Prof. Dr. Stefan Brass
Dipl.-Inform. Annett Thüring
Institut für Informatik
MLU Halle-Wittenberg

04. März 2013

Vorlesung "Objektorientierte Programmierung" — Probeklausur —

Name: _____

Matrikelnummer: _____

Studiengang: _____

Aufgabe	Punkte	von	Zeit
1 (Programmierung: Klasse)		9	15 min
2 (Programmierung: Array)		6	10 min
3 (Arithmetischer Ausdruck, Toter Code)		2	10 min
4 (Globale/Lokale Var., Call by Value/Ref.)		2	5 min
5 (Array, Referenzen)		1	5 min
6 (Rekursion)		2	5 min
7 (Vererbung)		2	5 min
8 (Fehlertypen)		3	5 min
9 (Fehlersuche)		3	10 min
Zusatzaufgabe		0 (+2)	10 min
Summe		30	80 min

☐ Ich fühle mich gesundheitlich in der Lage, diese Prüfung abzulegen. (Bitte sprechen Sie mit dem Aufsichtspersonal, falls Sie sich krank fühlen.)

☐ Für den Fall, daß ich nicht korrekt zu dieser Prüfung angemeldet sein sollte, erkläre ich mich sowohl damit einverstanden, daß ich rückwirkend angemeldet werde, als auch damit, daß diese Prüfung nicht zählt (Entscheidung des Dozenten).

Unterschrift: _____

Hinweise:

- Bearbeitungsdauer: 90 Minuten
- Skript, Bücher, Notizen sind erlaubt. Notebooks, PDAs, etc. dürfen nicht verwendet werden. Mobiltelefone bitte ausschalten (oder mit Aufsicht besprechen).
- Die Klausur hat 15 Seiten. Bitte prüfen Sie die Vollständigkeit.
- Bitte benutzen Sie den vorgegebenen Platz. Wenn Sie auf die Rückseite ausweichen müssen, markieren Sie klar, daß es eine Fortsetzung gibt.
- Tauschen Sie keinesfalls irgendwelche Dinge mit den Nachbarn aus. Notfalls rufen Sie eine Aufsichtsperson zur Kontrolle.
- Bei Aufgabe 8 zum Ankreuzen sollten Sie wenigstens raten, wenn Sie die richtige Lösung nicht wissen (wenn Sie nichts ankreuzen, haben Sie den Punkt auf jeden Fall verloren). Es ist jeweils genau eine Antwort pro Teilaufgabe richtig.
- Fragen Sie, wenn Ihnen eine Aufgabe nicht klar ist!
- Zum (garantierten) Bestehen benötigen Sie 60% der Punkte. Die Grenze wird möglicherweise gesenkt.

Zum Nachschlagen:

Aufgabe 1 (Programmierung: Klasse) 9 Punkte

Programmieren Sie eine Klasse `Ware`. Objekte der Klasse `Ware` sollen zwei Attribute haben: `name` und `preis`.

- Der Name soll ein klassischer String sein.
- Der Preis soll ein `int`-Wert sein (Preis in Cent).

Ihre Klasse soll drei Methoden haben:

- Den Konstruktor mit Parametern für die beiden Attribute. Bei der Konstruktion eines Objektes sollen also Name und Preis mit angegeben werden, z.B.

  ```
  Ware w1 = new Ware("Milch", 50);
  ```

  (wenn Milch 50 Cent kostet). Der Konstruktor soll testen, ob der gegebene Preis nicht negativ ist (also größer oder gleich 0), andernfalls soll er die Meldung `"Preis nicht positiv"` ausgeben. Auch im Fehlerfall sollen die Attribute des Objektes korrekt initialisiert werden, der Preis dann auf 0.
- Die Methode `get_name`, die den Namen liefert. Sie hat natürlich keine Parameter.
- Die Methode `get_preis`, die den Preis liefert (ebenfalls ohne Parameter).

