プログラミングA 第6回・宿題

宿題の提出は Moodle で木曜日までに行ってください。各宿題ごとに提出ファイルを zip 等で一つのファイルにまとめて該当する Moodle の課題に提出しなさい。この資料や関係するコードをインターネットなどに公開することは著作権上、禁止されています。

1 宿題 1

以下を満たす Factory クラス、CarFactory クラス、BikeFactory クラスをそれぞれ定義し、FactoryTest から利用できることを確認の上、Factory.java、CarFactory.java、BikeFactory.java の全ての内容を回答欄に貼り付けなさい。

- Factory クラスがスーパークラス、CarFactory クラスおよび BikeFactory クラスがサブクラスである。
  - Factory クラスは String 型のインスタンスフィールド name を持つ。
  - Factory クラスは String を引数にとり、name を初期化するコンストラクタを持つ。
  - Factory クラスは、引数を取らず、String を戻り値とする produce メソッドを持つ。ただし Factory インスタンスに対して produce を呼び出すと null が返る。
- CarFactory クラスおよび BikeFactory クラスは、Factory クラスから name を継承する。
  - CarFactory クラスは、引数を取らず、Factory クラスのコンストラクタを利用して name を” 車工場” として初期化するコンストラクタを持つ。
  - BikeFactory クラスは、引数を取らず、Factory クラスのコンストラクタを利用して name を” 自転車工場” として初期化するコンストラクタを持つ。
- CarFactory クラスおよび BikeFactory クラスは produce メソッドをオーバライドする。
  - CarFactory クラスの produce を呼び出すと、” 車” が返る。
  - BikeFactory クラスの produce を呼び出すと、” 自転車” が返る。

なお、FactoryTest を実行すると標準出力に以下が出力される。

```
自転車工場の自転車
車工場の車
車工場の車
```

参考資料：FactoryTest.java

```
class FactoryTest {
  public static void main(String[] args) {
    Factory[] factories = new Factory[] {
      new BikeFactory(),
      new CarFactory(),
      new CarFactory()
    };
    for(int i = 0; i < factories.length; i++) {
      System.out.println(factories[i].name + "の" + factories[i].produce());
    }
  }
```

```
}
```

2　宿題 2

X-Y 座標系における描画プログラム開発のため、以下を満たすクラス Point、クラス Line、および、アブストラクトクラス Figure を作成し、全てを回答欄に貼り付けよ。

- クラス Point は点を表現する。以下を含む。
  - コンストラクタは引数として、点の X 座標, Y 座標における位置を表す int 型の x, y をとる。
  - 引数がなく戻り値の型が String のインスタンスメソッド draw() を実行すると、戻り値として位置の文字列表現（例えば ”(5,8)”）を得る。
  - int 型の引数 mx, my をとり戻り値の型が void のインスタンスメソッド move() を実行すると、位置を X 座標, Y 座標のそれぞれについて mx, my だけ移動させる。例えば当初の位置が (5,8) であれば (5+mx, 8+my) へ移動させる。
- クラス Line は、Point を用いて 2 点間の直線を表現する。以下を含む。
  - コンストラクタは引数として、開始点の Point インスタンス p1 と終了点の Point インスタンス p2 をとる。
  - 引数がなく戻り値の型が String のインスタンスメソッド draw() を実行すると、戻り値として開始点の位置と終了点の位置を連結した文字列表現（例えば ”((5,8),(100,200))”）を得る。
  - int 型の引数 mx, my をとり戻り値の型が void のインスタンスメソッド move() を実行すると、開始点と終了点の位置を X 座標, Y 座標のそれぞれについて mx, my だけ移動させる。例えば当初 ((5,8), (10,20)) であれば ((5+mx,8+my),(10+mx,20+my) へ移動させる。
- 抽象クラス Figure は、Point と Line のスーパクラスであり、それらに共通する手続きをまとめあげる。以下を含む。
  - 引数がなく戻り値の型が String のメソッド draw() を持つ。
  - int 型の引数 mx, my をとり戻り値の型が void のメソッド move() を持つ。

以上に基づき、FigureClient を実行すると以下に示す期待結果を標準出力に得る。ここで if 文や switch・case 文を使用せずに Point, Line, Figure のプログラムを作成し、すべてを回答欄に記述しなさい。それぞれの実装にあたり上記にないメソッドやフィールド等を加えて構わない。

出力結果：

```
(100,200)
((105,208)(110,220))
```

参考資料：FigureClient.java

```
public class FigureClient {
   public static void main(String[] args) {
      Figure[] figures = new Figure[2];
      figures[0] = new Point(0,0);
      figures[1] = new Line(new Point(5,8), new Point(10,20));
```

```
    for(int i = 0; i < figures.length; i++) {
      figures[i].move(100,200);
      System.out.println(figures[i].draw());
    }
  }
}
```

3　宿題 3

以下を満たす Countable インタフェース、Text クラス、IntArray クラスをそれぞれ定義し、CountableTest から利用できることを確認の上、Countable.java、Text.java、IntArray.java の全ての内容を回答欄に貼り付けなさい。

- Countable インタフェースは、戻り値の型が int かつ引数をとらないメソッド count を宣言する。
- Text クラスおよび IntArray クラスは、Countable インタフェースを実装する。
- Text クラスのコンストラクタは引数に文字列（String インスタンス）をとり、同文字列を適当なフィールドから参照し続ける。ただし引数に null が与えられた場合は、空の文字列定数（つまり ”” ）を参照すること。
- Text クラスの count メソッドを呼び出すと、当該インスタンスのフィールドから参照している文字列の長さを返す。
- IntArray クラスのコンストラクタは引数に整数の配列（int の配列型）をとり、同配列を適当なフィールドから参照する。ただし引数に null が与えられた場合は、要素数 0 の int の配列（つまり new int[0] ）を参照すること。
- IntArray クラスの count メソッドを呼び出すと、当該インスタンスのフィールドから参照している整数の配列の要素数を返す。

なお、CountableTest を実行すると標準出力に以下が出力される。

```
26
6
3
```

参考資料：CountableTest.java

```
public class CountableTest {
  public static void main(String[] args) {
    Countable[] countables = new Countable[3];
    countables[0] = new Text("ABCDEFGHIJKLMNOPQRSTUVWXYZ");
    countables[1] = new Text("100503");
    countables[2] = new IntArray(new int[]{ 100, 50, 3 });
    for(int i = 0; i < countables.length; i++) {
      System.out.println(countables[i].count());
    }
  }
```

```
}
```