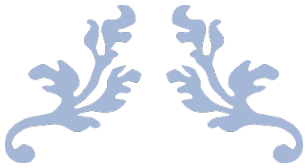


MSCS 630 – SECURITY ALGORITHMS AND PROTOCOLS

Lab 3

FEBRUARY 27, 2018

THOMPSON RAJAN
CWID: 20082947

1. Finding Determinant of a Matrix

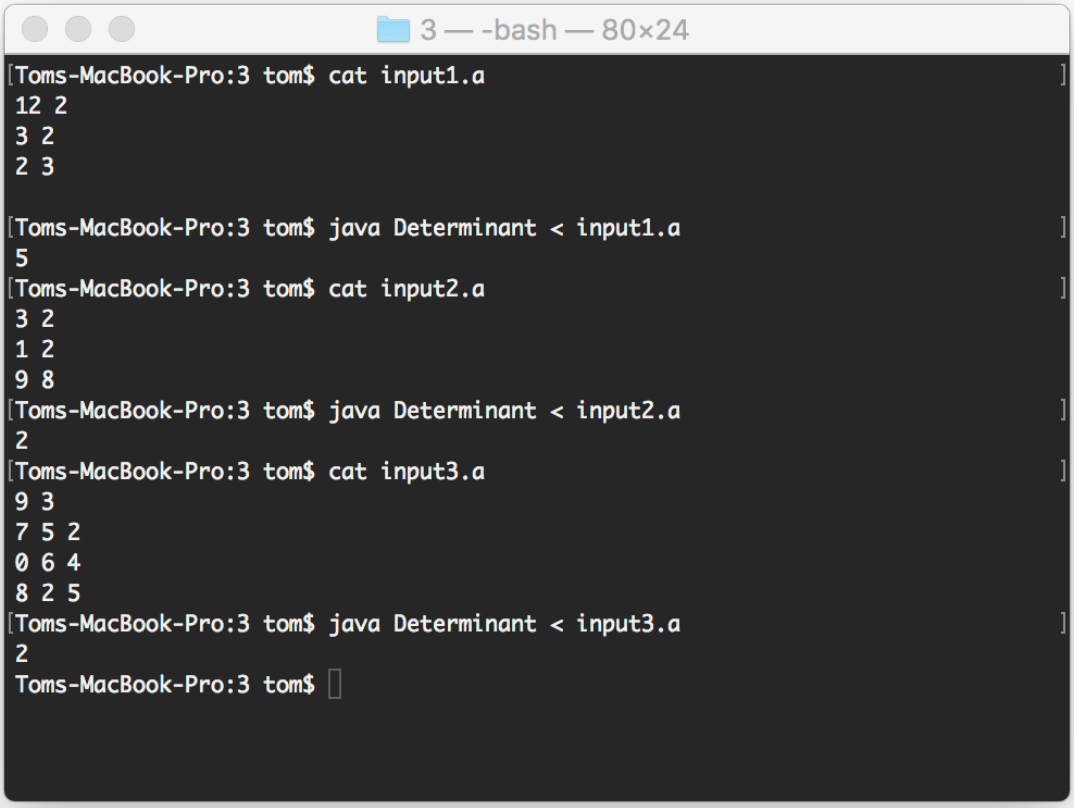
```
/**
 * File: Determinant.java
 * Author: Thompson Rajan
 * Course: MSCS 630 Security Algorithms and Protocols
 * Assignment: Lab 3
 * Due Date: Wednesday, February 28, 2018
 * Version: 1.0
 *
 * This file contains the implementation to estimate the determinant of a
 * matrix in modulo m
 */

import java.util.Scanner;

/**
 * This class estimates the determinant of an n x n matrix in modulo m
 */
public class Determinant {

  /**
   * This method calculates the determinant of a matrix recursively in modulo m.
   * @param m - Modulo input
   * @param A - Given matrix
   * @return - Determinant of matrix A in modulo m
   */
  public static int cofModDet(int m, int[][] A){
    int det = 0;
    int n = A.length;
    int sign = 1;
    int p = 0;
    int q = 0;

    // Return first element in mod m if the matrix size is 1
    if(n == 1){
      det = A[0][0] % m;
    }
    else{

      // Initialize sub-matrix to store co-factors.
      int b[][] = new int[n-1][n-1];
      for(int x = 0 ; x < n ; x++){

        // Indices to pick co-factor elements
        p=0;
        q=0;

        for(int i = 1; i < n; i++){
          for(int j = 0; j < n; j++){
            if(j != x){

              // Pick co-factor for corresponding element
              b[p][q++] = A[i][j] % m;
              if(q % (n-1) == 0){

                // Reset column when at last column of the current row &
                // move to next row
                p++;
```

```
                q=0;
              }
            }
          }
        }
        // Estimate determinant from co-factors
        det = det + A[0][x] * cofModDet(m,b) * sign;

        // Flip signs
        sign = sign * (-1);
      }
    }
    return det % m;
  }

  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    // Get modulo m and matrix size n
    int m = in.nextInt();
    int n = in.nextInt();
    int A[][] = new int[n][n];

    // Get matrix input
    for(int i= 0; i < n ; i++) {
      for (int j = 0; j < n; j++) {
        A[i][j] = in.nextInt() % m;
      }
    }
    System.out.println(cofModDet(m, A));
  }
}
```

Date: 02-27-18

Thompson Rajan
CWID: 20082947

Output:



