小テスト

準備 プログラムを作る前に、以下の操作をしてファイルの準備をしておくこと。

1. GitHub から自分のリポジトリを clone しておく（既に今日の授業で実行済みの場合は不要）
   ```
   $  mygitclone4d  「自分の GitHub ユーザ名」
   $  cd prog4d-(ユーザ名)
   $  ./myconf
   ```
2. 今回の小テスト用のフォルダを作って移動する
   ```
   $ cd ~/prog4d-(ユーザ名)
   $ mkdir test110
   $ cd test110
   ```
3. テキストエディタでプログラムを開き、**先頭行に Java のコメントとして自分の番号と名前を書く**
   ```
   $ gedit test110.java &
   ```

【問 1】 次のような整数を扱うクラス Base を考える。

```
//「1_10_Base.java」にあるクラス Base と同じ
class Base {
    protected int num;
    public Base() {
        num = 0;
    }
    public Base(int n) {
        num = n;
    }
    public void setNum(int n) {
        num = n;
    }
    public int getNum() {
        return num;
    }
    public void showNum() {
        System.out.printf("num: %d\n", num);
    }
}
```

クラス Base を継承して、次のようなフィールドとメソッドを追加し、クラス Positive を作成して、動作を確認してください。

- フィールドとして、num に前回代入されていた整数（previous）を持つ。(int 型)
- 次のような 1 個のコンストラクタを持つ。

```
public Positive()
  //引数付きの super() を使って、フィールド num を 10 にする
  //フィールド previous に-1 を代入する
```

- 次のようなメソッド setNum をオーバーライドする。

```
public void setNum(int n)
  //引数 n が正（0 は含まない）かどうかを判別し、
  //・真の場合、num を previous に代入し、n をフィールド num に代入して、代入されたことを出力する
  //・偽の場合、代入されなかったことを出力する
  //出力するメッセージは実行結果を参照
```

- 次のようなメソッド showNum をオーバーライドする。(出力処理の順番に注意)

```
public void showNum()
  //フィールド previous を出力する（出力の様子は実行結果を参照）
  //super を使って、スーパークラスのメソッド showNum を呼び出す
```

main の処理とその実行結果は以下のようになります。

```
[main を持ったクラス]
class Pd10test {
    public static void main(String[] args) {
        //インスタンスを作成する
        Base b1 = new Base();
        Base b2 = new Positive();
        //showNum の動作確認
        System.out.println("--- b1 ---");
        b1.showNum();
        System.out.println("--- b2 ---");
        b2.showNum();
        //setNum の動作確認
        System.out.println("--- 正の値の場合 ---");
        b2.setNum(7);
        b2.showNum();
        System.out.println("--- 負の値の場合 ---");
        b2.setNum(-3);
        b2.showNum();
    }
}
```

```
[実行結果]
  --- b1 ---
  num: 0
  --- b2 ---
  前回の値: -1
  num: 10
  --- 正の値の場合 ---
  値を代入しました
  前回の値: 10
  num: 7
  --- 負の値の場合 ---
  値を代入しません
  前回の値: 10
  num: 7
```

小テストの注意点 (20 点)

- 他人の力は借りずに、自分だけでプログラムを作成する。つまり、**通常の定期試験と同様**。

小テスト中に参照できるもの

- 教科書, 配付資料
- 自分のホームディレクトリ（ホームフォルダ）以下に保存されているファイル

* **上記以外の情報を参照することは不正行為とする**
(例：USB で接続された機器に保存されているファイルの参照, ネットワークを介した情報の参照など)

答案の提出

1. **提出する全てのファイルの先頭行に、C のコメントとして自分の番号と名前を書く**
2. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
$ git commit -m "小テスト 10 提出"
$ git push origin master
```

3. 提出が完了しているかを確認したい人は声をかけて下さい。(その場で教員側の画面で確認します)

小テストの模範解答

```
/* 自分の番号と名前をここに書く */
class Base {
    protected int num;
    public Base() {
        num = 0;
    }
    public Base(int n) {
        num = n;
    }
    public void setNum(int n) {
        num = n;
    }
    public int getNum() {
        return num;
    }
    public void showNum() {
        System.out.printf("num: %d\n", num);
    }
}

//正の値を保持するクラス
class Positive extends Base {
    int previous;  //1 つ前の整数を覚えておく
    //コンストラクタ
    public Positive() {
        super(10);
        previous = -1;
    }
    //正のみを代入する num のセッタ
    //setNum をオーバライドする
    public void setNum(int n) {
        //正かどうかを調べて値を代入する
        if(n > 0) {
            previous = num;
            num = n;
            System.out.println("値を代入しました");
        } else {
            System.out.println("値を代入しません");
        }
    }
    //showNum をオーバライドする
    public void showNum() {
        System.out.printf("前回の値: %d\n", previous);
        super.showNum();
    }
}

class Pd10test {
    public static void main(String[] args) {
        //インスタンスを作成する
        Base b1 = new Base();
```

```
        Base b2 = new Positive();
        //showNum の動作確認
        System.out.println("--- b1 ---");
        b1.showNum();
        System.out.println("--- b2 ---");
        b2.showNum();
        //setNum の動作確認
        System.out.println("--- 正の値の場合 ---");
        b2.setNum(7);
        b2.showNum();
        System.out.println("--- 負の値の場合 ---");
        b2.setNum(-3);
        b2.showNum();
    }
}
```