German University in Cairo
Media Engineering and Technology
Prof. Dr. Slim Abdennadher
Dr. Wael Abouelsaadat
Dr. Mohammed Abdel Megeed

Introduction to Computer Programming, Spring Term 2017
Practice Assignment 9

Discussion: 13.5.2017 - 18.5.2017

Exercise 9-1 Subset

Write a Java method `subset` that takes two arrays of integers as parameters and **returns** `true` if and only if the first array is a subset of the second array, otherwise the method should return `false`. Assume that the arrays do not consist of duplicates. For example:

```
subset({1,2,3}, {1,2,3,5,6}) returns true
subset({1,2,3}, {2,4,5,1,3}) returns true
subset({}, {1,2,3,5,6}) returns true
subset({1,2,3}, {2,4,5,1}) returns false
```

Write a `main` method to test your program. The `main` method should display either

```
Array 1 is a subset of  Array 2
```

or

```
Array 1 is not a subset of Array 2
```

Solution:

```
class Subset
{

public static boolean member(int x, int[] a) {
    int n = a.length;
    for (int i = 0; i < n; i++) {
        if (x == a[i]) return true;
    }
    return false;
}


public static boolean subset(int[] sub, int[] sup) {
    int m = sub.length;
    for (int i = 0; i < m; i++)
        if (!member(sub[i], sup)) return false;
    return true;
}



    public static void main (String[] args) {
```

```
        int[] a = {1,2,6};
        int[] b = {1,2,6,3,7,4,8,5};

        if (subset(a,b))
           System.out.println("Array 1 is contained in Array 2");
        else
           System.out.println("Array 1 is not contained in Array 2");
    }
}
```

Exercise 9-2 Matrix Addition
To be discussed in the Labs

Matrix addition involves taking two 2-D arrays of the same height and width (let us call them A and B) and then, for each position in the matrices, adding the value from `A` in that position to the value from `B` in that position and placing it is a third matrix, `C`, in that position. Thus, matrix addition takes this form (for two 3×3 matrices):

$$\begin{pmatrix} A_1 & A_2 & A_3 \\ A_4 & A_5 & A_6 \\ A_7 & A_8 & A_9 \end{pmatrix} + \begin{pmatrix} B_1 & B_2 & B_3 \\ B_4 & B_5 & B_6 \\ B_7 & B_8 & B_9 \end{pmatrix} = \begin{pmatrix} A_1 + B_1 & A_2 + B_2 & A_3 + B_3 \\ A_4 + B_4 & A_5 + B_5 & A_6 + B_6 \\ A_7 + B_7 & A_8 + B_8 & A_9 + B_9 \end{pmatrix}$$

Write a method that accepts 3 2-D arrays of `double` (that is, three matrices). This method should determine whether one of the matrices is the result of matrix addition of the other two.

Hint: Break your solution into several methods.

Solution:

```
public static double[][] whichIsSum (double[][] x, double[][] y, double[][] z) {
        if (isSum(x, y, z)) {
                return x;
        }

        else {
            if (isSum(y, x, z)) {
                return y;
        }
        else {
            if (isSum(z, x, y)) {
                return z;
                }
                else
                        return null;
                }
        }
}
public static boolean isSum (double[][] s, double[][] a, double[][] b) {
        for (int i = 0; i < s.length; i++)
                for (int j = 0; j < s[0].length; j++)
                        if (s[i][j] != a[i][j] + b[i][j])
                                return false;

        return true;
}
```

Exercise 9-3 2-D Arrays

Write a Java program that given a two-dimensional array, reorders the rows such that the row with the highest row sum is the first row.

