錬成問題

▪ インタフェース内で宣言するメンバには、既定の属性が与えられる。メソッドの属性は **abstract** かつ［ (1) ］で、フィールドの属性は［ (1) ］かつ［ (2) ］かつ［ (3) ］である。なお、インタフェースそのものには既定の属性として［ (1) ］が与え［ (4) ］。

▶［ (4) ］の選択肢：(a)られる　(b)られない

▪ 正しい記述には○を、誤った記述には×を記入せよ。

［ (5) ］…インタフェース内で宣言するメソッドは、必ず本体を定義しなければならない。

［ (6) ］…インタフェースの名前の末尾は～ able でなければならない。

［ (7) ］…単一インタフェースからインタフェースを派生することができる。

［ (8) ］…複数インタフェースからインタフェースを派生することができる。

［ (9) ］…単一クラスからの派生と単一インタフェースの実装を同時に行うことができる。

［ (10) ］…複数インタフェースの実装を行うことができる。

［ (11) ］…インタフェース型の変数を作ることができる。

［ (12) ］…インタフェース型のインスタンスを生成することができる。

［ (13) ］…インタフェース型の変数は、それを実装したクラスのインスタンスを参照することができる。

［ (14) ］…インタフェース型の変数に **instanceof** 演算子を適用することはできない。

［ (15) ］…インタフェースを実装する際は、キーワード **extends** を用いる。

［ (16) ］…インタフェースを実装するクラスでは、実装するメソッドに対して **public** 属性を与えてもよいし、与えなくてもよい。

［ (17) ］…インタフェースを実装するクラスで、すべてのメソッドを実装しなければ、そのクラスは抽象クラスとして宣言しなければならない。

［ (18) ］…スーパークラスのメソッドをオーバライドしたメソッド呼出しの解決が動的結合で行われるのに対し、インタフェースのメソッドをオーバライドしたメソッド呼出しの解決は静的結合で行われる。

［ (19) ］…インタフェースを実装するクラスの中では、インタフェース内で宣言されたフィールドを、必ず“インタフェース名．フィールド名”の形式でアクセスしなければならない。

［ (20) ］…データ隠蔽を実現するために、インタフェースで定義するフィールドは、原則として非公開のアクセス性を与えるべきである。

［ (21) ］…インタフェース型は参照型の一種である。

▪ 以下に示すのは、着色可能なウェアラブルロボットクラスとそれをテストするプログラムである。

```
// ウェアラブルインタフェース
[ (22) ] Wearable {
   void putOn();          // 着る
   void putOff();         // 脱ぐ
}
```

```
// 着色インタフェース
[ (23) ] Colorable {
   int RED = 1;           // 赤
   int GREEN = 2;         // 緑
   int BLUE = 3;          // 青
   void changeColor(int color);     // 色変更
}
```

```
//--- ウェアラブルコンピュータ クラス ---//
class WearableComputer [ (24) ] Wearable {
   private String name;          // 名前

   [ (25) ](String name) { this.name = name; }

   [ (26) ] void putOn()  { System.out.println(name + " ON!!"); }
   [ (27) ] void putOff() { System.out.println(name + " OFF!!"); }
}
```

```
//--- 着色可能なウェアラブルロボット クラス ---//
class WearableRobot [ (28) ] Wearable, [ (29) ] {
   private int color;              // 色

   WearableRobot(int color) { [ (30) ](color); }

   [ (31) ] void changeColor(int color) { [ (32) ].color = color; }

   [ (33) ] String toString[ (34) ]  {
      [ (35) ] (color) {
         [ (36) ] RED   : return "赤ロボット";
         [ (37) ] GREEN : return "緑ロボット";
         [ (38) ] BLUE  : return "青ロボット";
      }
      return "ロボット";
   }

   [ (39) ] void putOn()  { System.out.println( [ (40) ] + " 装着!!"); }
   [ (41) ] void putOff() { System.out.println( [ (42) ] + " 解除!!"); }
}
```

```
//--- テスト ---//
class Test {

   public static void main(String[] args) {
      [ (43) ][] w = {
         new WearableComputer("HAL"),
         new WearableRobot(Colorable.RED),
         new WearableRobot(Colorable.GREEN),
      };

      [ (44) ] (Wearable k [ (45) ] w) {
         [ (46) ].putOn();
         [ (47) ].putOff();
         System.out.println();
      }
   }
}
```

```
HAL ON!!
HAL OFF!!

赤ロボット 装着!!
赤ロボット 解除!!

緑ロボット 装着!!
緑ロボット 解除!!
```