JAVA プログラミング 1

2020 年 8 月 17 日実施

プログラミング課題 2

担当　石原

**注 意 事 項**

※すべての問題は出題範囲の学習内容と記載する追加の説明を用いることで達成可能である．出題範囲以外の知識を用いてより良いコードを記述することは認める．また，出題範囲の学習内容を「必ず利用すること」を強制するものではない．
※各問題で作成する JAVA のソースファイル（○○○.java）のみを提出すること．
※各問題で作成する JAVA のソースファイル名は指定に従うこと．
※各問題で作成する JAVA のソースファイルの文字コードはシフト JIS とする．
※実行例の赤字はキーボード入力を表すものである．
※キーボード入力の数値の制約として〔型（値の範囲）〕を記載している場合は従うこと．また〔型〕とのみ記載している場合は当該型が扱える値の範囲の意味である．以下に例を示す．
〔int 型（0 ~ 5）〕は 0 から 5 までの整数
〔int 型（0 以上）〕は 0 以上で int 型が扱える値の範囲の上限までの整数
〔int 型〕は int 型が扱える値の範囲の整数

**問題 1　8 進 10 進変換**

8 進数の 4 桁の数値（$a_4a_3a_2a_1$）は 10 進数へ式 1 により変換できる．

$$10\text{進数} = a_4\times8^3 + a_3\times8^2 + a_2\times8^1 + a_1\times8^0 \qquad (式\ 1)$$

ここで各桁 $a_i$ は 0 から 7 の値をとる．下の実行例にならい，キーボード入力により 8 進数の数値を上位の桁から入力〔各桁 int 型（0 ~ 7）〕し，10 進数〔int 型〕に変換して画面に出力するコードを記述しなさい．但し，ファイル名を Code01.java として保存し，コンパイルと実行ができることとする．

〔実行例〕
8 進数を 10 進数に変換します
4 桁の 8 進数を上位より入力してください
4 桁目の値は?>0
3 桁目の値は?>1
2 桁目の値は?>2
1 桁目の値は?>3
8 進数 0123 は 10 進数で 83 です

**問題２　計算ドリル**

1 桁（数値 1~9）同士の四則演算の問題をランダムに出題するドリルを作成する．出題された問題を 2 度連続して間違えるとドリルは終了する．ここで，除算は商のみを解答する．例えば 5÷2 は 2 である．また，1 桁の数値（1~9）と四則演算の種類（4 通り）は乱数（使い方を以下に示す）を用いてランダムに選択する．下の実行例にならい，問題を表示した後，キーボード入力により解答〔int 型〕を入力させ，ドリルを実行するコードを記述しなさい．但し，ファイル名を Code02.java として保存し，コンパイルと実行ができることとする．

〔乱数の求め方〕

int 型の変数 value に整数 a から b(a<b)までの乱数を代入するコードは次の通りである．

```
int value = a + (int)(Math.random()*(b-(a-1)));
```

〔実行例１〕

```
計算問題を出題します
1 問目
 3-4 = ? -1
2 問目
 8-9 = ? 1
 8-9 = ? -1
3 問目
 8+3 = ? 11
4 問目
 2-5 = ? 2
 2-5 = ? 3
終了します
```

〔実行例２〕

```
計算問題を出題します
1 問目
 3-7 = ? -4
2 問目
 8/5 = ? 1
3 問目
 7/8 = ? 0
4 問目
   :
```

**問題３　面積推定**

三角関数の正弦 sin(x)において 0≦x≦π の弧と x 軸で囲まれる領域 A の面積を求める．

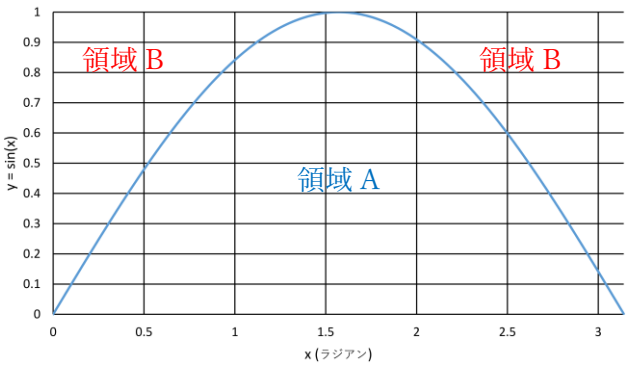


図１　正弦関数

ここでは乱数を用いて求める方法を考える．まず，変数 x〔double 型〕と y〔double 型〕に，それぞれ 0≦x≦π と 0≦y≦1 の範囲で実数の乱数（求め方を以下に示す）を生成して代入する．得られた座標(x, y)が領域 A（x 軸上と正弦線上を含む）と領域 B のいずれに含まれるかを判断してそれぞれカウントする．この乱数の生成とカウントの処理を 1 回とし 100 万回繰り返す．その後，領域 A と領域 B が占める全体の面積π×1 と，領域 A と領域 B のカウントの合計に対する領域 A のカウントの割合を用いて以下のように面積を推定する．

推定面積 = π×(領域 A カウント数)/(領域 A カウント数 + 領域 B カウント数)　(式 2)

