目次

★3　クラス変数とクラスメソッド（承前）

★3.4　Java API に現れるクラス変数，クラスメソッド（承前）

String クラス (p.30)

前回登場した Math クラスと同じく，String も java.lang パッケージのクラスである (☆ 1)．文字列を表す (☆ 2)．String クラスについてはいろいろ知っておくとよいことがあるが，この授業では簡単にしか触れない．今のところは以下の説明と例を理解しておこう．

- Java では文字列を String クラスのオブジェクトとして扱う．
- " と " で囲まれた文字列は文字列定数であり，new 演算子で生成しなくとも使える．
- （文字列）+（文字列）という演算を行うと，二つの文字列を連結した文字列が得られる．

☆1) java.lang パッケージは自動的に import される．前回資料の Math クラスの説明参照．

☆2) C 言語では文字列を char 型の配列に格納していたが，Java では扱い方がずいぶん異なっている．

文字列の扱い

```
System.out.println("HOGE!");
System.out.println("What is \"HOGE\" ?");

String s = "hoge", s1 = "fuga";
System.out.println(s);
System.out.println(s + s1);  // 文字列に対する+演算は文字列の連結

int x = 10, y = 26;
System.out.println(x);  // 文字列に変換して出力
System.out.println("x = " + x); // +演算の対象のうち一方でも文字列
                               // なら文字列同士の連結
System.out.println(x + y);      // この場合は？
System.out.println(x + "" + y); // この場合は？

// String クラスのインスタンスメソッドの使用例
System.out.println(s + "は" + s.length() + "文字です");

if(s.equals("Hoge")){
   System.out.println(s + "は Hoge に等しいです");
}else{
   System.out.println(s + "は Hoge に等しくありません");
}
```

左の実行結果 (*)

```
HOGE!
What is "HOGE" ?

hoge
hogefuga

10
x = 10
36
1026

hoge は 4 文字です

hoge は Hoge に等しくありません
```

(*) この実行結果は，左のソースと対応づけやすいように適当に改行を挿入してある．

［発展］文字列同士の比較には，上記のように String クラスのインスタンスメソッド equals を用いる．==で比較してもうまくいかないので注意（p.37 傍注 31) 参照)．

★4 処理の流れ

★4.1 for 文による繰返し (p.31)

C 言語同様に for 文が使える: `for(` 初期化式 `;` 繰返し条件式 `;` ループの更新式 `)` 繰り返す文

- 「条件式」には boolean 型の式を書く（詳細は後の節を参照）.
- 「初期化式」に変数宣言を書くこともできる → Q.2

for 文を使った例 T41.java

```
 1  public class T41 {
 2     public static void main(String[] args){
 3        TurtleFrame f = new TurtleFrame();
 4        Turtle m = new Turtle();
 5        f.add(m);
 6        int i;
 7        for(i = 0; i < 5; i++){
 8           m.fd(100);
 9           m.rt(72);
10        }
11     }
12  }
```

```
(1) i ← 0

(2) i < 5 ?   yes
(4) 実行 ( このとき i は 0 )
(3) i++ ( i ← 1 )

(2) i < 5 ?   yes
(4) 実行 ( このとき i は 1 )
(3) i++ ( i ← 2 )

          :

(2) i < 5 ?   yes
(4) 実行 ( このとき i は 4 )
(3) i++ ( i ← 5 )

(2) i < 5 ?   no
```

Q1. T41 を実行中，i=2 のときに 9 行目の処理を行ったあと，かめはどこにいる？ 向きは？

Q2. 以下の二つはどう違うだろう？

for ブロックの外で i を宣言

```
int i;
for(i = 0; i < 3; i++){
    System.out.println("i = " + i);
}
System.out.println(
  "for 文終了後の i の値は " + i);
```

for 文中で i を宣言

```
for(int i = 0; i < 3; i++){
    System.out.println("i = " + i);
}
System.out.println(
  "for 文終了後の i の値は " + i);
```

Q3. p.34 の練習問題 4.4 をやろう（以下を書きかえよう）.

P431.java

```
for(int k = 1; k <= 20; k++){
   m.rt(90); m.fd(10);
   m.lt(90); m.fd(k);
}
```

P432.java

```
for(int k = 1; k <= 20; k++){
   m.rt(90); m.fd(10);
   m.lt(90); m.fd(k);
}
```

★ 4.2 繰返しの繰返し（ネスト） (p.35)

for 文の「繰り返す文」の所にまた for 文を書けば，繰返しを繰り返すことができる（これを繰返しの**ネスト**という）．for 文に限らず，様々な繰返し文でネストできる（当然，for 文の中に while 文を入れたりしても構わない）．また，ネストは繰返しに限らない（if 文のネストなど）．

Q4. p.35 の T43.java の 9 行目と 10 行目の間に `System.out.println("i = " + i + ", j = " + j);` という文を挿入すると，どんな出力が得られますか．

