Ｊａｖａプログラミング I

１１回目　多次元配列　課題

１．int 型の 20 行 20 列の 2 次元配列を宣言しなさい。この配列に 1 または 0 を以下の規則で代入しなさい。

（規則）i 行 j 列の配列要素に、

・「i+j の値が 6 で割り切れる」または「i-j の値が 6 で割り切れる」場合　→ 1 を代入

・そうでない場合　→ 0 を代入

この配列を画面出力する際に、1 と 0 をそれぞれ□と■として画面に表示しなさい。

ヒント：2 重 for 文と if 文を用いて上手に値を各配列要素に代入していきましょう

（実行例）

□■■■■■□■■■■■□■■■■■□■

■□■■■□■□■■■□■□■■■□■□

■■□■□■■■□■□■■■□■□■■■

■■■□■■■■■□■■■■■□■■■■

■■□■□■■■□■□■■■□■□■■■

■□■■■□■□■■■□■□■■■□■□

□■■■■■□■■■■■□■■■■■□■

■□■■■□■□■■■□■□■■■□■□

■■□■□■■■□■□■■■□■□■■■

■■■□■■■■■□■■■■■□■■■■

■■□■□■■■□■□■■■□■□■■■

■□■■■□■□■■■□■□■■■□■□

□■■■■■□■■■■■□■■■■■□■

■□■■■□■□■■■□■□■■■□■□

■■□■□■■■□■□■■■□■□■■■

■■■□■■■■■□■■■■■□■■■■

■■□■□■■■□■□■■■□■□■■■

■□■■■□■□■■■□■□■■■□■□

□■■■■■□■■■■■□■■■■■□■

■□■■■□■□■■■□■□■■■□■□

２．行列Aを用いて int 型 2 次元配列 array を初期化しなさい。次に、同じ大きさの 4 行 4 列の int 型 2 次元配列 t_array を作成しなさい。行列Aの転置行列 ${}^tA$を配列 t_array に求めるコードを書きなさい。転置行列とは i 行 j 列の値と j 行 i 列の値を入れかえた行列です。

$$A=\begin{pmatrix}1 & 2 & 3 & 4\\ 5 & 6 & 7 & 8\\ 9 & 10 & 11 & 12\\ 13 & 14 & 15 & 16\end{pmatrix}$$

ヒント：2 重 for 文を用いて array[i][j]の値を t_array[j][i]に代入していきましょう

（実行例）
転置行列を求めます
行列：
```
 1  2  3  4
 5  6  7  8
 9 10 11 12
13 14 15 16
```
転置行列：
```
 1  5  9 13
 2  6 10 14
 3  7 11 15
 4  8 12 16
```

３．行列A, Bを用いてそれぞれ int 型 2 次元配列 array_A と array_B を初期化しなさい。その積を 3 行 3 列の配列 array_AB に求めるコードを書きなさい。

$$A=\begin{pmatrix}1 & 2 & 1 & -3\\ -2 & 1 & -2 & 0\\ 1 & -1 & 1 & -1\end{pmatrix}, B=\begin{pmatrix}1 & 1 & 2\\ 2 & -3 & -1\\ -1 & 2 & 1\\ 2 & 1 & 2\end{pmatrix}$$

ヒント：配列 array_AB[i][j]を求めるときも for 文を用います。全部で 3 重 for 文です。

（実行例）
行列の積を求めます
```
  1   2   1  -3
 -2   1  -2   0
  1  -1   1  -1
X
  1   1   2
  2  -3  -1
 -1   2   1
  2   1   2
=
 -2  -6  -5
  2  -9  -7
 -4   5   2
```

４．あるクラスの学生が 3 科目の試験を受けた。学籍番号と試験結果を配列に入力して、各学生について合計を求めよ。さらに、合計に応じて順位をつけて表の形で出力するプログラムを作成しなさい。但し、学生数は最初に入力すること。

（配列の構成）学生数が num の場合：

```
String[] student_ID = new String[num];  // 各学生の学籍番号
int[][] score_table = new int[num][3];  // 各学生の 3 科目の点数
int[] total = new int[num];             // 各学生の点数の合計
int[] ranking = new int[num];           // 順位
```

