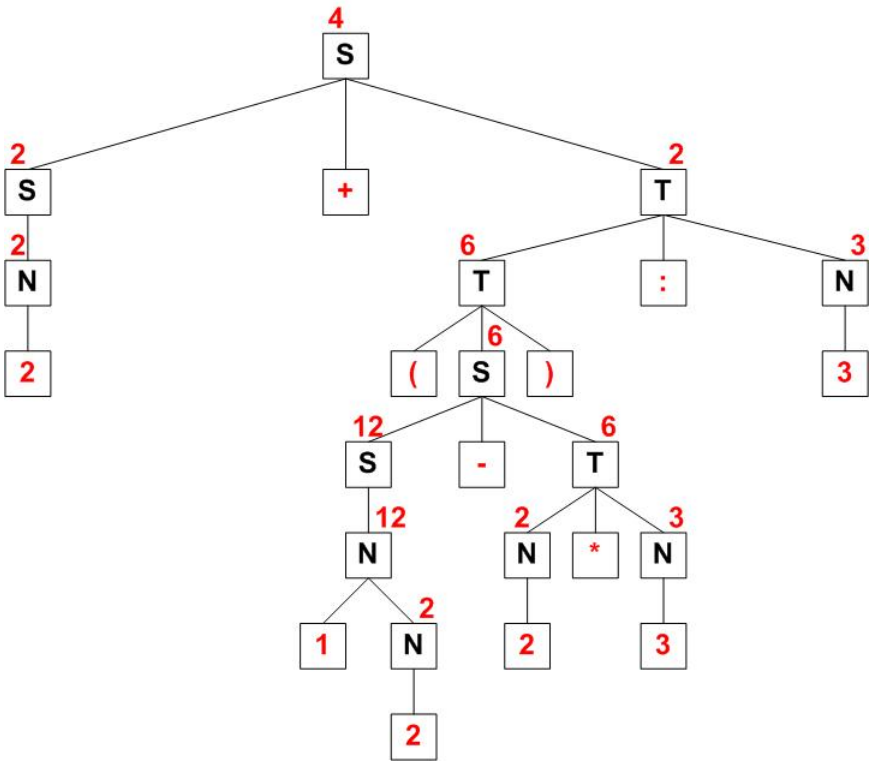
Technische Universität München
Fakultät für Informatik
D. Gmach, M. Rickert,
Dr. M. Schneider

WS 2005/2006
Lösungsvorschläge zu Blatt 13
03. Februar 2006

Übungen zu Einführung in die Informatik I

Aufgabe 52 Arithmetische Ausdrücke (Lösungsvorschlag)

a) Ableitungsbaum:



b) Kellerautomat $(Q, q_0, \Gamma, Z_0, F, \delta)$ über $\sum = \{0,1,2,3,4,5,6,7,8,9,+,-,*,:,(,)\}$
:
$Q = \{q_0, q_{number}, q_{left}, q_{right}, q_{op}, q_f\}$
$\Gamma = \{L, \perp\}$
$Z_0 = \perp$
$F = \{q_f\}$
δ:

