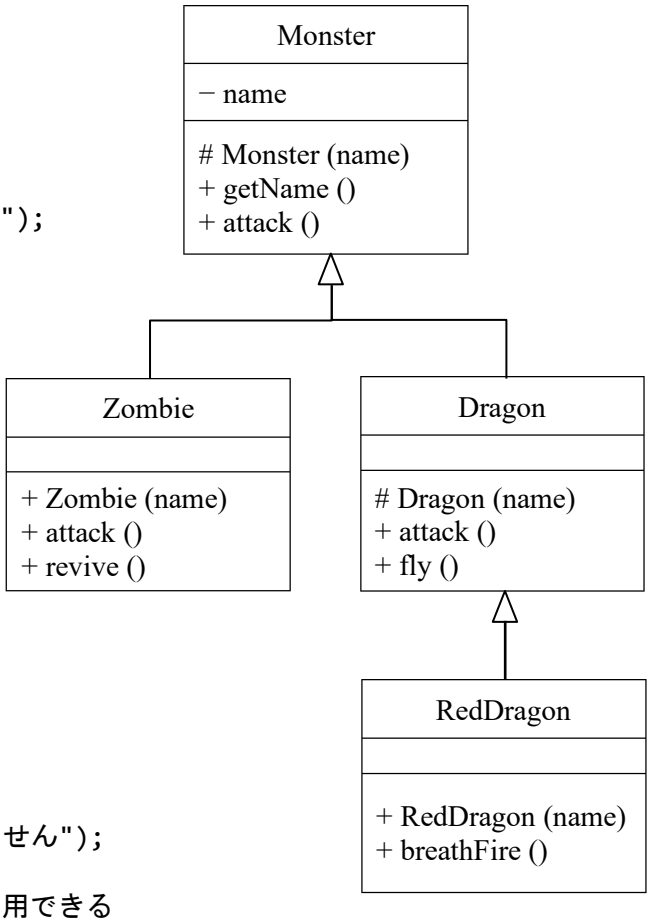
プログラミングⅡ　2022　第10回　演習課題　「多態性（ポリモーフィズム）」

1. 右の UML のクラス図で示したクラスを作成し，下記のプログラムの空欄を適切に埋めて動作を理解せよ。
各メソッドの処理内容は，インスタンスの名前とそのメソッドが実行されたことが表示される程度でよい。




```
public class Main {
   public static void main(String [] args) {

      Zombie z1 = new  空欄(1)  ("ゾンビ A");
      System.out.println(z1.getName());
      z1.revive();

      RedDragon d1 = new  空欄(2)  ("ドラゴン A");
      System.out.println(d1.getName());
      d1.breathFire();

      // 親クラス型の変数に代入できる
      Monster m1 = new Zombie("ゾンビ B");
      m1.attack();
      Monster m2 = d1;
      m2.attack();

      // 親クラス型の仮引数に渡すことができる
      attackTwice(z1);
      attackTwice(d1);

      // クラスのインスタンスかチェックできる
      if (m1  空欄(3)  Zombie)
         System.out.println("m1 は Zombie です");
      else
         System.out.println("m1 は Zombie ではありません");

      // 型チェック後ダウンキャストすればメソッドを利用できる
      if (  空欄(4)  Dragon) {
         Dragon d =  空欄(5)  m2;
         d.fly();
      } else {
         System.out.println("m2 は Dragon ではありません");
      }

      // 親クラス型の配列に子クラスのインスタンスを格納できる
       空欄(6)  [] a = new  空欄(7)  [4];
      a[0] = z1;  a[1] = d1;  a[2] = m1;  a[3] = m2;

      for (  空欄(8)  m : a) {
         m.attack();
         if (m  空欄(9)  ) {
             空欄(10)  d =  空欄(11)  m;
            d.breathFire();
         }
      }
   }

   // メソッドの引数は、宣言したクラスの子クラスのインスタンスも受け取ることができる
   public static void attackTwice(  空欄(12)  m) {
      m.attack();
      m.attack();
   }
}
```

2. 下記は第 7 回の 3.のプログラムを拡張したものである。プログラム中の※印の部分を適切に補って正しく動作するようにプログラムを完成させ，実行結果を示せ。

```
/* Main.java */
import java.util.Scanner;

public class Main {
   public static void main(String [] args) {

      // 10 冊入る本棚を生成する
      Bookshelf bshelf = new Bookshelf(10);

      // 本を生成して，本棚にしまう
      Book b1 = new Book("人間失格", "太宰治");
      bshelf.putBook(b1);

      Book b2 = new PictureBook(
         "どうぞのいす", "香山美子", "柿本幸造");
      bshelf.putBook(b2);

      Book b3 = new Magazine(
         "ムー", "ムー編集部", 2021, 12);
      bshelf.putBook(b3);

      // ※ 適当に本棚いっぱいまで本を加える

      bshelf.listBooks();

      // 指定された順番（位置）の本を取り出す
      Scanner sc = new Scanner(System.in);
      System.out.print("番号？ ");
      int i = sc.nextInt();

      Book b = bshelf.takeBook(i);

      // ※ 本があった場合だけその情報を表示する

      bshelf.listBooks();
   }
}

/* Book.java */
// 書籍を表すクラス
public class Book {

   private String title;   // タイトル
   private String author;  // 著者

   // ※ コンストラクタと getter を作成する
}

/* PictureBook.java */
// 絵本を表すクラス
public class PictureBook extends Book {

   private String illustrator;  // 絵の作者

   // ※ コンストラクタと getter を作成する
}
```

```
/* Magaine.java */
// 雑誌を表すクラス
public class Magazine extends Book {

   private int year;   // 発行年
   private int month;  // 発行月

  // ※ コンストラクタと getter を作成する
}

/* Bookshelf.java */
// 本棚を表すクラス
public class Bookshelf {

   // 本をしまう配列
   private Book [] books;

   // コンストラクタでは，本をしまう配列を作る
   public Bookshelf(int n) {
      books = new Book[n];
   }

   // 本をしまう
   public boolean putBook(Book b) {
      for (int i = 0; i < books.length; i++) {
         if (books[i] == null) {
            // ※ 配列の空いている要素にしまう


            return true;
         }
      }
      return false;
   }

   // 本を取り出す
   public Book takeBook(int i) {
      if (i < 0 || books.length <= i)
         return null;

      // ※ 配列の指定された要素から取り出す

      books[i] = null;
      return b;
   }

   // 本の一覧を表示する
   public void listBooks() {

      // ※ 本棚のすべての本の詳細情報を表示する

      // 設計としては，多態性で対応した方がよいが，
      // この問題では練習のため，instanceof を使う
   }
}
```

3. 前問に関して，Book，PictureBook，Magazine，Bookshelf の関係を UML のクラス図で表せ。