

# 社工情報処理実習 1

## – 課題 9 の解答例 –

安藤 和敏

筑波大学社会工学系

2000年6月20日

### 1. 行列の積

これができずに、

```
mars24: [57]: java Matrix
java.lang.ArrayIndexOutOfBoundsException: 3
    at Matrix.time(Compiled Code)
    at Matrix.main(Compiled Code)
```

というエラーをもらって、それ以上分からないという人が多かった。`java.lang.ArrayIndexOutOfBoundsException` というエラーについては、授業中にも説明しましたし、教科書にも書いてある通り、「配列の範囲を越えて配列の中身を参照しようとして起こった」エラーです。

この間違いが発生した原因は、二次元配列  $Z$  の第  $(i, j)$  成分を

$$Z[i][j]=X[i][1]*Y[1][j]+X[i][2]*Y[2][j]+X[i][3]*Y[3][j];$$

として計算しているからです。これは

$$Z[i][j]=X[i][0]*Y[0][j]+X[i][1]*Y[1][j]+X[i][2]*Y[2][j];$$

とすべきです。(私のヒントがかえって、悪かったのでしょうか。)

プログラムはファイル 1 を見よ。

誰もやってくれなかったのですが、 $A^3B^2$  は、

```
product(product(product(A,A),A),product(B,B));
```

とやれば計算できます。実際に任意の整数  $m, n \geq 1$  に対して、 $A^m B^n$  が `product` メソッドを使って計算できます。分かりますか？

### 2. 二項分布

プログラムの例はファイル 2 を見よ。

### 3. 電子辞書

プログラムの例はファイル 3 を見よ。

ちょっと変えれば、調べるべき単語をコマンドラインから与えることができます。ファイル 3 を見よ。

さらに、和英にしたいのであれば、

```

public class matrix {

    public static void main (String[] args) {

        double[][] A,B,C;
        A = new double[][] { {1,2,3}, {4,5,6}, {7,8,9} };
        B = new double[][] { {1,4,7}, {2,5,8}, {3,6,9} };

        printMatrix(product(A,B));
    }

    public static double[][] product (double[][] X, double[][] Y) {

        double[][] Z = new double[3][3];

        for (int i=0; i<3; i++) {
            for (int j=0; j<3; j++) {
                double tmp = 0.0;
                for (int k=0; k<3; k++) {
                    tmp = tmp +X[i][k]*Y[k][j]; //第(i,j)成分を計算
                }
                Z[i][j] = tmp;
            }
        }

        return Z;
    }

    public static void printMatrix (double[][] X) {

        for (int i=0; i< X.length;i++) {

            for (int j=0;j<X[i].length;j++) {

                System.out.print(X[i][j]+ " ");
            }
            System.out.println("");
        }
    }
}

```

ファイル 1.1: matrix.java

```

public class coin3 {

    public static int numOfheads (int N) { // numOfheads メソッドの定義

        int head = 0; // 表の出る回数を数える変数

        for (int i=0; i<N; i++) { // 以下を N 回繰り返す
            if (Math.random() <= 1/2.0) { // もし表がでたら
                head ++; // head に 1 を足す
            }
        } // for 文の終わり
        return head; // head を返す

    } // numOfheads メソッドの定義終了

    public static void printGraph (int n) { //printGraph メソッド

        for (int i=0; i<n;i++) {
            System.out.print("*");
        }
        System.out.println("");

    }

    public static void main (String[] args) { // main メソッドの定義

        int frequency[];
        frequency = new int[101];

        for (int k=0; k<500; k++) { // 以下を 500 回繰り返す
            int heads = numOfheads(100); // 乱数を発生させて
            frequency[heads]++; // その度数をカウント
        }

        for (int i=0; i<101; i++) { // ヒストグラムを書く。
            printGraph(frequency[i]);
        }

    } // main メソッドの定義終了

} // coin3 クラスの終わり

```

ファイル 2.1: coin3.java

```

import java.io.*;

public class dict {

    public static void main(String[] args) {
        BufferedReader reader =
            new BufferedReader(new InputStreamReader(System.in),1);

        String tango[][] = { {"document", "文書"}, {"bird", "鳥"},
                              {"human", "人間"}, {"face", "顔"},
                              {"array", "配列"}, {"water", "水"},
                              {"good", "良い"} };

        int N = 6; // 単語数

        try {

            String line = reader.readLine(); // 英単語を読み込み
                                             // line 変数に代入する。

            boolean found = false; // 単語が見付かれれば true にする
                                   // 変数 */

            int i=0; // 配列の添字、最初は 0 から

            while (!found && i < N) { // 単語が見付かるか、あるいは、
                                     // 配列の最後に到達するまで
                                     // 以下を繰り返す。*/

                if (line.equals(tango[i][0])) { // もし見付かれれば
                    System.out.println(tango[i][1]); // 対応する和訳を出力
                    found = true; // found を true にする
                }

                i++; // 添字を一つ大きくする
            } // while の終り

            if (!found) { // 見付からなかったら、
                System.out.println("該当する単語は見付かりませんでした。");
            }

        } catch (IOException e) { // try の終り。
            System.out.println(e);
        } catch (NumberFormatException e) {
            System.out.println("入力が正しくありません。");
        }

    } // main メソッドの終り

} // dict クラスの終り

```

ファイル 3.1: dict.java

```

import java.io.*;

public class dict2 {

    public static void main(String[] args) {
        BufferedReader reader =
            new BufferedReader(new InputStreamReader(System.in),1);

        String tango[][] = { {"document", "文書"}, {"bird", "鳥"},
                              {"human", "人間"}, {"face", "顔"},
                              {"array", "配列"}, {"water", "水"},
                              {"good", "良い"}
                              };

        int N = 6;           // 単語数

        if (args.length != 1) {
            System.out.println("使い方: java dict2 単語");
            System.exit(0);
        }

        String line = args[0];           // コマンドラインの文字列を
                                         // line 変数に代入する。

        boolean found = false;           /* 単語が見付かれれば true にする
                                         変数 */

        int i=0;                          // 配列の添字、最初は 0 から

        while (!found && i < N) {         /* 単語が見付かるか、あるいは、
                                         配列の最後に到達するまで
                                         以下を繰り返す。*/

            if (line.equals(tango[i][0])) { // もし見付かれれば
                System.out.println(tango[i][1]); // 対応する和訳を出力
                found = true;                  // found を true にする
            }

            i++;                             // 添字を一つ大きくする
        }                                     // while の終り

        if (!found) {                      // 見付からなかったら、
            System.out.println("該当する単語は見付かりませんでした。");
        }

        } // main メソッドの終り
} // dict クラスの終り

```

ファイル 3.2: dict2.java

```
if (line.equals(tango[i][0])) { // もし見付かれれば
    System.out.println(tango[i][1]); // 対応する和訳を出力
    found = true; // found を true にする
}
```

を

```
if (line.equals(tango[i][1])) { // もし見付かれれば
    System.out.println(tango[i][0]); // 対応する和訳を出力
    found = true; // found を true にする
}
```

に変えれば良い。(0 と 1 を入れ換えただけ。)