Programmering Fortsättning

Övning 2, Typer

Joachim von Hacht

1 Kompatibilitet

1. Rita ett UML-diagram som visar hur klasserna och gränssnitten nedan är relaterade (korrekta UML-pilar). Den omslutande klassen Compatibility tas inte med. Avgör och motivera därefter för varje punkt a)-h) i main nedan om: 2p

- Det uppstår ett kompileringsfel.
- Det kommer att bli ett runtime-fel.
- Om det fungerar och i så fall vad som skrivs ut.

```
package _1_1;
public class Compatibility {

    public interface IA {
        public void doA();
    }

    public interface IB {
        public void doB();
    }

    // static has no impact (just because this class is inside other class)
    public static class C implements IA {
        public void doA() {
            System.out.println("C doA");
        }

        public void doC() {
            System.out.println("C doC");
        }
```

```
}

public static class D extends C implements IB {
    public void doC() {
        System.out.println("D doC");
    }

    public void doB() {
        System.out.println("D doB");
    }
}

public static class E extends D {
    public void doB() {
        System.out.println("E doB");
    }
}

public static class F extends C implements IB {
    public void doB() {
        System.out.println("F doB");
    }
}

public static void main(String[] args) {
    // a
    C c = new E();
    c.doC();

    // b
    Object o = new E();
    System.out.println(o instanceof D);

    // c
    C c1 = new D();
    c1.doC();

    // d
    IB b = new E();
    D d = (D) b;
    d.doB();

    // e
    C c2 = new F();
```

```
            c2.doB();

            // f
            IB b1 = new F();
            D d1 = (D) b1;

            // g
            D d2 = new C();
            d2.doA();

            // h
            IA a = new F();
            C c3 = a;
            c3.doA();
        }
    }
```

2 Gränssnitt

1. Kommer följande att fungera eller ej. Förklara? 2p

```
a)
public interface IA {
   public int doIt();
   public double doIt();
}
b)
public interface IA {
   public abstract void doIt();
   public abstract int doOther();
   public abstract int doYetOther(int i);
}

public interface IB {
  public abstract void doIt();
  public abstract void doOther();
  public abstract int doYetOther(String i);
}

public class ImplAB implements IA, IB {
    :
}

c)
```

```
public interface IA {
  public void doIt();
  public void doIt(int i);
}

public class ImplA implements IA {
  public void doIt() {
    System.out.println("Doing it");}
  public void doIt(int i) {
    System.out.println("Doing it int" + i);}
  public void doIt(String s) {
    System.out.println("Doing it string " + s); }
}
```

2. Vad skrivs ut i main? Varför? Red ut det hela! 2p

```
package _2_2;
public interface IA {
  public void doIt(Object o);
  public void doIt(B b);
}

package _2_2;
public class B {
   public void sayIt() {
      System.out.println("I'm B");
   }
}

package _2_2;
public class ImplA implements IA {
   public void doIt(Object o) {
      System.out.println("Hello");
   }
   public void doIt(B b) {
      b.sayIt();
   }
}

package _2_2;
public class ImplA2 implements IA {
   public void doIt(Object o) {
      System.out.println("See you");
   }
```

```
   public void doIt(B b) {
      b.sayIt();
   }
}

package _2_2;
public class Main {
   public static void main(String[] args) {
      IA ia = new ImplA();
      ImplA2 a2 = new ImplA2();
      B b = new B();
      Object o = b;
      ia.doIt(o);
      ia = a2;
      ia.doIt(o);
      ia.doIt(b);
   }
}
```

3 Overload och Override

1. Vad kommer att skrivas ut? Varför? Kommer programmet att exekvera utan problem? 1p

```
package _3_1;
public class Main {
   public static void main(String[] args) {
        Classifier c = new Classifier();
        Object[] o = new Object[] {
                new Integer(0), new String(), new Double(0)
        };
        for (int i = 0; i < o.length; i++) {
            c.classify(o[i]);
        }
    }
}

package _3_1;
public class Classifier {
   public void classify(Integer i) {
        System.out.println("It's an integer");
    }
    public void classify(String s) {
```

```
        System.out.println("It's a String");
    }
    public void classify(Object o) {
        System.out.println("Don't know, it's anything...");
    }
}
```

2. Vad kommer att skrivas ut i Main? Varför? 2p

```
package _3_2;
public class Vehicle {
    protected int licNumb;
    public Vehicle(int licNumb) {
        this.licNumb = licNumb;
    }
    public boolean equals(Vehicle v) {
        if (this == v) {
            return true;
        }else if (v == null) {
            return false;
        }else if (getClass() != v.getClass()) {
            return false;
        }
        final Vehicle other = (Vehicle) v;
        if (licNumb != other.licNumb){
            return false;
        }
        return true;
    }
}
```

```
package _3_2;
public class Car extends Vehicle {
    private int numbSeats;
    public Car(int licNumb, int numbSeats) {
        super(licNumb);
        this.numbSeats = numbSeats;
    }
    public boolean equals(Car c) {
        if (this == c) {
            return true;
        } else if (!super.equals(c)) {
            return false;
```

```
        } else if (getClass() != c.getClass()) {
            return false;
        }
        final Car other = (Car) c;
        if (numbSeats != other.numbSeats){
            return false;
        }
        return true;
    }
}

package _3_2;
public class Main {
    public static void main(String[] args) {
        Car c1 = new Car(123,1);
        Car c2 = new Car(123,2);
        Vehicle v1 = c1;
        Vehicle v2 = new Vehicle(123);
        if( v1.equals(c2)){
            System.out.println("v1 eq c2");
        }
        if( !v1.equals(v2)){
            System.out.println("v1 NOT eq v2");
        }
    }
}
```

4 Arrayer

1. Kompilerar koden nedan? Fungerar det runtime? Motivera! 1p

```
package _4_1;
public class Main {
    public static void main(String args[]) {
       B[] b = new B[]{ new B(), new B(), new B() };
       upDate( b );
       b[1].doB();
    }
    public static void upDate( A[] a ){
        a[0] = null;
        a[1] = new A();
        a[2] = new B();
    }
}
```

```
package _4_1;
public class A {
    public void doA(){
        System.out.println(this.getClass() + " doing A");
    }
}

package _4_1;
public class B extends A {
    public void doB(){
        System.out.println( this.getClass() + " doing B");
    }
}
```