〔乱数の求め方〕

double 型の変数 x に 0 以上 a 以下の実数を乱数で代入するコードは次の通りである．

```
import java.util.*; // キーボード入力の「import java.io.*;」と同様にコードの最初に記載
double x = new Random().nextDouble() * a;
```

〔π の表記〕

円周率π（3.141592…）はコード上では Math.PI と表現する．

〔正弦関数〕

double 型の変数 s に x ラジアンの正弦を代入するコードは次の通りである．

```
double s = Math.sin(x);
```

下の実行例にならい，面積の推定値を画面に出力するコードを記述しなさい．但し，ファイル名を Code03.java として保存し，コンパイルと実行ができることとする．

〔実行例〕

正弦の面積を求めます
面積はおおよそ 2.0001577947610136 です

**問題４　交通シミュレーション**

**設問１**　直線の道路を走る車の交通を，１次元配列を用いてシミュレーションする．各車は左から右へ一方向に進み，前方に車がいないとき前進し，車がいるとき停止する．道路は長さ 64 の一次元配列 road〔int[]型〕で表現し，車の存在は各配列要素の値を 1 とすることで表現する．車がいない配列要素は 0 である．図２に１次元配列 road のイメージを示す．

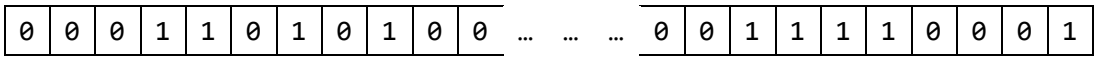


図２　道路と車の１次元配列 road による表現イメージ

次に，車の前進と停止は，各配列要素を表１の規則で更新することで表現する．添え字 i の配列要素 road[i]の次の更新は，後方 road[i-1]，自身 road[i]と前方 road[i+1]の車の有無より決まる．例えば，左から 4 番目の規則では，後方 0 自身 1 前方 1 であれば次の更新では自身は 1 になる．この更新規則は，配列上の各車が前方の配列要素に他の車が存在しないとき１つ前進し，前方に他の車がいるときその場に停止する動作を表現する．

表１　更新規則

| 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

ここで，両端の配列要素（road[0]と road[63]）には，周期的境界条件（配列の末端は先頭に繋がる）を用いる．すなわち，road[0]の後方は road[63]であり，road[63]の前方は road[0]である．

続いて，配列の更新の手順について説明する．配列 road の各配列要素を表１に従い更新するとき，同サイズの別配列 road_update（int[]型）を用いて行う．配列 road の各配列要素において，次の更新の値を一旦配列 road_update へ保存する．具体的には，road[i-1]，road[i]と road[i+1]に表１を適用した結果を road_update[i]へ保存する．すべての添え字 i について終了したら，配列 road_update のすべての配列要素の値を，配列 road の同じ位置の配列要素へ代入し戻す，この一連の配列の更新（以後，ターンという）を無限に繰り返す．シミュレーションの速度を調整するために，1 ターンを終えるたびに次のコードを実行して 200ms の一時停止を行うこととする．

```
try{
  Thread.sleep(200);
}catch(Exception e){}
```

配列 road の初期値について説明する．配列 road の各配列要素の初期値は車の出現確率 rate (%)〔int 型 0~100〕により設定する．すなわち添え字 i の配列要素の初期値は次のように設定する．他の配列要素も同様に行う．車の出現確率 rate はキーボード入力によりシミュレーションの起動時に入力することとする．

```
road[i] = (int)(Math.random()*100) < rate ? 1 : 0;
```

最後に配列 road の画面出力について説明する．配列 road の画面出力はターン毎に行う．各配列要素は 1 または 0 をもち，道路と車をより視覚的に表現するため 0 を'_'として，1 を'O'（大文字のオー）として横 1 行に出力を行う．その後，改行'¥n'は行わずに，復帰'¥r'を用いて，同行の行頭にカーソルを戻し，次のターンの画面出力は 1 つ前のターンの画面出力を上書きしながら行うこととする．

下の実行例にならい，交通シミュレーションを画面に出力するコードを記述しなさい．但し，ファイル名を Code04.java として保存し，コンパイルと実行ができることとする．

〔実行例〕

```
交通シミュレーションを行います
車の出現確率（0~100）を入力してください
>37
_O_O__O____O_O__O_O_O____O_O_____O_O__O___OOOO_O_O_____O_O_O_OO
```

※実行すると上の O は右方向へ 1 マスずつ進んでいるように表示される．

※JAVA プログラミング 1 のホームページにある動作例の動画も併せてご覧ください．

**設問２**　上記で作成した交通シミュレーションに信号機の処理を加えることを考える．信号は青（false）または赤（true）のいずれかとし，変数 trafficlight〔boolean 型〕により保持する．信号機の位置は配列 road の添え字 32 の位置とし，変数 trafficlight_pos〔int 型〕に代入しておく．配列の更新を 16 ターンを行う毎に変数 trafficlight の true（赤）と false（青）を反転する．