（実行例）

```
成績処理を行います
学生数を入力してください
3
1 人目：
学籍番号?>A001
科目 1 の点数?>47
科目 2 の点数?>9
科目 3 の点数?>63
2 人目：
学籍番号?>A002
科目 1 の点数?>54
科目 2 の点数?>90
科目 3 の点数?>89
3 人目：
学籍番号?>A003
科目 1 の点数?>12
科目 2 の点数?>34
科目 3 の点数?>87
```

番号	科目 1	科目 2	科目 3	合計	順位
A001	47	9	63	119	3
A002	54	90	89	233	1
A003	12	34	87	133	2

５．4 行 13 列の数値パターンで int 型 2 次元配列を初期化しなさい。その後、配列に格納された 0 から 9 の各数値を以下の 1 文字で置き換えて画面に表示しなさい。このとき、改行を 1 行毎に入れて下さい。
ヒント：switch 文をうまく用いてそれぞれの値を文字に置き換えて出力しましょう。

（数値パターン）
```
0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0
2, 3, 4, 1, 5, 6, 0, 2, 3, 4, 0, 5, 6
7, 1, 8, 8, 6, 0, 0, 7, 1, 8, 8, 6, 0
0, 8, 3, 9, 0, 0, 0, 0, 8, 3, 9, 0, 0
```

（数値と文字の対応）

数値	文字
0	[空白ｽﾍﾟｰｽ]
1	_
2	<
3	' ←（表示はｴｽｹｰﾌﾟｼｰｹﾝｽで ¥' とします）
4	)
5	,
6	/
7	(
8	=
9	-

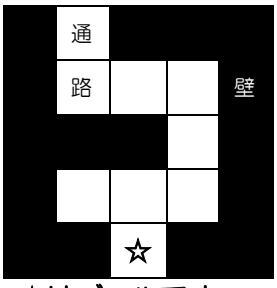
６．次の足し算ドリルと解答結果を用いて 5 行 3 列の int 型 2 次元配列を初期化しなさい。この配列を読み込み、答えが正しければ○を、誤っていれば×を実行例のように出力しなさい。
ヒント：if~else 文を用いてそれぞれの解答が正しいかどうか判断しましょう。

（足し算ドリル）　　（2 次元配列表現）
```
 2  +    3  =   5        2    3    5
 6  +    8  =  11        6    8    11
-5  +    2  =  -3        -5   2    -3
 7  +    7  =  14        7    7    14
-2  + (-6) =   8        -2   -6   8
```

（実行例）
```
○ 問 1  2+3=5
× 問 2  6+8=11
○ 問 3  -5+2=-3
○ 問 4  7+7=14
× 問 5  -2+-6=8
```

7．次の迷路を 5 行 5 列の int 型 2 次元配列 map で表現しなさい。各セルは下に示すように整数と対応付けることとします。

（迷路）



☆はゴールです

（整数との対応）

□　→　0
■　→　1
☆　→　2

（迷路の 2 次元配列表現）

```
int[][] map={
  {1, 0, 1, 1, 1},
  {1, 0, 0, 0, 1},
  {1, 1, 1, 0, 1},
  {1, 0, 0, 0, 1},
  {1, 1, 2, 1, 1}};
```

次にユーザの現在位置を 1 行 2 列の int 型 1 次元配列 user_pos で表現します。最初ユーザは map[0][1]の位置にいるとし、以下のように初期化しておきます。

```
int[] user_pos={0,1};
```

〔1〕上のマップ配列 map とユーザ現在位置 user_pos を画面に表示しなさい。ここで、ユーザ現在位置は、'・'（点）として表現することとします。

（実行例 1）

```
■・■■■
■　　■
■■■　■
■　　■
■■☆■■
```

〔2〕キーボード入力により、ユーザを移動できるようにします。入力された文字により下のようにユーザを移動させます。壁がある場合は移動せずに同じ場所に留まります。もし、ゴールに辿り着いたらプログラムを終了します。

（キーと移動方向の対応）
w　→　上へ 1 マス移動
s　→　下へ 1 マス移動
a　→　左へ 1 マス移動
d　→　右へ 1 マス移動

（実行例 2）
■・■■■
■　　■
■■■　■
■　　■
■■☆■■
s

■　■■■
■・　　■
■■■　■
■　　■
■■☆■■
d

■　■■■
■　・　■
■■■　■
■　　■
■■☆■■
：
a

■　■■■
■　　■
■■■　■
■　・　■
■■☆■■
s

■　■■■
■　　■
■■■　■
■　　■
■■・■■
ゴールです!!

