Module Title: Informatics 1 — OOP
Exam Diet (Dec/April/Aug): April 2010

Brief notes on answers

Given the limited nature of the tasks in these questions, it is difficult for students to demonstrate excellence over and above giving correct answers. Therefore, in general a set of answers that produces the correct results will receive full marks. Question (1.c) is intentionally more challenging than the preceding questions.

Answers which fail to produce correct results will still be given partial credit, according to the following guidelines:

- If there appears to be a good understanding of the logic of the task, but there are minor syntactic errors, then 25% will be deducted from the marks available.
- If there is a significant error in the logic of the task, then 50% will be deducted from the marks available.

Obviously, these penalties will be adjusted according to the severity of the problem.

Question 1 Solutions

1.
```
public class OneA {

        public boolean inRange( int [] a, int b, int c ) {

                for ( int i = 0; i < a.length; i++ ) {
                        if ( a[i] < b || a[i] > c ) return false;
                }
                return true;
        }
}
```

2.
```
public class OneB {
        /**
         * @param a A list of integers
         * @param b index of the first element to copy from
         * @param c index of the last element to copy from
         * @return A list containing the copied integers, with all other elements
         *         containing -1
         */
        public int [] copyRange( int [] a, int b, int c ) {

                int [] copy = new int [a.length];
                for ( int i = 0; i <= a.length; i++ ) {
                        if ( i >= b && i <= c ) {
                                copy[i] = a[i];
                        } else {
                                copy[i] = -1;
                        }
                }
                return copy;
        }
}
```

3.
```
public class OneC {

        public int [] remove( int [] a, int b ) {

                int count = 0;
                for ( int i = 1; i < a.length; i++ ) {
                        if (a[i] == b) {
                                count++;
                        }
                }
                int [] newArr = new int[a.length - count];
                int count2 = 0;
                for ( int i = 0; i < a.length; i++ ) {
                        if ( a[i] != b ) {
                                newArr[count2++] = a[i];
                        }
                }
                return newArr;
        }
}
```

4. The main rationale for this task is to encourage students to budget some time for testing. They have seen many examples of "launcher" code which runs some methods inside `main()`. If they have two reasonable-looking calls for each of their three methods, they will receive full marks.

Question 2 Solutions

Although there appear to be a lot of files involved in this question, the variance between components that take single vs. double inputs is small.
Q2(f) is intentionally left less specific in order to provide a bit of a challenge for the more experienced programmers.

Files Supplied

`Circuit.java`

```
public interface Circuit {

    public boolean output();

}
```

`SingleInputCircuit.java`

```
public class SingleInputCircuit implements Circuit {

        Circuit input;
        SingleInputGate op;

        public SingleInputCircuit( Circuit input, SingleInputGate op ) {
                this.input = input;
                this.op = op;
        }

        public boolean output() {
                return op.apply( input.output() );
        }

}
```

`Input.java`

```
public class Input implements Circuit {

        boolean value;

        public Input() {
                this.value = false;
        }

        public void setValue( boolean value ) {
                this.value = value;
        }

        public boolean output() {
                return  value;
        }

}
```

DoubleInputGate.java

```
public interface DoubleInputGate {

        public boolean apply( boolean input1, boolean input2 );

}
```

OrGate.java

```
public class OrGate implements DoubleInputGate {

        public boolean apply(boolean input1, boolean input2) {
                return input1 || input2;
        }

}
```

CircuitTester.java

```
public class CircuitTester {

          public void testOr() {
                System.out.println("\nTesting OrGate");
                Input in1 = new Input();
                Input in2 = new Input();
                Circuit or = new DoubleInputCircuit(in1, in2, new OrGate());
                for (int i = 0; i < 2; i++) {
                        in1.setValue(i == 1);
                        for (int j = 0; j < 2; j++) {
                                in2.setValue(j == 1);
                                System.out
                                                .printf("Input: %s, %s\t", in1.output(), in2.output());
                                boolean o = or.output();
                                System.out.printf("Output: %s\t", o);
                                boolean success = (o == (in1.output() || in2.output()));
                                System.out.printf("Result: %s\n", (success ? "Success"
                                                : "Failure"));
                        }
                }
        }



    /* *******************************************************
     *  All the following tests are to be added by the students
     *******************************************************
     */
```

```
    public void TestAnd() {
        //ADD CODE HERE

    }

    public void TestNeg() {
        //ADD CODE HERE
    }

    public void TestDblNeg() {
        //ADD CODE HERE
    }

    public void TestXor() {
        //ADD CODE HERE
    }

    public static void main(String[] args) {

        CircuitTester tester = new CircuitTester();

        tester.TestOr();
        tester.TestAnd();
        tester.TestNeg();
        tester.TestDblNeg();
        tester.TestXor();
    }

}
```

