COMP1405 - Assignment #3

(Due: **Monday, October 19th at 9:00AM**)

(1) Sodoku

Beginning with the **SodokuBoardDisplayProgram** from chapter 5 of the notes, write a function called **isComplete()** that takes a **byte[][]** representing a Sodoku board and returns a **boolean** indicating whether or not the board is "*complete*". Note that in order to be "*complete*", the following conditions must be true (as shown in the picture below):

- each row of the board must have the digits 1 through 9.
- each column of the board must have the digits 1 through 9.
- each 3x3 square of the board must have the digits 1 through 9.

In the test code, **board2** should be complete while **board1** is not. Below are some additional boards to try out (only board 7 below is complete). Make sure that your main program tests them all out and displays the proper answer as a result of calling your **isComplete()** method for each board.

```
byte[][]     board3 = {{5,3,4,6,7,8,9,1,2},
                       {6,7,2,1,9,5,1,4,8},
                       {1,9,8,3,4,2,5,6,7},
                       {8,5,9,7,6,1,4,2,3},
                       {4,2,6,8,5,3,7,9,1},
                       {7,1,3,9,2,4,8,5,6},
                       {9,6,1,5,3,7,2,8,4},
                       {2,8,7,4,1,9,6,3,5},
                       {3,4,5,2,8,6,1,7,9}};

byte[][]     board4 = {{5,3,4,6,7,8,9,1,2},
                       {6,7,2,1,9,5,3,4,8},
                       {1,9,8,3,4,2,5,6,7},
                       {8,5,9,7,6,1,4,2,3},
                       {4,2,6,7,5,3,7,9,1},
                       {7,1,3,9,2,4,8,5,6},
                       {9,6,1,5,3,7,2,8,4},
                       {2,8,7,4,1,9,6,3,5},
                       {3,4,5,2,8,6,1,7,9}};

byte[][]     board5 = {{1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9},
                       {1,2,3,4,5,6,7,8,9}};

byte[][]     board6 = {{1,1,1,1,1,1,1,1,1},
                       {2,2,2,2,2,2,2,2,2},
                       {3,3,3,3,3,3,3,3,3},
                       {4,4,4,4,4,4,4,4,4},
                       {5,5,5,5,5,5,5,5,5},
                       {6,6,6,6,6,6,6,6,6},
                       {7,7,7,7,7,7,7,7,7},
                       {8,8,8,8,8,8,8,8,8},
                       {9,9,9,9,9,9,9,9,9}};
```

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

```
byte[][]     board7 = {{1,2,3,4,5,6,7,8,9},
                       {4,5,6,7,8,9,1,2,3},
                       {7,8,9,1,2,3,4,5,6},
                       {2,3,1,5,6,4,8,9,7},
                       {5,6,4,8,9,7,2,3,1},
                       {8,9,7,2,3,1,5,6,4},
                       {3,1,2,6,4,5,9,7,8},
                       {6,4,5,9,7,8,3,1,2},
                       {9,7,8,3,1,2,6,4,5}};

byte[][]     board8 = {{1,2,3,4,5,6,7,8,9},
                       {2,3,1,5,6,4,8,9,7},
                       {3,1,2,5,4,5,9,7,8},
                       {4,5,6,7,8,9,1,2,3},
                       {5,6,4,8,9,7,2,3,1},
                       {6,4,5,9,7,8,3,1,2},
                       {7,8,9,1,2,3,4,5,6},
                       {8,9,7,2,3,1,5,6,4},
                       {9,7,8,3,1,2,6,4,5}};
```

---

(2) Fishing

Below is a program called **LakeSimulationProgram** that attempts to simulate two people catching some fish in two different lakes. The lakes are represented by two unique arrays ... one array represents the kinds of each fish in the lake and the other represents the size (in cm) of the fish. These two arrays, for example, represent a lake with exactly three fish in it:

```
String[]  kinds = {"Pike","Pike","Pike"};
int[]     sizes = {35, 6, 10};
```

People are allowed to fish in the lake and they may keep up to at most 3 fish that they catch, but no more. Also, they are not allowed to keep Sunfish nor are they allowed to keep any fish which is less than 10cm in size. When a person fishes, they keep their fish in two similar-looking arrays as follows:

```
String[]  kinds = new String[3];
int[]     sizes = new int[3];
int       count = 0;
```

Note that the above arrays allow for at most three fish to be stored. The count indicates the number of fish currently caught and kept, which is different from the capacity of the array, which is fixed at 3.

