

JavaプログラミングⅡ

3回目 クラスの機能（１）アクセス制限、オーバーロード 課題

確認○×問題

次の各文は正しいか誤っているか答えなさい。

- (1) クラスの private メンバは、そのクラスからのみアクセス可能なメンバである
- (2) 一般に、クラスのフィールドはどこからでもアクセスできるように public メンバで宣言すべきである
- (3) クラスは private メンバと public メンバを同時に持つことはできない
- (4) public メンバはどこからでもアクセスできるメンバであるため、宣言する際には注意が必要である
- (5) クラスのカプセル化は、クラスの内部のデータを外部から隠ぺいして、データの保護や保守に大きく貢献する。
- (6) クラスのカプセル化は、ポリモーフィズムと呼ばれる。
- (7) オーバーロードとは、戻り値の型が異なる同じ名前と仮引数のメソッドを複数宣言することである。
- (8) オーバーロードされたメソッドを呼び出すとき、与えられた実引数の違いにより実際に呼び出されるメソッドが決定される。

難易度★★★

課題１ 次のクラス **Score** は得点を管理するクラスです。変数 **point** には0以上100以下の値のみ代入されるようにしたいです。空欄を埋めてクラスをカプセル化してコードを完成させなさい。

```
class Score
{
     int point; // 得点

    // 得点設定メソッド
     void setPoint(int p){ // 値を検査して0以上100以下であれば代入
        
    }
    // 現設定値取得メソッド
     int getPoint(){ // 現在の値を戻り値として返す
        
    }
}

class Assignment3_1
{
    public static void main(String[] args){
        Score sc=new Score();
        sc.setPoint(90); // 90 点を設定してみる
        System.out.println(sc.getPoint()); // 設定値の表示
        sc.setPoint(-10); // -10 点を設定してみる
        System.out.println(sc.getPoint()); // 設定値の表示
    }
}
```

難易度★★★

課題2 次のクラス **EvenInt** は偶数を管理するクラスです。変数 **num** には偶数のみ代入されるようにしたいです。空欄を埋めてクラスをカプセル化してコードを完成させなさい。

```
class EvenInt
{
     int num; // 偶数値

    // 偶数設定メソッド
     void setInt(int i){ // 値を検査して偶数であれば代入
        
    }
    // 現設定値取得メソッド
     int getInt(){ // 現在の値を戻り値として返す
        
    }
}

class Assignment3_2
{
    public static void main(String[] args){
        EvenInt ei=new EvenInt();
        ei.setInt(12); //偶数の設定を試みる
        System.out.println(ei.getInt()); // 結果の表示
        ei.setInt(15); //奇数の設定を試みる
        System.out.println(ei.getInt()); // 結果の表示
    }
}
```

難易度★★★

課題3 メソッドのオーバーロードの機能を用いて、次に示すクラス **OverLoadTest** 内のメソッドを一つにまとめなさい。

```
class OverLoadTest
{
    private int i;
    private double d;
    private String s;

    public void setInt(int n){
        i=n;
    }
    public void setDouble(double n){
        d=n;
    }
    public void setString(String n){
        s=n;
    }

    public void dispValue(){
        System.out.println("int="+i+", double="+d+", String="+s);
    }
}
```

```
class Assignment3_3
{
    public static void main(String[] args){
        OverLoadTest olt=new OverLoadTest();

        //olt.setInt(10);
        //olt.setDouble(12.5);
        //olt.setString("Hello");
        //olt.dispValue();

        // 次のようにアクセスしたい！！
        olt.setValue(10);
        olt.setValue(12.5);
        olt.setValue("OverLoad");
        olt.dispValue();
    }
}
```

難易度★★★

課題4 個人データを管理するクラス **Person** を次のように宣言しなさい。このとき、メソッド内では入力値が正しいかどうかのチェックを行った後、**private** フィールドに値を設定するようにすること。

〔private フィールド〕

```
String name; // 名前
int age;      // 年齢（0 以上）
char gender; // 性別（F or M）
```

〔public メソッド〕

```
void setName(String); // 名前を設定
String getName();     // 名前を取得
void setAge(int);      // 年齢を設定
int getAge();          // 年齢を取得
void setGender(char); // 性別を設定
char getGender();      // 性別を取得
```

難易度★★★

課題5 クラス **Person** の動作をチェックするために、メイン関数からクラス **Person** のオブジェクトを生成して、オブジェクトに値を設定するコードを記述しなさい。また、設定した値を取得して出力しなさい。

難易度★★★

課題6 メソッドのオーバーロードの機能を用いて、クラス **Person** のメソッド **setName(Sting)**と **setAge(int)**、**setGender(char)**を同一のメソッド **setData()**で実現しなさい。