プログラミングA 第8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

添付の Access クラスの以下の各要素について、どの範囲から利用できるか（アクセス範囲）を簡潔に説明せよ。

- クラス自身
- name フィールド
- デフォルトコンストラクタ
- 引数を取るコンストラクタ
- dosome メソッド

参考資料：Access.java

```
package test;

class Access {
   String name = null;
   private Access() {
      super();
   }
   public Access(String s) {
      this();
      name = s;
   }
   protected void dosome() {
   }
}
```

2 宿題 2

添付の Access2 クラスおよび Access3 クラスがある。このとき、以下に答えよ。

- Access2 から、Access の name フィールドにアクセスできるか？ できない場合は当該フィールドのアクセス修飾子をどのように変更すればよいか？
- Access2 から、Access の dosome メソッドにアクセスできるか？ できない場合は当該メソッドのアクセス修飾子をどのように変更すればよいか？
- Access3 から、Access の name フィールドにアクセスできるか？ できない場合は当該フィールドのアクセス修飾子をどのように変更すればよいか？

- Access3 から、Access の dosome メソッドにアクセスできるか？ できない場合は当該メソッドのアクセス修飾子をどのように変更すればよいか？

参考資料：Access2.java

```
package test;

public class Access2 extends Access {
   public Access2() {
      super(null);
   }
}
```

参考資料：Access3.java

```
import test.Access2;

public class Access3 extends Access2 {

}
```

3 宿題 3

クラス DataClient の main メソッドにおいて、別のクラス Data のインスタンスフィールド number および text に直接アクセスしている。このとき、number および text について外部から直接にはアクセスでいないように隠蔽し、Data の外部から値を取得および参照するためのメソッドをそれぞれ新たに設けよ。この修正に伴い、DataClient も修正して、Data および DataClient の全体を提出せよ。

参考資料：Data.java

```
public class Data {

   public int number = 0;
   public String text = "TEXT";

}
```

参考資料：DataClient.java

```
public class DataClient {

   public static void main(String[] args) {
      Data d = new Data();
      d.number = 100;
      d.text = "NEW";
      System.out.println(d.number);
      System.out.println(d.text);
   }
```

```
}
```

4　宿題 4

Queue は、整数を要素として扱えるキューを表している。キューとは、先入れ先出し（First In First Out: FIFO）のデータ構造である。ここで以下のそれぞれを簡潔に説明せよ。理解にあたり、Queue 自身が持つ main メソッドを実行するとよい。

- enqueue メソッドの仕組み。
- dequeue メソッドの仕組み。
- フィールド values について、キューのサイズ（キューが扱う要素数の最大値）よりも、一つ大きなサイズ（配列が扱う要素数の最大値）の配列としている理由。逆に言うと、values = new int[size]; としていない理由。

参考資料：Queue.java

```
public class Queue {
   private int[] values = null;
   private int first = 0;
   private int last = 0;

   public Queue(int size) {
      values = new int[size+1];
   }

   public void enqueue(int data) {
      if (isFull()) throw new RuntimeException();
      values[last] = data;
      last = (last + 1) % values.length;
      System.out.println("Enqueue: " + data);
   }

   public int dequeue() {
      if(isEmpty()) throw new RuntimeException();
      int data = values[first];
      first = (first + 1) % values.length;
         System.out.println("Dequeue: " + data);
           return data;
   }

   public boolean isFull() {
      return (((last + 1) % values.length) == first);
   }

   public boolean isEmpty() {
```

```
    return (last == first);
  }

  public static void main(String[] args) {
    Queue q = new Queue(3);
    for(int i = 0; !q.isFull(); i++) q.enqueue(i);
    q.dequeue();
    q.enqueue(4);
    q.dequeue();
    while (!q.isEmpty()) q.dequeue();
  }
}
```

5　宿題 5

以下のそれぞれについて，この小テスト（宿題）内で扱うプログラムソースコードを参照する形で説明せよ．

(1) パッケージの使いどころ（何を目的としてどのような場合に用いるのか）
(2) アクセス修飾子の種類とアクセス範囲
(3) スレッドの使いどころ