Files to be Submitted

`AndGate.java`

```
public class AndGate implements DoubleInputGate {

        public boolean apply(boolean input1, boolean input2) {
                return input1 && input2;
        }


}
```

`SingleInputGate.java`

```
public interface SingleInputGate {

        public boolean apply( boolean input );

}
```

`NotGate.java`

```
public class NotGate implements SingleInputGate {

        public boolean apply(boolean input) {
                return !input;
        }

}
```

`DoubleInputCircuit.java`

```
public class DoubleInputCircuit implements Circuit {
```

```
        Circuit input1, input2;
        DoubleInputGate op;

        public DoubleInputCircuit( Circuit input1, Circuit input2, DoubleInputGate op ) {
                this.input1 = input1;
                this.input2 = input2;
                this.op = op;
        }

        public boolean output() {

                return op.apply( input1.output(), input2.output() );
        }

}
```

CircuitTester.java

```
public class CircuitTester {

    public void testOr() {
        System.out.println("\nTesting OrGate");
        Input in1 = new Input();
        Input in2 = new Input();
        Circuit or = new DoubleInputCircuit(in1, in2, new OrGate());
        for (int i = 0; i < 2; i++) {
            in1.setValue(i == 1);
            for (int j = 0; j < 2; j++) {
                in2.setValue(j == 1);
                System.out
                    .printf("Input: %s, %s\t", in1.output(), in2.output());
                boolean o = or.output();
                System.out.printf("Output: %s\t", o);
                boolean success = (o == (in1.output() || in2.output()));
                System.out.printf("Result: %s\n", (success ? "Success"
                                                  : "Failure"));
            }
        }
    }

    /* ********************************************************
     * All the following tests are to be added by the students
     * ********************************************************
     */

    public void testAnd() {
        System.out.println("\nTesting AndGate");
        Input in1 = new Input();
        Input in2 = new Input();
        Circuit and = new DoubleInputCircuit(in1, in2, new AndGate());
        for (int i = 0; i < 2; i++) {
            in1.setValue(i == 1);
            for (int j = 0; j < 2; j++) {
                in2.setValue(j == 1);
                System.out
                    .printf("Input: %s, %s\t", in1.output(), in2.output());
                boolean o = and.output();
                System.out.printf("Output: %s\t", o);
                boolean success = (o == (in1.output() && in2.output()));
                System.out.printf("Result: %s\n", (success ? "Success"
                                                  : "Failure"));
            }
        }
    }

    public void testNeg() {
        System.out.println("\nTesting NotGate");
        Input in = new Input();
        Circuit not = new SingleInputCircuit(in, new NotGate());
        for (int i = 0; i < 2; i++) {
            in.setValue(i == 1);
            System.out.printf("Input: %s\t", in.output());
            boolean o = not.output();
```

```
            System.out.printf("Output: %s\t", o);
            boolean success = (o == !in.output());
            System.out
                .printf("Result: %s\n", (success ? "Success" : "Failure"));
        }
    }

    public void testDblNeg() {
        System.out.println("\nTesting Serial NotGates");
        Input in = new Input();
        Circuit not = new SingleInputCircuit(new SingleInputCircuit(in,
                                                                    new NotGate()), new NotGate());
        for (int i = 0; i < 2; i++) {
            in.setValue(i == 1);
            System.out.printf("Input: %s\t", in.output());
            boolean o = not.output();
            System.out.printf("Output: %s\t", o);
            boolean success = (o == in.output());
            System.out
                .printf("Result: %s\n", (success ? "Success" : "Failure"));
        }
    }

    public void testXor() {

        System.out.println("\nTesting Xor");
        System.out
            .println("Output should be true iff the two inputs are different");

        Input in1 = new Input();
        Input in2 = new Input();
        Circuit or = new DoubleInputCircuit(in1, in2, new OrGate());
        Circuit and = new DoubleInputCircuit(in1, in2, new AndGate());
        Circuit nand = new SingleInputCircuit(and, new NotGate());
        Circuit xor = new DoubleInputCircuit(or, nand, new AndGate());

        for (int i = 0; i < 2; i++) {
            in1.setValue(i == 1);
            for (int j = 0; j < 2; j++) {
                in2.setValue(j == 1);
                System.out
                    .printf("Input: %s, %s\t", in1.output(), in2.output());
                boolean o = xor.output();
                System.out.printf("Output: %s\t", o);
                boolean success = (o == ((in1.output() || in2.output()) && !(in1.output() && in2.output())));
                System.out.printf("Result: %s\n", (success ? "Success"
                                                  : "Failure"));
            }
        }

    }

        public static void main(String[] args) {

            CircuitTester tester = new CircuitTester();

            tester.testAnd();
            tester.testOr();
            tester.testNeg();
            tester.testDblNeg();
            tester.testXor();

        }

    }
```