錬成問題

▪ **abstract** 付きで宣言されたクラスは“ (1) クラス”であり、**abstract** 付きで宣言されたメソッドは“ (2) メソッド”である。上位クラスの (2) メソッドをオーバライドしてメソッド本体を定義することを“メソッドを (3) する”という。

▪ 正しい記述には○を、誤った記述には×を記入せよ。

(4) … (1) クラスは、(2) メソッドを必ず 1 個以上もたなければならない。

(5) … (1) クラスは、**final** クラスとして宣言することができる。

(6) … (2) メソッドを 1 個でももつクラスは、(1) クラスでなければならない。

(7) … (2) メソッドには、公開アクセス性を与える必要がある。

(8) … (1) クラスは、コンストラクタをもつことはできない。

(9) … (1) クラス型のクラス型変数を作ることはできる。

(10) … (1) クラス型のインスタンスを生成することはできる。

(11) … (1) クラス型のクラス型変数は、下位クラス型のインスタンスを参照できる。

(12) … メソッドの中では、同一クラスに所属する (2) メソッドを呼び出すことができる。

(13) … 上位クラスの非 (2) メソッドを、(2) メソッドとしてオーバライドすることはできない。

(14) … 上位クラスの (2) メソッドを、非 (2) メソッドとしてオーバライドすることはできない。

(15) … (1) クラスから派生したサブクラスでは、必ず (2) メソッドをオーバライドしてメソッド本体を定義しなければならない。

(16) … **java.lang.Object** クラスでは、**public String** toString() が (2) メソッドとして定義されている。

(17) … 自作のクラスで **public String** toString() を (2) メソッドとして定義することができる。

▪ 文書化コメントは、(18) と (19) とで囲んで記述する注釈であり、ホームページ記述言語として知られている (20) のタグを埋め込むことができる。文書化コメントをもとにドキュメントを生成するツールが (21) である。ソースプログラムの中に **import** 宣言とクラス宣言とがある場合、クラスに対する文書化コメントは、(22) に位置していなければならない。

▶ (22) の選択肢：(a) **import** 宣言の前　(b) **import** 宣言とクラス宣言の間

▪ 以下に示すのは、動物を表すクラス群と、それを利用するプログラムである。

```
//--- 動物クラス ---//
  (23)  class Animal {
   private String name;                    // 名前

   Animal(String name) { this.name = name; }

    (24)  void bark();                     // 吠える
    (25)  (26)  String toString();         // 文字列表現を返す

   String getName() { return name; }

   void introduce() {
      System.out.print(  (27)  + "だ");
       (28) ;
   }
}
```

```
//--- 犬クラス ---//
class Dog  (29)  Animal {
   private String type;                    // 犬種

   Dog(String name, String type) {
       (30) (name);  this.type = type;
   }

    (31)  bark() { System.out.println("ワンワン!!"); }

    (32)  (33)  toString() { return type + "の" + getName(); }
}
```

```
//--- 猫クラス ---//
class Cat  (34)  Animal {
   private int age;                        // 年齢

   Cat(String name, int age) { super(name);  this.age = age; }

    (35)  bark() { System.out.println("ニャ～ン!!"); }

    (36)  (37)  toString() { return age + "歳の" + getName(); }
}
```

```
//--- 動物クラスのテスト ---//
class AnimalTester {

    (38)  static void main(String[] args) {
       (39)  a = {
          new Dog("タロー", "柴犬"),     // 犬
          new Cat("マイケル", 7),        // 猫
          new Dog("ハチ公", "秋田犬"),   // 犬
       };

       for (Animal k : a) {
          (40) ;
          System.out.println();
       }
    }
}
```

```
柴犬のタローだワンワン!!
7歳のマイケルだニャ～ン!!
秋田犬のハチ公だワンワン!!
```