信号機が赤のときの車の動作は，配列 road において信号機の位置（添え字 32）とその前方（添え字 33）の配列要素を更新するとき，表２と表３の更新規則をそれぞれ用いることで表現できる．信号機が青の時は表１と同じ．

表２　信号機が赤のときの更新規則（信号機の位置（添え字 32）の配列要素の更新）

| 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

表３　信号機が赤のときの更新規則（信号機の前方位置（添え字 33）の配列要素の更新）

| 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

下の実行例にならい，信号機を考慮した交通シミュレーションを画面に出力するコードを記述しなさい．但し，ファイル名を Code04.java として保存し，コンパイルと実行ができることとする．なお，設問 2 は設問 1 の内容を含むため，設問 1 と同名のファイル名で保存し，課題４からは１つのソースファイルを提出すること．

〔実行例〕

```
交通シミュレーションを行います
車の出現確率（0~100）を入力してください
>37
__________0_0_0__0_0_0_0_000000000__________0_0_0_0_0_0_0_0_______
```

※実行すると上の 0 は右方向へ１マスずつ進んでいるように表示される．

※信号機のある位置 0 で，車が信号停止したり，再発進したりする状況がみられる．

※JAVA プログラミング１のホームページにある動作例の動画も併せてご覧ください．

**問題５　海戦ゲーム**

縦 5 マス×横 5 マスの海域に潜む長さ 3 の潜水艦を探し当てるゲームである．海域は 5 行 5 列の 2 次元配列 map〔int[][]型〕で表現する．各配列要素がとる整数とその意味は表４の通りとし，初期値は 0 である．

表４　2 次元配列 map の配列要素の意味

| 配列要素の整数 | 意味 | 画面出力の記号 |
|---|---|---|
| 0 | 未探索 | ? |
| 1 | 潜水艦にヒット | * |
| 2 | ミス | . |

潜水艦が潜む位置は 5 行 5 列の２次元配列 shipmap〔int[][]型〕で表現する．各配列要素がとる整数とその意味は表５の通りである．潜水艦の長さは 3 であり，縦または横（斜めの配置はなし）に 3 つ連続する 1 により潜水艦を表現する．

表５　2 次元配列 shipmap の配列要素の意味

| 配列要素の整数 | 意味 |
|---|---|
| 0 | なし |
| 1 | 潜水艦が潜んでいる |

ゲーム開始時に一度だけ潜水艦の位置を乱数により次の手順で決定する．

１．縦 5 マス×横 5 マス（海域）から 1 マスを乱数（使い方は問題２参照）により選ぶ．

２．選ばれたマスを起点とし長さ 3 の潜水艦が配置可能な上/下/左/右方向の候補を絞る．

３．配置可能な候補から 1 つを乱数（使い方は問題２参照）により選ぶ．

４．決定した配置を 2 次元配列 shipmap の該当する配列要素に 1 を代入して記録する．

次に，ゲーム進行と配列 map の更新について述べる．海域と潜水艦の配置が完了したら，表４にしたがい海域を該当する記号で画面出力を行う．続いて，キーボード入力により探索する海域の x 座標〔int 型 0~4〕と y 座標〔int 型 0~4〕を入力する．入力された座標(x, y)が未探索の場合〔map[y][x] == 0〕には，表６の処理を実行し，入力された座標(x, y)が探索済みの場合〔map[y][x] != 0〕には，「どうやら調査済みのようだ」を画面出力する．続いて，潜水艦の位置がすべて発見されたか確認を行い，すべて発見されていれば「発見できました」を画面出力してゲームを終了し，そうでなければ座標の入力に戻り繰り返す．

表６　配列 map の更新処理

| 入力された座標(x,y)に | 画面表示メッセージ | 配列 map への更新 |
|---|---|---|
| 潜水艦あり | どっか～ん、命中！ | map[y][x] = 1; |
| 潜水艦なし + 8 近傍にあり | ぴっぴっ、近いようだ | map[y][x] = 2; |
| 潜水艦なし + 8 近傍になし | 近くにはないようだ | map[y][x] = 2; |

下の実行例にならい，海戦ゲームを実行するコードを記述しなさい．但し，ファイル名を Code05.java として保存し，コンパイルと実行ができることとする．

〔実行例〕
```
海戦ゲームを始めます
長さ 3 の潜水艦 1 台の位置を探し当てましょう
?????
?????
?????
?????
?????
x 座標の値を入力してください>1
y 座標の値を入力してください>1
近くにはないようだ
?????
?.???
?????
?????
?????
x 座標の値を入力してください>3
y 座標の値を入力してください>1
どっか〜ん、命中！
?????
?.?*?
?????
?????
?????
x 座標の値を入力してください>3
y 座標の値を入力してください>0
ぴっぴっ、近いようだ
???.?
?.?*?
?????
?????
?????
x 座標の値を入力してください>3
y 座標の値を入力してください>2
どっか〜ん、命中！
???.?
?.?*?
???*?
?????
?????
x 座標の値を入力してください>3
y 座標の値を入力してください>3
どっか〜ん、命中！
???.?
?.?*?
???*?
???*?
?????
発見できました♪
```