平成23年度基盤システム演習A第3回レポート

学籍番号：0312010142
講座名　：澤本研
氏名　　：藤田　拓

目 次

◆課題

1 Studentクラス

```
//Student.java
//Person subclass

public class Student extends Person {
    String lab;
    String num;

    public Student( String name, int age, String lab, String num ) {
        super( name, age );
        this.lab = lab;
        this.num = num;
    }

    public Student() {
        super();
        this.lab = "";
        this.num = "";
    }

    public boolean isMylab( String l ) {
        return lab.equals( l );
    }

    public String selfIntroduction() {
        String si = super.selfIntroduction();
        return ( si + "I'm a member of the " + lab + " lab. " + "My student ID is " + num + "." );
    }

    public String account() {
        int yea = 'a' + ( Integer.parseInt( num.substring( 3, 7 ) ) % 26 );

        return ( "g031" + (char)yea + num.substring( 7, 10 ) );
    }

    public static void main(String [] args) {
        Student p1 = new Student( "Fujita", 20, "Dais", "0312010142" );
        System.out.println( p1.selfIntroduction() );

        String s = "Dais";

        if( p1.isMylab( s ) ) {
```

```
            System.out.println(p1.name + "'s lab is " + s + ".");
        } else {
            System.out.println(p1.name + "'s lab is not " + s + ".");
        }

        System.out.println( "My account is " + p1.account() + ".");
    }
}
```

紡歯腟

```
$ javac Student.java
$ java Student
Hi, my name is Fujita. I am 20 years old.I'm a member of the Dais lab. My student ID is 0312010142.
Fujita's lab is Dais.
My account is g031i142.
```

2 TypeTest

```
b1 = false
b2 = true
b3 = false
b4 = true
```

■ b1 は a の値が TypeB1 のクラスではないので false
b2 は b の値が TypeA のサブクラスなので true
b3 は c の値が TypeB1 のクラスでは無いので false
b4 は c の値が TypeB2 のクラスなので true

3 Usage インターフェイス

3.1 Student

```
//StudentUsage.java
//Person subclass

public class StudentUsage extends Person implements Usage {
    String lab;
    String num;

    public StudentUsage( String name, int age, String lab, String num ) {
        super( name, age );
        this.lab = lab;
        this.num = num;
    }
```

```
public StudentUsage() {
    super();
    this.lab = "";
    this.num = "";
}

public boolean isMylab( String l ) {
    return lab.equals( l );
}

public String selfIntroduction() {
    String si = super.selfIntroduction();
    return ( si + "I'm a member of the " + lab + " lab. " + "\nMy student ID is " + num + "." );
}

public String account() {
    int yea = 'a' + ( Integer.parseInt( num.substring( 3, 7 ) ) % 26 );

    return ( "g031" + (char)yea + num.substring( 7, 10 ) );
}

public void usage() {
    System.out.println( "スーパークラス:Person" );
    System.out.println( "学生を表すクラス" );
}

public void methods() {
    System.out.println( "boolean isMylab( String l ) 自分の所属研究室かどうかの検査" );
    System.out.println( "String selfIntroduction() 自己紹介を返す" );
    System.out.println( "void account() 学生番号からユーザアカウントを作成し返す" );
}

public void fields() {
    System.out.println( "String lab: 所属研究室を格納 (" + lab + ")" );
    System.out.println( "String num: 学籍番号を格納 (" + num + ")" );
}

public static void main(String [] args) {
    StudentUsage p1 = new StudentUsage( "Fujita", 20, "Dais", "0312010142" );
    System.out.println( p1.selfIntroduction() );

    String s = "Dais";

    if( p1.isMylab( s ) ) {
        System.out.println(p1.name + "'s lab is " + s + ".");
    } else {
```

```
            System.out.println(p1.name + "'s lab is not " + s + ".");
        }

        System.out.println( "My account is " + p1.account() + ".");
    }
}
```

実行結果

```
$ java UsageStudent
スーパークラス:Person
学生を表すクラス
boolean isMylab( String l ) 自分の所属研究室かどうかの検査
String selfIntroduction() 自己紹介を返す
void account() 学生番号からユーザアカウントを作成し返す
String lab: 所属研究室を格納 (Dais)
String num: 学籍番号を格納 (0312010142)
```

3.2 Queue

```
//QueueUsage.java

public class QueueUsage implements Usage {
    private int count;
    private int capacity;
    private int capacityIncrement;
    String [] itemList;

    public QueueUsage() {
        count    = 0;
        capacity = 5;
        capacityIncrement = 2;
        itemList = new String[capacity];
    }

    public QueueUsage(String [] list) {
        count = list.length;
        capacity = list.length;
        capacityIncrement = 5;
        itemList = list;
    }

    public void push (String obj) {
        if(count == capacity) {
            capacity += capacityIncrement;
            String [] tempList =new String[capacity];
```

```
            for (int i = 0; i < count; i++) {
                tempList[i] = itemList[i];
            }
            itemList = tempList;
        }

        itemList[count] = obj;
        count++;
    }

    public String shift() {

        if (count == 0) {
            return null;
        }
        else {
            String temps = itemList[0];
            int y = 0;
            for(int i = 1; i < count; i++) {
                itemList[y] = itemList[i];
                y++;
            }
            count--;
            return temps;
        }
    }

    public void printItems() {
        for(int i = 0; i < count; i++) {
            System.out.print(itemList[i] + ",");
        }
        System.out.println(" ");
    }


    public void usage() {
        System.out.println( "キューを行うクラス" );
    }

    public void methods() {
        System.out.println( "void push(String obj) キューに値を入れる" );
        System.out.println( "String shift() プッシュされたら一個前にずらす。容量を越えたら
先頭の値を返す" );
        System.out.println( "void printItems() キューの内容を表示" );
    }
```

```
    public void fields() {
        System.out.println( "private int count: キューのアイテム数" );
        System.out.println( "private int capacity: キューの配列の数量" );
        System.out.println( "private int capacityIncrement: 配列の数量を増やす" );
        System.out.println( "String [] itemList: キューの配列" );

    }
}

//UsageQueue.java

public class UsageQueue {
    public static void main (String [] args) {
        String [] items = {};
        QueueUsage mystack = new QueueUsage(items);

        mystack.usage();
        mystack.methods();
        mystack.fields();
    }
}
```

実行結果

```
$ java UsageQueue
キューを行うクラス
void push(String obj) キューに値を入れる
String shift() プッシュされたら一個前にずらす。容量を越えたら先頭の値を返す
void printItems() キューの内容を表示
private int count: キューのアイテム数
private int capacity: キューの配列の数量
private int capacityIncrement: 配列の数量を増やす
String [] itemList: キューの配列
```