Martin-Luther-Universität Halle-Wittenberg
Fachbereich Mathematik und Informatik
Dr. R. Picht

Telefon: +49 (345) 55-24 756
Email: picht@informatik.uni-halle.de

Halle, 22. Juni 2006

Programmiersprachen (SS 2006)

Übungsserie 12

Aufgabe 1 (Vererbung, Java)

Ordnen Sie folgende aus der Biologie bekannte Gattungsbegriffe in eine Klassenhierarchie: Rose, Fisch, Tier, Affe, Lebewesen, Schimpanse, Vogel, Pflanze, Mensch.

Aufgabe 2 (Vergleiche in Java)

Zwei Objekte sind in Java genau dann gleich, wenn ihre Referenzen identisch sind. Demgegenüber können unterschiedliche Objekte durchaus dieselben Attributwerte beinhalten. Java bietet deshalb mit == einen Referenzvergleich, der genau dann *true* liefert, wenn beide Objekte identisch sind. Auf Objekte ist dagegen eine Methode *equals* definiert, die einen Wertevergleich der Attribute vornimmt.

a. Welches Ergebnis hat das folgende Codestück:
```
String a = "te";
a += "xt";
String b = "text";
String c = a;
if (a == b ) System.out.println("a==b");
if (a.equals(b) ) System.out.println("a.equals(b)");
if (a == c ) System.out.println("a==c");
```

b. Implementieren Sie eine Vergleichsfunktion für die folgende Klasse:
```
class Person {
   int alter;
   boolean mann;
   String name;
}
```

c. Fügen Sie der Klasse *Person* ein Attribut *personalNummer* hinzu und nehmen Sie an, daß die Personalnummer eindeutig ist. Wie kann die Vergleichsfunktion *equals* umdefiniert werden?

Aufgabe 3 (Entwurfsentscheidungen)

a. Beschreiben Sie den Unterschied zwischen Vererbung und Import.

b. Beschreiben Sie den Unterschied zwischen Vererbung und Überladen.

c. Was ist der Unterschied zwischen einem Typ und einer Klasse?

Aufgabe 4 (Vererbung und Untertyp)

Wenn x ein Objekt der Klasse A und y ein Objekt der Klasse B ist, die von A abgeleitet wurde, warum ist dann $x.m$ nach der Zuweisung $x := y$ (Zeigerzuweisung in Simula) unzulässig, wenn m eine Methode von B, nicht aber von A ist?
Warum verletzt dies nicht das Subtypprinzip?

Aufgabe 5 (Vererbung, Java)

Gegeben ist folgende Subklassen-Struktur.
Berechnen Sie das Ergebnis des Aufrufs $new\,C().bar(2)$ unter Beachtung der dynamischen Bindung.

```
class A{
 public int foo(int x){
   return 2*x;
 }
 public int bar(int x){
   return foo(x+1)+1;
 }
}

class B extends A{
 public int foo(int x){
   if (x>5)
      return super.foo(x);}
   else
        return foo(bar(x));}
 }
}

class C extends B{
 public int foo(int x){
   return bar(x+2);
 }
 public int bar(int x){
   return super.foo(2*x)-1;
 }
}
```

Aufgabe 6 (Initialisierung und Vererbung)

Während der Initialisierung von Objekten ist es möglich , daß Initialisierungsroutinen von Vorfahren-Klassen aufgerufen werden müssen. Bei mehrfacher Vererbung können sich hieraus Fragen über die Reihenfolge der Initialisierungen ergeben. Beschreiben Sie dieses Problem und umreißen Sie eine mögliche Lösung.

Aufgabe 7 (Sichtbarkeit in Java)

Um zu verhindern, daß fremde Klassen bzw.deren Programmierer in unzulässiger Weise die Attribute einer Klasse modifizieren oder auslesen, wird in den Codierungsstandards empfohlen, Attribute zu privatisieren. Dadurch kann sichergestellt werden, daß Attribute von außen nicht zugreifbar sind und damit keine Verletzung von Invarianten erfolgen können. In den frühen Phasen der Entwicklung eines Systems ist es jedoch häufig sinnvoll, Attribute nicht in ihrer Sichtbarkeit zu beschränken bzw. explizit zu modellieren, ob das Attribut für andere Klassen zugreifbar sein soll. Dies geschieht mit den Schlüsselwörternpublic, protected und private. Ein Beispiel für solche Klasse ist:

```
class B {
  public     A attr1;
  protected  A attr2;
  private    A attr3;
             A attr4;
}
```

Sichern Sie die Klasse so ab, daß kein unbefugter Zugriff auf ihre Argumente erfolgen kann.

- Privatisieren Sie alle Variablenkomponenten und ermöglichen Sie die gleichen Lese- und Schreibzugriffe auf Attribute wie bisher durch Einführen von geeigneten Methoden.
  Welche Vorteile hat der Zugriff durch Methoden gegenüber dem direkten Attributzugriff?
- In Java gibt es nur Schlüsselwörter, die Lese- und Schreibzugriffe gleichermaßen einschränken. Häufig ist es jedoch so, daß ein Attribut von beliebigen Klassen gelesen, aber nur von der eigenen Klasse modifiziert werden darf. Wie müssen die mZuf griffsmethoden mit Schlüsselwörtern geschützt werden, um einen readonly-Zugriff zu realisieren?
- Welche Bedeutung hat die Verwendung des Schlüsselwortes protected in Java in Bezug auf Zugriffsschutz?

Besprechung ist am 27. Juni 2006.