Question Sheet 1

Name: _____ ID: _____

These questions do not have a formal, definitive answer. They are meant to be food for thoughts. Feel free to seek answers on browsing the Internet, talking to other software developers or reading books.

1. Many authors say that there are two kinds of polymorphism: universal and ad-hoc. What is the difference between one and the other? How is each of these types of polymorphism further divided?

2. Write a Java program that defines an overloaded method.

3. Most of the well known programming languages do not let the programmer to use the return type to differentiate overloaded methods. Why that is so?

4. What is a type coercion? Write an example of a Java program that contains an implicit coercion.

5. Which of the following calls of the function `f` are legal?

```
public class Coercion {
  public static void f(double x) {
    System.out.println(x);
  }
  public static void main(String args[]) {
    f(3.1416);
    f((byte)1);
    f((short)2);
    f('a');
    f(3);
    f(4L);
    f(5.6F);
  }
}
```

6. Which functions will be called in the program below?

```
public class Square {
  public static int square(int a) {
    System.out.println("Square of int: " + a);
    return a * a;
  }
  public static double square(double a) {
    System.out.println("Square of double: " + a);
    return a * a;
  }
  public static void main(String args[]) {
    square(1);
    square(1.0);
    square('a');
  }
}
```

7. Which type of polymorphism characterizes the program below, and how does it relate to Liskov's Substitution Principle?

```
public class Sub {
  public static void print(Object o) {
    System.out.println(o);
  }
```

```
  public static void main(String[] a) {
    print(new String("dcc024"));
    print(new Integer(42));
    print(new Character('a'));
  }
}
```

8. Describe the problems in the code below, and show how to improve it.

```
List myIntList = new LinkedList();
myIntList.add(new Integer(0));
Integer x = (Integer) myIntList.iterator().next();
```

9. Write an interface `Cont`, that is a *wrapper*. A wrapper's only purpose is to encapsulate an object. Then write a generic class `ContImpl`, that implements `Cont`, so that the code below will be legal:

```
public static void main(String args[]) {
  ContImpl<Integer> c = new ContImpl<Integer>();
  c.set(2);
  System.out.println(c.get());
}
```

10. In general, if S is subtype of T, then it is not the case that $G\langle S\rangle$ is subtype of $G\langle T\rangle$. Why is that so? Illustrate with a piece of code the dangers of removing this rule of a programming language with subtyping and parametric polymorphism.

11. Are there differences, in terms of implementation, of the two functions below?

```
public static void printCollection(Collection c) {
  Iterator i = c.iterator();
  while (i.hasNext()) {
    System.out.println(i.next());
  }
}
public static void printCG(Collection<?> c) {
  for (Object e : c) {
    System.out.println(e);
  }
}
```

12. Why is the code `Collection<?> c = new ArrayList<String>()` illegal in Java?

13. What is the problem with the code below? Re-write this function, so that the code is improved.

```
public void drawAll(List<Shape> shapes, Graphics g) {
  for (Shape s: shapes) {
    s.draw(g);
  }
}
```