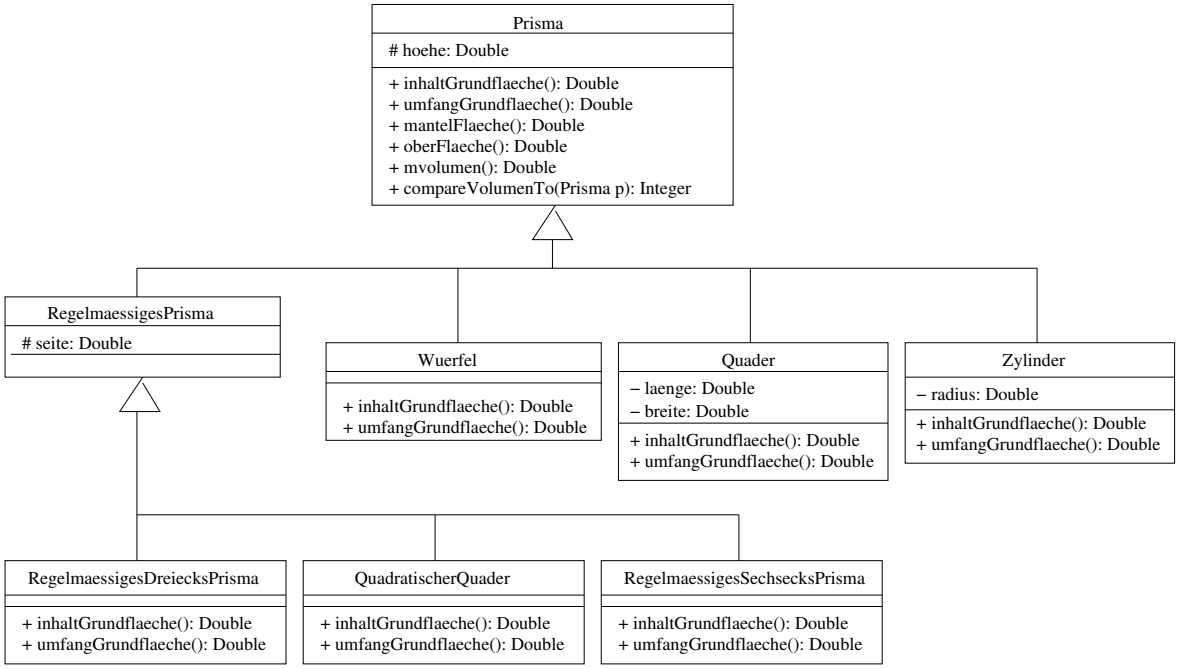
$$
\begin{aligned}
(q_0, n, \perp) &\rightarrow (q_{number}, \perp) \\
(q_{op}, n, x) &\rightarrow (q_{number}, x) \\
(q_{number}, n, x) &\rightarrow (q_{number}, x) \\
(q_{left}, n, L) &\rightarrow (q_{number}, L) \\
(q_{number}, o, x) &\rightarrow (q_{op}, x)
\end{aligned}
$$

$$
\begin{aligned}
(q_{right}, o, x) &\rightarrow (q_{op}, x) \\
(q_0, (, \perp) &\rightarrow (q_{left}, L) \\
(q_{op}, (, x) &\rightarrow (q_{left}, Lx) \\
(q_{left}, (, L) &\rightarrow (q_{left}, LL) \\
(q_{number},), L) &\rightarrow (q_{right}, \varepsilon) \\
(q_{right},), L) &\rightarrow (q_{right}, \varepsilon) \\
(q_{number}, \varepsilon, \perp) &\rightarrow (q_f, \perp) \\
(q_{right}, \varepsilon, \perp) &\rightarrow (q_f, \perp)
\end{aligned}
$$

mit
$n \in \{0,1,2,3,4,5,6,7,8,9\}$, $x \in \Gamma$, $o \in \{+,-,:,*\}$

Aufgabe 53 Vererbung (Lösungsvorschlag)

a) Eine geeignete Modellierung würde folgendermaßen aussehen:



b) Die Operationen `inhaltGrundflaeche()` und `umfangGrundflaeche()` sind von der Grundfläche des konkreten Prismas abhängig und können erst dort implementiert werden. Alle anderen Operationen können bereits in der Klasse `Prisma` implementiert werden und diese Implementierung an alle Unterklassen zur Wiederverwendung vererben. Dabei stützen sie sich teilweise auf die beiden Operationen `inhaltGrundflaeche()` und `umfangGrundflaeche()` ab.