Die Attribute sollen von außen nur über die Methoden `get_name` und `get_preis` zugreifbar sein, man soll z.B. außerhalb der Klasse nicht direkt eine Zuweisung schreiben können. Da die Klasse über keine Änderungsfunktionen verfügt, bleiben Name und Preis nach der Konstruktion fest (es hängt von der jeweiligen Anwendung ab, ob das realistisch ist). Dagegen sollen die drei Methoden selbstverständlich von außen zugreifbar sein.

In der zweiten Aufgabe finden Sie eine Anwendung der Klasse `Ware` und ein kurzes Testprogramm. Falls Ihnen die Aufgabenstellung nicht ganz klar ist, könnte es eventuell helfen, sich das schon anzuschauen.

Es ist Platz für die Lösung auf der nächsten Seite. Beachten Sie, daß auch für ein fehlendes oder falsches Semikolon ein halber Punkt abgezogen werden kann. Versuchen Sie also, Syntaxfehler zu vermeiden. Bemühen Sie sich außerdem um Verständlichkeit und guten Programmierstil. Es können auch für schlechten Stil Punkte abgezogen werden!

Platz für die Lösung von Aufgabe 1 (Klasse `ware`):

Aufgabe 2 (Programmierung: Array) 6 Punkte

Gegeben sei folgende Klasse für Kassenbons/Kassenzettel (Quittungen z.B. von Lebensmittelgeschäften). Sie enthält eine Folge von Referenzen auf Objekte der Klasse `Ware` in einem Feld (Array). Immer wenn eine Ware eingescannt oder eingetippt wird, wird eine Referenz auf das entsprechende `Ware`-Objekt im Array an die nächste freie Position gespeichert. Das Array kann also mehrfach den gleichen Wert (Referenz auf das gleiche `Ware`-Objekt) enthalten, nicht unbedingt hintereinander (mehrere Stück der gleichen Ware müssen auf dem Band nicht immer zusammen liegen).

```
public class Bon
{
    private Ware waren[];
    private int naechster_eintrag;  // naechster freier Eintrag im Array
    private int noch_frei;          // Anzahl noch freier Positionen

    public Bon()
    {
        waren = new Ware[100];
        naechster_eintrag = 0;
        noch_frei = waren.length;
    }

    // Methode zum Anhaengen einer Ware an den Bon:
    public void gescannt(Ware w)
    {
        if(noch_frei == 0)
            System.out.println("Zu viele Waren auf einen Bon!");
        else
        {
            waren[naechster_eintrag++] = w;
            noch_frei--;
        }
    }

    // Hier kommt Ihre Methode:
    public int rabatt(Ware w, int n)
    {
        ...
    }
}
```

Der Laden hat öfters Sonderangebote der Form "Kaufen Sie 5 Mehrkornbrötchen zum Preis von 4", allgemein: Wenn man `n` Stück der gleichen Ware `w` kauft, muß man nur `n-1` bezahlen, d.h. man bekommt einen Rabatt in Höhe des Preises der Ware `w`.

Man kann das Sonderangebot auch mehrfach in Anspruch nehmen, wenn man im obigen Beispiel etwa 10 Mehrkornbrötchen kauft, muß man nur 8 bezahlen, d.h. der Rabatt ist dann `2*` der Preis eines Mehrkornbrötchens. Kauft man dagegen 9 Stück, ist der Rabatt

nur der einfache Preis. Selbstverständlich kann der Rabatt auch `0` sein, wenn man weniger als `n` Stück der Ware `w` gekauft hat.

Schreiben Sie die Methode `rabatt` mit den Parametern `w` und `n`, wie oben angegeben, welche den erhaltenen Rabatt in Cent zurckgibt.. Sie müssen dazu die Methode `get_preis` der Klasse `Ware` verwenden.

Noch ein Hinweis: Der Vergleich der Einträge im Array und der gegebenen Ware `w` soll die Adresse der Objekt vergleichen (also Werte vom Typ `Ware`). Verschiedene Objekte zählen als verschiedene Waren (selbst wenn sie zufällig den gleichen Namen hätten — das wird die Anwendung verhindern).

