9. Programming. One possible solution appears below.

```
public static boolean isMatch(String pattern, String text) {
    int j = 0;
    for (int i = 0; i < pattern.length(); i++) {
        char ch1 = pattern.charAt(i);
        if (ch1 == '*') {
            int diff = text.length() - pattern.length() + 1;
            if (diff < 0) {
                return false;
            } else {
                j += diff;
            }
        } else {
            if (j >= text.length()) {
                return false;
            }
            char ch2 = text.charAt(j);
            j++;
            if (ch1 != '.' && ch1 != ch2) {
                return false;
            }
        }
    }
    return j == text.length();
}
```

below is a solution to the 4-point problem:

```
   public static boolean isMatch(String pattern, String text) {
        if (text.length() != pattern.length()) {
            return false;
        }
        for (int i = 0; i < pattern.length(); i++) {
            char ch1 = pattern.charAt(i);
            char ch2 = text.charAt(i);
            if (ch1 != '.' && ch1 != ch2) {
                return false;
            }
        }
        return true;
    }
```