８．宝探しゲームを作りましょう。5×5 の地図から掘り起こす場所をキーボード入力し、宝の場所を掘り当てます。宝の場所はランダムに決まります。掘り起こした場所に宝があった場合は「●お宝発見♪」と表示して終了します。もし、掘り起こした場所の８近傍（上下左右斜め）に宝がある場合は「●ピッピッ!!」と画面にお知らせが出ます。
ヒント：下の作成手順の例を参考してみましょう

（実行例）
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■

掘る場所を入力してください
行[0～4] ?
2
列[0～4] ?
2
●ピッピッ!!
■■■■■
■■■■■
■■・■■
■■■■■
■■■■■

掘る場所を入力してください
行[0～4] ?
2
列[0～4] ?
1
●ピッピッ!!
■■■■■
■■■■■
■・・■■
■■■■■
■■■■■

掘る場所を入力してください
行[0～4] ?
1
列[0～4] ?
1
●お宝発見♪
■■■■■
■☆■■■
■・・■■
■■■■■
■■■■■

（作成手順の例）

〔ステップ 1〕

まず、地図を 5 行 5 列の int 型 2 次元配列 map で表現します。

```
int[][] map=new int[5][5];
```

〔ステップ 2〕

各配列要素の整数の意味は次のようにします。最初すべての配列要素に 0 を入れておきます。

0 … 掘っていない場所
1 … 掘ったとき、何もなかった場所
2 … 掘ったとき、お宝があった場所

〔ステップ 3〕

次に、宝の位置をランダムで決めます。

0～4 の乱数は次のようにして得ることができます。

```
int trow, tcol;
trow=(int)(5*Math.random());  ※Math.random()は Java プログラミングⅡで解説します
tcol=(int)(5*Math.random());
```

変数 trow と tcol に 0～4 までのいずれかの整数が代入されます。

〔ステップ 4〕

続いて、各配列要素の整数を以下の記号に置き換えて地図を表示します。

0 → ■
1 → ・
2 → ☆

〔ステップ 5〕

掘り起こす場所の行番号（0～4）と列番号（0～4）をキーボードから入力し、掘り起こした場所に応じて、下記のようにメッセージを出力し、その配列要素の状態を変更します。

・宝があった場合	メッセージ「●お宝発見♪」	配列要素の値を 2 に変更
・何もなく、		
8近傍に宝がある場合	メッセージ「●ピッピッ!!」	配列要素の値を 1 に変更
8近傍に宝がない場合	メッセージなし	配列要素の値を 1 に変更

〔ステップ 6〕

〔ステップ 4〕に戻り処理を繰り返します。もし、宝が見つかったら処理を終了します。

9. ドット画面で図形を描画しましょう。8×8 のドット画面上で始点(sx,sy)と終点(dx,dy)の 2 点をキーボード入力し、2 点間をドットの直線で描画します。原点(0,0)は左上とし、右方向へ x 軸正方向(0～7)、下方向へ y 軸正方向(0～7)とします。

ヒント：ドット画面は int 型の 2 次元配列で表現し、0 を○に 1 を●として扱います

（実行例）

```
○○○○○○○○
○○○○○○○○
○○○○○○○○
○○○○○○○○
○○○○○○○○
○○○○○○○○
○○○○○○○○
○○○○○○○○
線を引きますか?(y or else)y
始点 x 座標を入力してください 4
始点 y 座標を入力してください 1
->終点 x 座標を入力してください 6
->終点 y 座標を入力してください 6
○○○○○○○○
○○○○●○○○
○○○○●○○○
○○○○○●○○
○○○○○●○○
○○○○○●○○
○○○○○○●○
○○○○○○○○
線を引きますか?(y or else)y
始点 x 座標を入力してください 4
始点 y 座標を入力してください 1
->終点 x 座標を入力してください 1
->終点 y 座標を入力してください 6
○○○○○○○○
○○○○●○○○
○○○●●○○○
○○○●○●○○
○○●○○●○○
○●○○○●○○
○●○○○○●○
○○○○○○○○
線を引きますか?(y or else)y
始点 x 座標を入力してください 2
始点 y 座標を入力してください 4
->終点 x 座標を入力してください 5
->終点 y 座標を入力してください 4
○○○○○○○○
○○○○●○○○
○○○●●○○○
○○○●○●○○
○○●●●●○○
○●○○○●○○
○●○○○○●○
○○○○○○○○
線を引きますか?(y or else)n
```