Hier ist noch ein Test-Programm, um die Aufgabenstellung zu verdeutlichen:

```
public class Test
{
    public static void main(String[] args)
    {
        Ware milch = new Ware("Milch", 50);
        Ware butter = new Ware("Butter", 80);
        Bon mein_Kassenzettel = new Bon();

        mein_Kassenzettel.gescannt(butter);
        mein_Kassenzettel.gescannt(butter);
        mein_Kassenzettel.gescannt(milch);
        for(int i=1; i<= 5; i++)
            mein_Kassenzettel.gescannt(butter);
        System.out.println("Aktion: Kauf 3 zahle 2 (fuer Butter)!");
        System.out.println("Rabatt beim Kauf von 7 Stueck Butter; " +
                mein_Kassenzettel.rabatt(butter, 3) + " Cent");
    }
}
```

Die Ausgabe dieses Programmstücks ist:

```
Aktion: Kauf 3 zahle 2 (fuer Butter)!
Rabatt beim Kauf von 7 Stueck Butter; 160 Cent
```

Platz für die Lösung:

```
int rabatt(ware_t w, int n)
```

Aufgabe 3 (Arithmetischer Ausdruck, Toter Code) 2 Punkte

Betrachten Sie folgende Funktion:

```
public static int f(int n, int m)
{
    int i = n + 1, j;
    i++;
    j = 27 * m;
    n = j  % 5;
    j = i * m;
    while(j > 100)
        j -= 100;
    return j;
}
```

a) Streichen Sie in obigem Programm die Zeilen, die für den berechneten Wert irrelevant sind, also ohne Änderung der Semantik der Prozedur weggelassen werden können. Das gilt natürlich für Programmcode, der niemals ausgeführt wird, aber auch für berechnete Werte, die später nie verwendet werden.

b) Wie könnte man nun die gesamte Berechnung der obigen Funktion in einem arithmetischen Ausdruck zusammenfassen, also den Rumpf der Prozedur auf eine einzige `return`-Anweisung verkürzen?

Lösung:

```
int f(int n, int m)

{

    return _____;

}
```

Aufgabe 4 (Globale/lokale Var., Call by Value/Ref.) 2 Punkte

Was gibt dieses Programm aus?

```
public class Aufgabe4
{   private static Int n = new Int(5);

    public static void main(String[] args)
    {   int n = 2;
        int j = f(n);
        System.out.println(n*100+j);                        // 2. Ausgabe
    }

    public static int f(int i)
    {   Int n = new Int(3);
        g(n, i);
        i = n.getwert();
        return i;
    }

    public static void g(Int a, int b)
    {   System.out.println("a =" + a.getwert() + ": b = " + b);  // 1. Ausgabe
        a.setwert(a.getwert() * n.getwert() * b);
    }
}
class Int
{   private int wert;

    public Int(int wert)
    {  this.wert = wert;
    }

    public int getwert()
    {  return wert;
    }

    public void setwert(int wert)
    {  this.wert = wert;
    }
}
```

1. Ausgabe: _____

2. Ausgabe: _____

Aufgabe 5 (Array, Referenzen) 1 Punkt

Was gibt dieses Programm aus?

```
public class Aufgabe5
{
    public static void main(String[] args)
    {
        char[] a = new char[10];
        char[] p = a;
        for(int i=0; i<a.length; i++)
            a[i] = 'a';

        final char[] q = {'x','y','z'};
        for(int i=0; i<q.length; i++)
            p[i] = q[i];
        System.out.println(a[1]);
    }
}
```

Ausgabe: _____

Aufgabe 6 (Rekursion) 2 Punkte

Was gibt dieses Programm aus?

```
public class Aufgabe6
{
    public static void main(String[] args)
    {
        System.out.println("Ergebnis = " + f(6));
    }

    public static int f(int i)
    {
        if(i > 3)
        {
            return i * f(i-1);
        }
        else
            return 1;
    }
}
```