★ 4.3 while 文による繰返し (p.36)

while 文もあるよ． `while(` 繰返し条件式 `)` 繰り返す文

Q5. p.34 の Sum41.java と同じものを while 文で書いてみよう．

Sum41.java の一部

```
int sum = 0;
for(int i = 1; i <= 10; i++)
  sum += i;
System.out.println("The sum is " + sum);
```

Q6. 上記をもとに while 文の条件を書きかえて「和がはじめて 20 を超えたときの和の値を表示する」ようにしてみよう（ヒント：和が 20 以下の間は繰り返す）．

Q7. 以下の二つはどう違うでしょう？

while

```
int i = 10;
while(i < 3){
  System.out.println("ほげ");
  i++;
}
```

do–while

```
int i = 10;
do{
  System.out.println("ほげ");
  i++;
}while(i < 3);
```

★ 4.4 boolean 型と論理演算子 (p.37)

C 言語では，論理値 (真 (true) と偽 (false)) を表すのに整数値を用い，「0 以外の数は真」を，「0 は偽」を表した．しかし，Java では論理値用の型として **boolean 型**が用意されている．boolean 型の値は true, false の 2 つのみである．

- C 言語と同様に，数値同士の比較などのために次のような演算子が使える．`>`，`<`，`>=`，`<=`，`==`，`!=` （詳細は p.37 参照）
- `==`と`!=`は，boolean 型や，参照型などでも使える．参照型の場合，両辺が同一のオブジェクトを指しているかどうかを調べることになる (☆ 3)．
- 文字列同士が等しいかどうか調べる場合は，String クラスのインスタンスメソッド `equals(String str)` を用いる（この講義資料の前半，p.37）．

C 言語同様に，論理積，論理和や否定の演算子も存在する．

`A && B`	A と B の論理積（A も B も true なら true，さもなくば false）
`A || B`	A と B の論理和（A,B の一方でも true なら true，さもなくば false）
`!A`	A の否定（A が true なら false，false なら true）

☆ 3) 例えば，
```
Turtle a = new Turtle();
Turtle b = new Turtle();
```
なら，`a == b` は `false`，
```
Turtle a = new Turtle();
Turtle b = a;
```
なら，`a == b` は `true`．

Q8. 右のソースを boolean 型を使って修正し，左の C プログラムと同じ動作をするものにしよう．

C 言語の場合
```
int x = 1;
/* x は 0 でないので↓は条件成立 */
if(x) printf("ほげ\n");
/* 1 は真だから↓は無限ループ */
while(1) printf("ふが\n");
```

Java ではこれはコンパイルエラー
```
int x = 1;
if(x) System.out.println("ほげ");

while(1) System.out.println("ふが");
```

Q9. `int` 型変数 `x` に 3 が代入されているとき，次の式の値はいくつか．

(a) `x > 0`　　(b) `x == 2`

Q10. 以下の条件を boolean 型の式で表しなさい．x と y の値はあらかじめ宣言された int 型変数 x,y に格納されていると仮定すればよい．

1. $x+3$ が 0 より大きくかつ x^2 が 100 未満である
2. x, y のいずれかが 1 である
3. x は 0 以上 10 未満である
4. $0 \leq x < 10$ または $20 \leq x < 30$
5. x は 3 以外の 3 の倍数である

★4.5 if 文 (p.38)

if 文の書き方，使い方は C 言語と同様．ただし，条件式は boolean 型．

Q11. 以下を実行するとどんな結果が得られますか．

if 文の例 T45.java（一部）

```
for(int i = 0; i < 5; i++){
  if(i == 3)
    m.setColor(java.awt.Color.red);
  m.fd(100);
  m.rt(72);
}
```

```
if( 条件式 ) 条件成立時だけ行う文
```

Q12. 以下を実行するとどんな結果が得られますか．

if–else の例

```
for(int i = 0; i < 9; i++){
  System.out.print(i);
  if(i % 3 == 1)
    System.out.println(" (^^)/");
  else
    System.out.println("");
}
```

```
if( 条件式 ) 条件成立時に行う文
else 条件不成立時に行う文
```

注 1: println("") は，改行だけする．引数なしでも同じ．
注 2: Q12 の例では，「条件成立時に行う文」も「条件不成立時に行う文」も一文なので {,} で囲んでいない．しかし，このようにしていると，文を追加する際に{,}も付け足すのを忘れてコンパイルエラーやバグを生みやすい．Q13 の例のように，一文しかなくても{,}で囲んでおく方がよいだろう．

Q13. 以下を実行するとどんな結果が得られますか．

else–if の例 （T47 改）

```
for(int i = 0; i < 12; i++){
  if(i % 4 == 0){
    m.setColor(java.awt.Color.red);
  }else if (i % 4 == 1) {
    m.setColor(java.awt.Color.green);
  }else{
    m.setColor(java.awt.Color.blue);
  }
  m.lt(30);
  m.fd(50);
}
```

```
if( 条件式 A ) 文 a
else if( 条件式 B ) 文 b
else if( 条件式 C ) 文 c
...
else 文 n
```

宿題？

次回は教科書第 4 章の残りと第 5 章の内容を説明します．あらかじめ読んでおくこと．例題のプログラムを作成し実行してみておくこと．