You are to examine the code below and properly complete the 4 incomplete static methods

```
public class LakeSimulationProgram {

    private static int  LIMIT = 3;               // max # of fish that can be caught
    private static int  MIN_KEEP_FISH_SIZE = 10; // fish < this size are to be thrown back
```

```
    // Display to the System console a list of the fish represented by the given arrays
    public static void listFish(String[] kinds, int[] sizes) {

        // ... COMPLETE THIS METHOD ... //

    }
```

```
    // Return true or false whether or not the fish can be kept
    // given its kind, size and the number of fish caught so far
    public static boolean shouldKeep(String kind, int size, int catchCount) {
```

```
        // ... COMPLETE THIS METHOD ... //

    }
```

```
    // Simulate a fisher trying to catch some fish and return #fish caught.
    public static int goFish(String[] lakeKinds, int[] lakeSizes, String[] fisherKinds,
                             int[] fisherSizes, int fisherCount) {

        // ... COMPLETE THIS METHOD (MUST CALL shouldKeep() METHOD PROPERLY)... //

    }
```

```
    // Simulate fish being thrown into a lake and return #fish returned successfully
    public static int throwBack(String[] lakeKinds, int[] lakeSizes, String[] fisherKinds,
                                int[] fisherSizes, int fisherCount) {

        // ... COMPLETE THIS METHOD ... //

    }
```

```
    public static void main(String [] args) {
        // Create a full lake with 10 fish
        String[]  whiteLakeFishKinds = {"Sunfish","Pickerel","Bass","Perch","Sunfish",
                                        "Pickerel","Pickerel","Bass","Sunfish","Sunfish"};
        int[]     whiteLakeFishSizes = {4, 25, 20, 30, 4, 15, 9, 12, 5, 12};

        // Create another lake with 3 fish
        String[]  silverLakeFishKinds = {"Pike","Pike","Pike"};
        int[]     silverLakeFishSizes = {35, 6, 10};

        // Create two people to fish in the lakes
        String[]  fredsFishKinds = new String[LIMIT];
        int[]     fredsFishSizes = new int[LIMIT];
        int       fredsFishCount = 0;

        String[]  suzysFishKinds = new String[LIMIT];
        int[]     suzysFishSizes = new int[LIMIT];
        int       suzysFishCount = 0;

        // Display the fish in the lakes
        System.out.println("White Lake's fish to begin:");
        listFish(whiteLakeFishKinds, whiteLakeFishSizes);
        System.out.println("Silver Lake's fish to begin:");
        listFish(silverLakeFishKinds, silverLakeFishSizes);

        // Simulate Fred trying to catch some fish in White Lake
        System.out.println("Fred attempts to catch some fish in White Lake ...");
        fredsFishCount += goFish(whiteLakeFishKinds, whiteLakeFishSizes,
                                 fredsFishKinds, fredsFishSizes, fredsFishCount);

        System.out.println("\nFred's fish now:");
        listFish(fredsFishKinds, fredsFishSizes);
        System.out.println("White Lake's fish now:");
        listFish(whiteLakeFishKinds, whiteLakeFishSizes);


        // Simulate Suzy trying to catch some fish in White Lake
        System.out.println("Suzy attempts to catch some fish in White Lake ...");
        suzysFishCount += goFish(whiteLakeFishKinds, whiteLakeFishSizes,
                                 suzysFishKinds, suzysFishSizes, suzysFishCount);

        System.out.println("Suzy's fish now:");
        listFish(suzysFishKinds, suzysFishSizes);
        System.out.println("White Lake's fish now:");
        listFish(whiteLakeFishKinds, whiteLakeFishSizes);
```

```
        // Simulate Suzy trying to catch some fish in Silver Lake
        System.out.println("\nSuzy attempts to catch some fish in Silver Lake ...");
        suzysFishCount += goFish(silverLakeFishKinds, silverLakeFishSizes,
                                 suzysFishKinds, suzysFishSizes, suzysFishCount);

        System.out.println("\nSuzy's fish now:");
        listFish(suzysFishKinds, suzysFishSizes);
        System.out.println("Silver Lake's fish now:");
        listFish(silverLakeFishKinds, silverLakeFishSizes);

        // Now simulate Suzy throwing her fish back into White Lake
        System.out.println("Suzy attempts to throw all of her fish into White Lake ...");
        suzysFishCount -= throwBack(whiteLakeFishKinds, whiteLakeFishSizes,
                                    suzysFishKinds, suzysFishSizes, suzysFishCount);

        // Now simulate Fred throwing his fish back into White Lake
        System.out.println("Fred attempts to throw all of his fish into White Lake ...");
        fredsFishCount -= throwBack(whiteLakeFishKinds, whiteLakeFishSizes,
                                    fredsFishKinds, fredsFishSizes, fredsFishCount);

        // Now let us see where the fish are
        System.out.println("\nSuzy's fish now:");
        listFish(suzysFishKinds, suzysFishSizes);
        System.out.println("Fred's fish now:");
        listFish(fredsFishKinds, fredsFishSizes);
        System.out.println("Silver Lake's fish now:");
        listFish(silverLakeFishKinds, silverLakeFishSizes);
        System.out.println("White Lake's fish now:");
        listFish(whiteLakeFishKinds, whiteLakeFishSizes);
    }
}
```