c) Die Oberklasse `Prisma` für beliebige gerade Prismen:

```
public class Prisma {
   // Gemeinsames Attribut aller Prismen ist die Hoehe:
   protected double hoehe;

   // Konstruktor:
```

```
   public Prisma (double h) {
      hoehe = h;
   }

   // Die Operationen umfangGrundflaeche() und inhaltGrundflaeche()
   // sind abhaengig von der Form des konkreten Prismas und werden
   // hier nur aus technischen Gruenden definiert;
   // In den verschiedenen Unterklassen muessen diese Methoden dann
   // ueberschrieben werden. Später werden sie
   // durch abstrakte Methoden ersetzt:

public double umfangGrundflaeche(){
return 0;
}
public double inhaltGrundflaeche(){
return 0;
}

  // Die anderen Operationen koennen bereits in dieser Klasse implementiert
  // werden:

  public double mantelFlaeche() {
     return umfangGrundflaeche() * hoehe;
  }

  public double oberFlaeche() {
     return mantelFlaeche() + 2 * inhaltGrundflaeche();
  }

  public double volumen() {
     return inhaltGrundflaeche() * hoehe;
  }

  public int compareVolumenTo(Prisma p) {
     double v1 = volumen();
     double v2 = p.volumen();

     if (v1 < v2)
        return -1;
     else if (v1 > v2)
        return 1;
     else
        return 0;
  }
}
```

Die Klasse `Wuerfel` als Unterklasse von `Prisma`:

```
public class Wuerfel extends Prisma {

  // Konstruktor:
  public Wuerfel(double h) {
     super(h);
  }
```

```
   // Implementation der Operationen umfangGrundflaeche() und
   // inhaltGrundflaeche():

  public double umfangGrundflaeche() {
     return 4 * hoehe;
  }

  public double inhaltGrundflaeche() {
     return hoehe * hoehe;
  }
}
```

Die Klasse `Quader` als Unterklasse von `Prisma`:

```
public class Quader extends Prisma {

  // Zusaetzlich sind die Attribute Laenge und Breite der Grundflaeche
  // noetig:
  private double laenge;
  private double breite;

  // Konstruktor:
  public Quader(double h, double l, double b) {
     super(h);
     laenge = l;
     breite = b;
  }

   // Implementation der Operationen umfangGrundflaeche() und
   // inhaltGrundflaeche():

  public double umfangGrundflaeche() {
     return 2 * (laenge + breite);
  }

  public double inhaltGrundflaeche() {
     return laenge * breite;
  }
}
```

Die Klasse `Zylinder` als Unterklasse von `Prisma`:

```
public class Zylinder extends Prisma {

  // Zusaetzlich ist das Attribute Radius Grundflaeche noetig:
  private double radius;

  // Konstruktor:
  public Zylinder (double h, double r) {
     super(h);
     radius = r;
  }
```

```
   // Implementation der Operationen umfangGrundflaeche() und
   // inhaltGrundflaeche():

  public double umfangGrundflaeche() {
     return 2 * radius * Math.PI;
  }

  public double inhaltGrundflaeche() {
     return radius * radius * Math.PI;
  }
}
```

Die Klasse `RegelmaessigesPrisma` als Unterklasse von `Prisma` und Oberklasse für alle regelmäßigen Prismen:

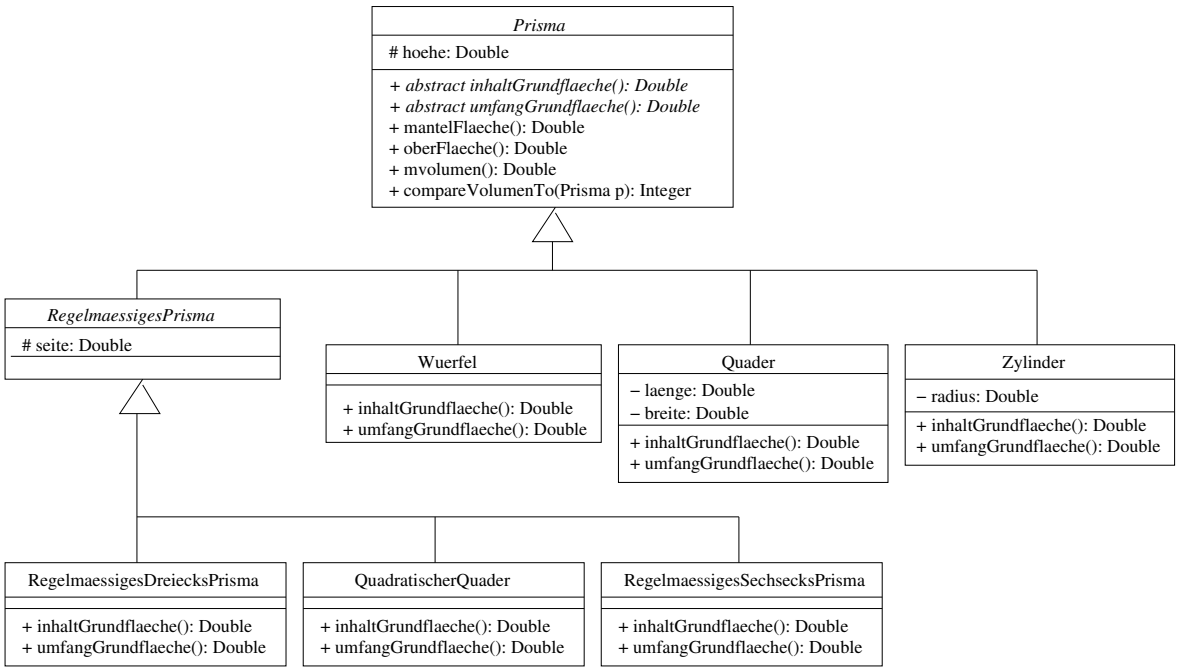
```
public class RegelmaessigesPrisma extends Prisma {

   // alle Grundflaechen regelmaessiger Prismen sind gleichseitige
   // Vielecke. Als zusaetzliches Attribut genuegt die Seitenlaenge:
   protected double seite;

   // Konstruktor:
   public RegelmaessigesPrisma(double h, double s) {
      super(h);
      seite = s;
   }
}
```

Die Klasse `RegelmaessigesSechsecksPrisma` als Unterklasse von `RegelmaessigesPrisma`:

```
public class RegelmaessigesSechsecksPrisma extends RegelmaessigesPrisma {

  // Konstruktor:
  public RegelmaessigesSechsecksPrisma(double h, double s) {
     super(s, h);
  }

   // Implementation der Operationen umfangGrundflaeche() und
   // inhaltGrundflaeche():

  public double umfangGrundflaeche() {
     return 6 * seite;
  }

  public double inhaltGrundflaeche() {
     return 3 * seite * seite * Math.sqrt(3) / 2;
  }
}
```

Aufgabe 54 Abstrakte Klassen (Lösungsvorschlag)

a) Sinnvollerweise werden die beiden Oberklassen `Prisma` und `Regelmäßiges Prisma` abstrakt deklariert. Das ergibt dann folgendes Klassendiagramm:



b) Die Klasse `Prisma` lautet dann:

```
abstract public class Prisma {
   // Gemeinsames Attribut aller Prismen ist die Hoehe:
   protected double hoehe;

   // Konstruktor:
   public Prisma (double h) {
      hoehe = h;
   }

   // Die Operationen umfangGrundflaeche() und inhaltGrundflaeche()
   // sind abhaengig von der Form des konkreten Prismas und koennen
   // hier nur abstrakt definiert werden:

  abstract public double umfangGrundflaeche();
  abstract public double inhaltGrundflaeche();

  // Die anderen Operationen koennen bereits in dieser Klasse implementiert
  // werden:
  // .......

}
```

die Klasse `Regelmaessiges Prisma`:

```
abstract public class RegelmaessigesPrisma extends Prisma {

   // alle Grundflaechen regelmaessiger Prismen sind gleichseitige
   // Vielecke. Als zusaetzliches Attribut genuegt die Seitenlaenge:
   protected double seite;
```

```
    // Konstruktor:
    public RegelmaessigesPrisma(double h, double s) {
        super(h);
        seite = s;
    }
}
```