Ausgabe: _____

Aufgabe 7 (Vererbung) 2 Punkte

Was gibt dieses Programm aus?

```
public class Aufgabe7
{
    public static void main(String[] args)
    {
        D d = new D();

        System.out.println("d.f() = " + d.f());
        System.out.println("d.g() = " + d.g());
        System.out.println("d.h(4) = " + d.h(4));

        C c = d;
        System.out.println("c.g() = " + c.g());
    }
}

class C
{
    public int f() {return 1; }
    public int g() {return 2; }
    public int h(int i) {return 10 * i; }
}
class D extends C
{
    public int g() {return 5; }
    public int h(double i) {return 20; }
}
```

Ausgabe: d.f() = _____

Ausgabe: d.g() = _____

Ausgabe: d.h() = _____

Ausgabe: c.g() = _____

Aufgabe 8 (Fehlertypen) 3 Punkte

Welche der folgenden Fehler würde der Compiler bemerken und eine Fehlermeldung ausgeben? (Gehen Sie dabei von einem normalen Compiler aus, ohne einen extrem hohen Warnungslevel: Es zählen also nur echte Fehler, keine Warnungen.) Wenn Sie nichts ankreuzen, ist der Punkt auf jeden Fall verloren. Sie sollten daher notfalls raten.

a) Variable nicht deklariert.

- ☐ Der Compiler meldet den Fehler.
- ☐ Dieser Fehler wird vom Compiler nicht gemeldet und kann nur durch Testen gefunden werden.

b) Endlosschleife.

- ☐ Der Compiler meldet den Fehler.
- ☐ Dieser Fehler wird vom Compiler nicht gemeldet und kann nur durch Testen gefunden werden.

c) Übergabe eines `double`-Wertes (z.B. 5.0) für einen Funktionsparameter, der als `int p` deklariert ist.

- ☐ Der Compiler meldet den Fehler.
- ☐ Dieser Fehler wird vom Compiler nicht gemeldet und kann nur durch Testen gefunden werden.

Aufgabe 9 (Fehlersuche) 3 Punkte

Das folgende Programm enthält (mindestens) 4 Fehler. Bitte geben Sie drei dieser Fehler an (es ist möglicherweise schwierig, alle fünf zu entdecken). Falls Sie mehr als drei Fehler angeben, werden nur die ersten drei gewertet (es gibt keine Extrapunkte). Geben Sie bitte jeweils die Zeilennummer mit an, in der sich der Fehler befindet. Es zählt auch nicht als Fehler, daß das Programm nichts Sinnvolles tut. Direkte Folgefehler zählen auch nicht.

```
 1 public class Aufgabe9
 2 {
 3     public static void main(String[] args)
 4     {
 5         C x = new C(5);
 6         int i = x.f();
 7         (i+1) = 4;
 8         i +=g(27);
 9         if(i=0) i++;
10         System.out.println(i + x.g(3));
11     }
12 }
13
14 class C
15 {
16     private int a;
17     private int f() { return a*a; }
18     public int g (int i) { return f() + i; }
19     C(int n) { a = n; }
20 }
```

1. Zeile: _____

   Begründung: _____

   _____

2. Zeile: _____

   Begründung: _____

   _____

3. Zeile: _____

   Begründung: _____

   _____

Zusatzaufgabe (Programmierung) 2 Extrapunkte

Erweitern Sie die Klasse `Ware` aus Aufgabe 1 um eine Methode `print_preis` (ohne Parameter). Diese Methode soll den Preis in der Form "Euro.Cent" ausgeben. Sie müssen also den abgespeicherten Preis (in Cent) durch 100 teilen und das Ergebnis ausgeben, dann einen Punkt, dann eventuell eine zusätzliche 0 (falls der Cent-Wert zwischen 0 und 9 liegt), und dann den Cent-Wert (Rest bei Division des gespeicherten Preises durch 100). Wenn der gespeicherte Preis also z.B. `109` ist, muß diese Methode `1.09` ausgeben.

Sie brauchen nur die Methode selbst zu schreiben (nicht noch einmal die Klasse oder eine Subklasse etc.).

Platz für die Lösung: