단국대학교 응용컴퓨터공학 JAVA 프로그래밍 I 중간고사 (2017년 가을학기) 2017년 10월 23일

학과 _____ 학번 _____ 이름 _____

중간고사

담당교수: 단국대학교 응용컴퓨터공학 박경신

- 답은 반드시 답안지에 기술할 것. 공간이 부족할 경우 반드시 답안지 몇 쪽의 뒤에 있다고 명기한 후 기술할 것. 그 외의 경우의 답안지 뒤쪽이나 연습지에 기술한 내용은 답안으로 인정 안 함. 답에는 반드시 네모를 쳐서 확실히 표시할 것.
- 답안지에 학과, 학번, 이름 외에 본인의 암호(4자리 숫자)를 기입하면 성적공고시 학번대신 암호를 사용할 것임.

1. 아래의 코드를 보고 빈 칸을 채워라. (20점)

```
public enum ArithmeticOperator {
        PLUS,  MINUS,  TIMES,  DIVIDE,  REMAINDER
}
public interface Scalable {
        void scale(int factor); // (1.1) scale by factor
}
public interface Moveable {
        void move(int x, int y); // (1.2) move by x,y
}
public abstract class Printer {
        private String name;
        protected Printer(String name) {
                this.name = name;
        }
        public String getName() {
                return this.name;
        }
        public abstract void print();// (1.3) print
}
public class Value extends Printer implements Scalable, Moveable {
        protected int a;
        protected int b;
        static int count = 0;
        public Value() {
                this(0, 0);
        }
```

학과 _____ 학번 _____ 이름 _____

```
public Value(int[] value) {
        this(value[0], value[1]); // (1.4) constructor
}
public Value(int a, int b) {
        super("Value" + (++count));
        set(a, b);
}
public void set(Value ab) { // set
        __this.a = ab.getA();__ // (1.5) set a
        __this.b = ab.getB();__ // (1.6) set b
}
public void set(int a, int b) { // set
        this.a = a;
        this.b = b;
}
public int getA() { // get a
        return this.a;
}
public int getB() { // get b
        return this.b;
}
public static void resetCount() { // reset count = 0
        count = 0;
}
public static int getCount() { // get count
        return count;
}
public String toString() { // toString override
        return getName() + " (" + this.a + "," + this.b + ")";
}
public void print() { // print override
        System.out.println(getName() + " (" + this.a + "," + this.b + ")");
}
public void scale(int factor) { // scale override
        this.a *= factor;
        this.b *= factor;
}
```

```
	public void move(int x, int y) { // move override
		this.a += x;
		this.b += y;
	}
}
public class Values {
	private Value[] values = null; // value array
	public Values(int size) {
		this.values = new Value[size];
	}
	public Values(Value[] values) {
		this.values = values;
	}
	public void setValueAt(int index, Value value) { // (1.7) set value if index in bound
		if (index >= 0 && index < this.values.length) this.values[index] = value;
	}
	public int getSize() { // (1.8) get size
		return this.values.length;
	}
	public Value[] getValues() { // (1.9) get values
		return this.values;
	}
	public Value getValueAt(int index) { // (1.10) get value at index
		try {
			return this.values[index];
		}
		catch (java.lang.ArrayIndexOutOfBoundsException e) {
			return null;
		}
	}
	public void print() {
		System.out.println(java.util.Arrays.toString(this.values));
	}
	public String toString() { // toString override
		return java.util.Arrays.toString(this.values);
	}
}
```

학과 _____ 학번 _____ 이름 _____

2. 다음 질문에 답하라 (80점)

```
class MainTest {
	static java.util.Scanner input = new java.util.Scanner(System.in);
	static void calculateByUserInput() { (2.4) calculateByUserInput
		System.out.print("Please enter the arithmetic operation (예시: 10 + 20): ");
		int a = input.nextInt();
		char op = input.next().charAt(0);
		int b = input.nextInt();
		__int c = calculate(getArithmeticOperator(op), a, b);___ // (2.4)
		System.out.println(a + " " + op + " " + b + " = " + c);
	}
	static char getOperatorChar(ArithmeticOperator op) { // (2.4)
		if (op == ArithmeticOperator.PLUS) return '+';
		else if (op == ArithmeticOperator.MINUS) return '-';
		else if (op == ArithmeticOperator.TIMES) return '*';
		else if (op == ArithmeticOperator.DIVIDE) return '/';
		else if (op == ArithmeticOperator.REMAINDER) return '%';
		else throw new AssertionError("Unknown operations");
	}
	static ArithmeticOperator getArithmeticOperator(char op) {
		switch (op) {
			case '+': return ArithmeticOperator.PLUS;
			case '-': return ArithmeticOperator.MINUS;
			case '*': return ArithmeticOperator.TIMES;
			case '/': return ArithmeticOperator.DIVIDE;
			case '%': return ArithmeticOperator.REMAINDER;
			default: throw new AssertionError("Unknown operations");
		}
	}
	static int calculate(ArithmeticOperator op, int x, int y) { // (2.3) calculate
		switch (op) {
			case PLUS: return x + y;
			case MINUS: return x - y;
			case TIMES: return x * y;
			case DIVIDE: return x / y;
			case REMAINDER: return x % y;
			default: throw new AssertionError("Unknown operations");
```

학과 _____ 학번 _____ 이름 _____

```
        }
    }
    static void print(String str, Object o) {
        System.out.print(str);
        System.out.println(o);
    }
    static void print(String str, int[] arr) {
        System.out.print(str);
        System.out.println(java.util.Arrays.toString(arr));
    }
    static void print(String str, Value[] arr) {
        System.out.print(str);
        System.out.println(java.util.Arrays.toString(arr));
    }
    static void fill(Value[] arr) { // (2.1)
        for (int i = 0; i < arr.length; i++)
            arr[i] = new Value(i*2, i*3);
    }
    static void fill(Values arr) { // (2.1)
        for (int i = 0; i < arr.getSize(); i++)
            arr.setValueAt(i, new Value((i+1)*10, i+2));
    }
    static void set(int a, int b) { // (2.7)
        a = b;
    }
    static void set(int[] arr) { // (2.7)
        arr[0] *= arr[1];
    }
    static void set(Value v) { // (2.7)
        v.set(v.getA() * v.getB(), v.getB());
    }
    public static void main(String[] args) {
        System.out.println("1."); ////////////////////////////////////////// (1)
        int[] arr1 = { 1, 2, 3 };
        int[] arr2 = { 4, 5, 6, 7, 8 };
        print("arr1: ", arr1); // arr1: [1, 2, 3]
        print("arr2: ", arr2); // arr2: [4, 5, 6, 7, 8]
```

학과 _____ 학번 _____ 이름 _____

```
Value[] vArray1 = new Value[5];
fill(vArray1);
print("vArray1: ", vArray1);
Values vArray2 = new Values(3);
fill(vArray2);
print("vArray2: ", vArray2);
Values vArray3 = new Values(vArray1);
print("vArray3: ", vArray3);
Values vArray4 = vArray2;
print("vArray4: ", vArray4);
vArray4.setValueAt(5, new Value(100, 200));
print("After setValueAt vArray4: ", vArray4);
System.out.println("2."); ////////////////////////////////////////// (2)
int i = 0;
while(i < vArray4.getSize()) { // (2.2) while
        Value v = vArray4.getValueAt(i++);
        v.print();
}
i = 0;
do { // (2.2) do-while
        Value v = vArray4.getValueAt(i++);
        v.print();
} while(i < vArray4.getSize());
for(Value v : vArray4.getValues()) { // (2.2) foreach
        v.print();
}
for(i=0; i<vArray4.getSize(); i++) { // (2.2) for
        Value v = vArray4.getValueAt(i);
        v.print();
}
System.out.println("3."); ////////////////////////////////////////// (3)
for (Value v : vArray4.getValues()) { // (2.3) calculate
        for (ArithmeticOperator op : ArithmeticOperator.values()) {
                int c = calculate(op, v.getA(), v.getB());
                System.out.println(v.getA() + " "
                        + getOperatorChar(op) + " " + v.getB() + " = " + c);
        }
```

```
        }
        System.out.println("4."); ////////////////////////////////////////// (4)
        calculateByUserInput();// (2.4) calculateByUserInput
        System.out.println("5."); ////////////////////////////////////////// (5)
        print("Value count=", Value.getCount()); // (2.5) static method
        Value.resetCount(); // (2.5) static method
        print("Value count=", Value.getCount()); // (2.5) static method
        System.out.println("6."); ////////////////////////////////////////// (6)
        Printer[] vArray5 = {new Value(1,2), new Value(3, 4)};
        for(Printer v : vArray5) {
                v.move(1, 2); // error => ((Moveable)v).move(1,2);
                v.print();
                v.scale(5); // error => ((Scalable)v).scale(5);
                v.print();
        }
        System.out.println("7."); ////////////////////////////////////////// (7)
        int[] intArray = {2, 3};
        set(intArray[0], intArray[1]);
        print("intArray : ", intArray);
        set(intArray);
        print("intArray : ", intArray);
        Value value = new Value(new int[] {2,3});
        set(value.getA(), value.getB());
        System.out.println("value : [" + value.getA() + "," + value.getB() + "]");
        set(value);
        System.out.println("value : [" + value.getA() + "," + value.getB() + "]");
        System.out.println("8."); ////////////////////////////////////////// (8)
        Object obj = new Value();
        if (obj instanceof Value) {
                ((Value)obj).move(10, 20);
                ((Value)obj).print();
                ((Value)obj).scale(3);
        }
        System.out.println(obj);
    }
}
```

학과 _____ 학번 _____ 이름 _____

2.1 main()에 (1)은 method overloading의 예를 보여주고 있다. 실행결과를 적고 그 이유를 설명하라 (예를 들어, 어떤 fill과 print 메소드를 사용한 것인지). (10점)

```
arr1: [1, 2, 3] // print(String str, int[] arr) 사용
arr2: [4, 5, 6, 7, 8] // print(String str, int[] arr) 사용
vArray1: [Value1 (0,0), Value2 (2,3), Value3 (4,6), Value4 (6,9), Value5 (8, 12)] // fill(Value[] arr) 와
print(String str, Value[] arr) 사용
vArray2: [Value6 (10,2), Value7 (20,3), Value8 (30,4)] // fill(Values arr)와 print(String str, Object o) 사용
vArray3: [Value1 (0,0), Value2 (2,3), Value3 (4,6), Value4 (6,9), Value5 (8, 12)] // Values(Value[] arr) 와
print(String str, Value[] arr) 사용
vArray4: [Value6 (10,2), Value7 (20,3), Value8 (30,4)] // vArray2 대입 print(String str, Object o) 사용
After setValueAt vArray4: [Value6 (10,2), Value7 (20,3), Value8 (30,4)] // setValueAt을 시도했으나 index
bound가 아니어서 지정못함 따라서 원래의 vArray4가 그대로 출력
```

2.2 main()에 (2)는 while-loop을 사용하여 vArray4를 출력하는 내용을 참고하여, do-while, foreach, for 구문을 완성하라. 위의 코드에 빈칸에 직접 작성한다. (10점)

2.3 main()에 (3)은 vArray4를 이용한 ArithmeticOperator 연산을 보여주고 있다. 실행결과를 적어라. (10점)

```
10 + 2 = 12
10 - 2 = 8
10 * 2 = 20
10 / 2 = 5
10 % 2 = 0
20 + 3 = 23
20 - 3 = 17
20 * 3 = 60
20 / 3 = 6
20 % 3 = 2
30 + 4 = 34
30 - 4 = 26
30 * 4 = 120
30 / 4 = 7
30 % 4 = 2
```