Date: 02-27-18

Thompson Rajan
CWID: 20082947

2. Converting a Plaintext to a Hex-Matrix

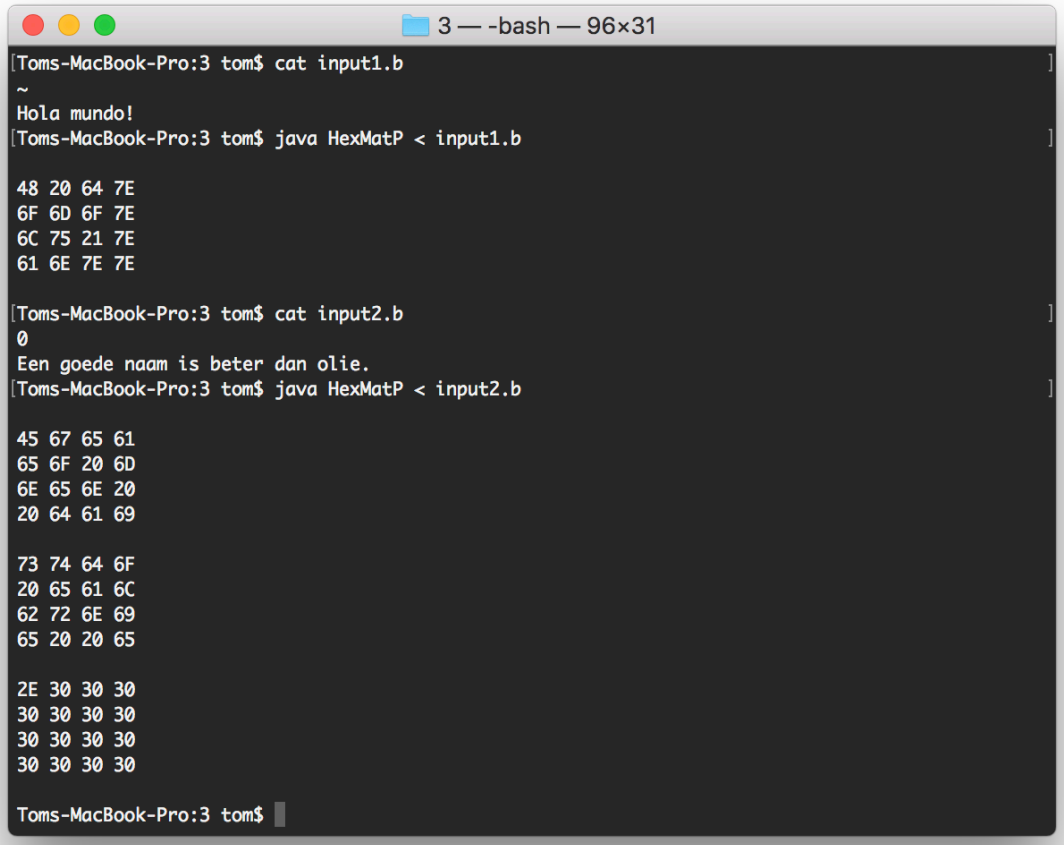
```
/**
 * File: HexMatP.java
 * Author: Thompson Rajan
 * Course: MSCS 630 Security Algorithms and Protocols
 * Assignment: Lab 3
 * Due Date: Wednesday, February 28, 2018
 * Version: 1.0
 *
 * This file contains the implementation to convert a plaintext to a
 * padded hex-matrix
 */

import java.util.Scanner;

/**
 * This class is used to convert a plaintext to hex matrix padded with a give
 * character s
 */
public class HexMatP {

  /**
   * This method takes a padding a character and a string representing a
   * plaintext and returns a padded hex-matrix.
   * @param s - Padding character
   * @param p - Plaintext string
   * @return - Returns padded hex - matrix
   */
  static int [][] getHexMatP(char s, String p){

    char m[] = p.toCharArray();

    // Initialize temporary 4 x 4 matrix to collect values in 4 x 4 order.
    int t[][] = new int[4][4];

    // This matrix is used to collect ordered values from t[][].
    int mat[][] = new int[(m.length / 16 + 1) * 4][4];

    // Indices for finalized matrix mat[][]
    int x = 0;
    int y = 0;


    System.out.println();

    // Trace current character
    int i = 0;

    while (i < m.length) {
      for (int j = 0; j < 4; j++) {
        for (int k = 0; k < 4; k++) {

          // Store ASCII if less than plaintext length.
          if (i < m.length) {
            t[j][k] = m[i];
            i++;
          }
```

```
            // Pad if greater than plaintext length.
            else {
              t[j][k] = s;
              i++;
            }
          }
        }

        // Store transposed matrix to the final matrix.
        for(int a = 0; a < 4; a++){
          for(int b = 0; b < 4; b++){
            mat[x][y++] = t[b][a];
            if(y == 4)
              y = 0;
          }
          x++;
        }
      }
      return mat;
    }

    public static void main(String[] args) {

      // Get padding character and plaintext.
      Scanner in = new Scanner(System.in);
      char s = in.nextLine().charAt(0);
      String p = in.nextLine();

      // Get hex-matrix
      int mat[][] = getHexMatP(s,p);

      for(int i=0; i < mat.length;i++){
        for(int j= 0 ; j< mat[0].length; j++){
          System.out.print(Integer.toHexString(mat[i][j]).toUpperCase() + " ");
        }
        System.out.println();
        if((i+1) % 4 == 0 ){
          System.out.println();
        }
      }
    }
}
```

Date: 02-27-18

Thompson Rajan
CWID: 20082947

Output:



Date: 02-27-18

Thompson Rajan
CWID: 20082947