If the program will be called with the following array:

```
1  3  5  9
2  100
2  2  3
```

then the output should be

```
2  100
1  3  5  9
2  2  3
```

The following steps should be performed:

a) Calculate row sum

b) Find index of row with maximum sum

c) Swap row of maximum sum with row 0

Solution:

```
public class Array2d {
        public static void main(String[] args) {

                int[][] m= new int[][]{new int[]{1,3,5,9},
                                       new int[]{2,100},
                                       new int[]{2,2,3}   };

                int[] rsum         = new int[m.length];   //array of row sums
                int highSum        = 0;                   //max row sum so far
                int highIndex = 0;                    //index of row with highSum

                for (int r=0; r<m.length; r++) {

                        //calculate row sum
                        for (int c=0; c<m[r].length; c++)
                                rsum[r] += m[r][c];

                        //if current row sum is highest, update highSum, highIndex
                        if (rsum[r] >= highSum) {
                                highSum   = rsum[r];
                                highIndex = r;
                        }
                }

                //swap row with highest sum into first row
                int[] tmp= m[0];
                m[0]= m[highIndex];
                m[highIndex]= tmp;

                //Print 2-d array
                for (int r=0; r<m.length; r++){
```

```
                    for (int c=0; c<m[r].length; c++)
                        System.out.print(m[r][c]+ "  ");
                    System.out.println();
                }
        }
}
```

Exercise 9-4 Two-dimensional array evaluation - Final Spring 2012
To be discussed in Tutorial

Given a two-dimensional, possibly ragged, array of booleans, write a method that evaluates the array such that the value of every row is the conjunction (logical AND) of all values in the row, and the value of the complete array is the disjunction (logical OR) of all row values

Solution:

```
public static boolean evaluate(boolean[][] a) {
    boolean formula = false;
    for (int i = 0; i < a.length; i++) {
        boolean clause = true;
        for (int j = 0; j < a[i].length; j++)
            clause = clause && a[i][j];
        formula = formula || clause;
    }
    return formula;
}
```

Exercise 9-5 Pattern Sequence - Final Spring 2013
To be discussed in Tutorial

Write a java method `isPatternSequence`, which tells if its array argument is an example of the sequence

`{{1},{2,2},{3,3,3},{4,4,4,4}....}`

ending with an array with k copies of value k, for some k.

Example:

```
Call:isPatterSequence({{1}})
Output: true

Call:isPatterSequence({{1},{2,2}})
Output: true

Call:isPatterSequence({{1},{2,2},{3,3,3},{4,4,4,4}})
Output: true

Call:isPatterSequence({{1},{2,2},{3,3,3,3}})
Output: false

Call:isPatterSequence({{1},{2,2},{3,3,3},{4,4,3,4}})
Output: false
```

Solution:

```
public static boolean isPatternSequence(int[][] data) {
    for (int i = 0; i < data.length; i++) {
        if (data[i].length == i + 1) {
        for (int j = 0; j < data[i].length; j++) {
            if (data[i][j] != i + 1)
                return false;
            }
        }

        else
            return false;
    }
    return true;
}
```

Exercise 9-6 Decimal to Binary

Write a Java program `DecToBin` to convert a decimal number to binary. Using a command-line argument, input a positive decimal integer. Convert the decimal number to binary and output the result on the console. Name your class `DecToBin`. For example:

```
PROMPT>java DecToBin 2567
Binary: 101000000111
Thank you
```

Your solution should include basic input verification, as illustrated in the following examples.

Hint You may use `boolean isDigit(char ch)` that determines if the specified character `ch` is a digit.

```
PROMPT>java DecToBin 26 104
Usage: java DecToBin positive_decimal_number

PROMPT:>java DecToBin 25abc
Argument format error
```

Solution:

```
public class DecToBin
{
        public static void main(String[] args)
        {
                // first check for proper invocation
                if (args.length != 1)
                {
                        System.out.println("Usage: java DecToBin decimal_value");
                        return;
                }

                // check for proper argument (before parsing)
                for (int i = 0; i < args[0].length(); i++)
                        if (!Character.isDigit(args[0].charAt(i)))
                        {
                                System.out.println("argument format error");
                                return;
```

```
                }

                // convert input string to decimal integer
                int decimal = Integer.parseInt(args[0]);

                // do the conversion
                String result = "";
                do
                {
                        if (decimal % 2 == 0)
                                result = "0" + result;
                        else
                                result = "1" + result;
                        decimal = decimal / 2;
                } while (decimal > 0);

                // print the result
                System.out.println(result);
        }
}
```