Here is the expected output that you should see once your program is working:

```
White Lake's fish to begin:
4 cm Sunfish
25 cm Pickerel
20 cm Bass
30 cm Perch
4 cm Sunfish
15 cm Pickerel
9 cm Pickerel
12 cm Bass
5 cm Sunfish
12 cm Sunfish

Silver Lake's fish to begin:
35 cm Pike
6 cm Pike
10 cm Pike

Fred attempts to catch some fish in White Lake ...
Caught a good one!
Caught a good one!
Caught a good one!

Fred's fish now:
25 cm Pickerel
20 cm Bass
30 cm Perch

White Lake's fish now:
4 cm Sunfish
4 cm Sunfish
15 cm Pickerel
```

```
9 cm Pickerel
12 cm Bass
5 cm Sunfish
12 cm Sunfish

Suzy attempts to catch some fish in White Lake ...
Caught a good one!
Caught a good one!
Suzy's fish now:
15 cm Pickerel
12 cm Bass

White Lake's fish now:
4 cm Sunfish
4 cm Sunfish
9 cm Pickerel
5 cm Sunfish
12 cm Sunfish


Suzy attempts to catch some fish in Silver Lake ...
Caught a good one!

Suzy's fish now:
15 cm Pickerel
12 cm Bass
35 cm Pike

Silver Lake's fish now:
6 cm Pike
10 cm Pike

Suzy attempts to throw all of her fish into White Lake ...
Fred attempts to throw all of his fish into White Lake ...

Suzy's fish now:

Fred's fish now:
25 cm Pickerel

Silver Lake's fish now:
6 cm Pike
10 cm Pike

White Lake's fish now:
4 cm Sunfish
35 cm Pike
12 cm Bass
15 cm Pickerel
4 cm Sunfish
30 cm Perch
9 cm Pickerel
20 cm Bass
5 cm Sunfish
12 cm Sunfish
```

NOTE: Submit a .zip file of a folder that contains all of your **.java** files. Submit your assignment using **CULearn**. Note that if your internet connection at home is down or does not work, we will not accept this as a reason for handing in an assignment late ... so make sure to submit the assignment WELL BEFORE it is due !

Please NOTE that you WILL lose marks on this assignment if any of your files are missing. You will also lose marks if your pseudocode is not written neatly with proper indentation. See examples in the notes for proper style.