2.4 main()에 (4)는 UserInput을 사용하여 ArithmeticOperator 를 연산하는 예를 보여주고 있다. 위의 코드에 빈칸에 직접 작성한다. (10점)

2.5 main()에 (5)는 Value 클래스의 static 멤버의 사용을 보여주고 있다. 실행결과를 적고 그 이유를 설명하라. (10점)

```
// Value 클래스의 count는 Value(int a, int b) 생성자가 호출될 때 증가한다. fill(vArray1)에서 생성자
// 5번 호출, fill(vArray2)에서 3번 호출, setValueAt(5, new Value(100,200))에서도 1번 더 호출
Value count = 9
// Value.resetCount() 후에 count는 0
Value count = 0
```

2.6 main()에 (6)는 abstract class와 interface를 사용하는 예를 보여주고 있다. 밑줄 친 두 부분은 error: cannot find method가 발생한다. 위의 코드에 에러를 수정하고 실행결과를 적어라. (10점)

```
((Moveable)v).move(1,2); // Printer 를 Moveable로 type casting
((Scalable)v).scale(5); // Printer 를 Scalable로 type casting
Value1 (2,4)
Value1 (10,20)
Value2 (4,6)
Value2 (20,30)
```

학과 _____ 학번 _____ 이름 _____

2.7 main()에 (7)는 set 메소드 parameter passing의 예를 보여주고 있다. 실행결과를 적고 그 이유를 설명하라. 자바에서 primitive type과 reference type의 경우 parameter passing이 어떻게 되는지 간단히 설명하라. (10점)

set(int, int)는 int (primitive type)을 parameter로 보내는 것이라 값이 복사된다.
set(int[])는 배열(reference type)을 parameter로 보내면 reference가 복사된다.
set(Value)는 클래스(reference type)을 parameter로 보내면 reference가 복사된다.

intArray: [2, 3] // set(int, int)후에 값 변동 없음
intArray: [6, 3] // set(int[])후에 값 변동 생김
value: [2, 3] // set(int, int)후에 값 변동 없음
value: [6, 3] // set(Value)후에 값 변동 생김

2.8 main()에 (8)는 upcasting, downcasting, dynamic binding의 예를 보여주고 있다. Upcasting, downcasting, dynamic binding을 자세히 설명하라. 위의 코드 (6) (8)에서 사례를 찾아서 설명하라. (10점)

upcasting은 하위 클래스의 객체를 상위 클래스의 객체로 변환하는 것으로 자동으로 타입변환된다.

downcasting은 상위 클래스의 객체를 하위 클래스의 객체로 변환하는 것으로 명시적으로 type casting해준다.

dynamic binding은 상위 클래스의 가상메소드(virtual method) 또는 추상메소드(abstract method)를 하위 클래스에서 메소드 재정의(method overriding)한 경우, 실행시 동적 바인딩이 이루어지며 해당 객체의 overriding된 메소드가 자동으로 호출되어지는 것이다.

```
Object obj = new Value(); // upcasting
if (obj instanceof Value) {
        ((Value)obj).move(10, 20); // downcasting
        ((Value)obj).print(); // downcasting
        ((Value)obj).scale(3); // downcasting
}
System.out.println(obj); // Value.toString 호출 dynamic binding (30, 60)
Printer[] vArray5 = {new Value(1,2), new Value(3, 4)}; // upcasting
for(Printer v : vArray5) {
        ((Moveable)v).move(1, 2); // type casting
        v.print(); // Value.print 호출 dynamic binding
        ((Scalable)v).scale(5); // type casting
        v.print(); // Value.print 호출 dynamic binding
}
```

-끝-