TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen WS 2008/09
Einführung in die Informatik 2 **Übungsblatt 1**
Prof. Dr. Helmut Seidl, T.M. Gawlitza, S. Pott 14.10.08

Abgabe: 21.10.08 (vor der Vorlesung)

Aufgabe 1.1 (P) Aussagenlogik

Zeigen oder widerlegen Sie (Verwenden Sie dabei für Aufgabe a) eine Wahrheitstabelle, für alle anderen Aufgaben die Äquivalenzregeln der Aussagenlogik):

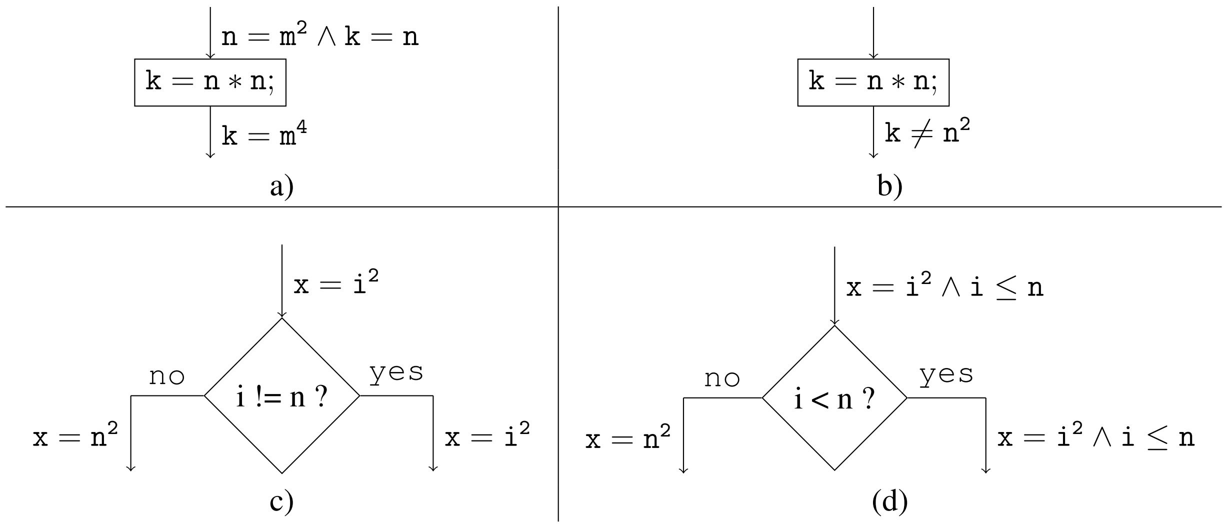
a) $A \iff B \equiv (A \Rightarrow B) \wedge (B \Rightarrow A)$
b) $A \iff B \equiv (A \wedge B) \vee (\neg A \wedge \neg B)$
c) $(\neg A \wedge (A \Rightarrow B)) \Rightarrow \neg B \equiv \mathbf{true}$
d) $(\neg B \wedge (A \Rightarrow B)) \Rightarrow \neg A \equiv \mathbf{true}$
e) $A \Rightarrow B \equiv \neg B \Rightarrow \neg A$
f) $(A \Rightarrow B) \wedge (B \Rightarrow C) \Rightarrow (A \Rightarrow C) \equiv \mathbf{true}$
g) $(A \Rightarrow B) \wedge (A \Rightarrow C) \equiv A \Rightarrow (B \wedge C)$

Vereinfachen Sie folgende Aussagen:

a) $(A \wedge \neg B) \vee (A \wedge B)$
b) $(A \Rightarrow B) \vee (B \Rightarrow A)$

Aufgabe 1.2 (P) Verifikation

Überprüfen Sie, ob folgende Zusicherungen lokal konsistent sind beziehungsweise ergänzen Sie fehldende Zusicherungen, so dass lokale Konsistenz hergestellt wird.



Aufgabe 1.3 (P) Verifikation

Gegeben sei folgendes MiniJava-Programm:

```
int n, i, r;
n = read();
i = 0;
r = 0;
while (i != n) {
        i = i + 1;
        r = r + 2*i*n;
        r = r - n;
}
write(r);
```

a) Erstellen Sie das Kontrollfluß-Diagramm!

b) Beweisen Sie, dass, falls eine Ausgabe erfolgt, $\mathtt{n}^3$ ausgegeben wird!

Zur Schleifen-Invariante: Zum Finden einer geeigneten Schleifen-Invariante gehen wir wie folgt vor: Wir bezeichnen den Wert der Programm-Variablen r nach der k-ten Iteration ($k = 0, 1, 2, \ldots$) einfach mal mit r_k. Insbesondere ist also $r_0 = 0$ (Der Schleifenrumpf wurde 0 mal ausgeführt). Weiterhin sieht man, dass sich der Wert der Programm-Variablen n nicht verändert. Der Wert der Programm-Variablen i ist nach der k-ten Iteration k.

Jetzt drücken wir den Wert r_k der Programm-Variablen r nach der k-ten Iteration mithilfe des Wertes r_{k-1}, k und des Wertes der Programm-Variablen n aus.

Scharfes Hinsehen führt zu folgender Vermutung:

$$r_k = \begin{cases} 0 & \text{falls } k = 0 \\ r_{k-1} + 2 \cdot k \cdot \mathtt{n} - \mathtt{n} & \text{falls } k > 0. \end{cases}$$

Das schreiben wir als Summe in der Form

$$r_k = \sum_{j=1}^{k} ??? = \sum_{j=1}^{\mathtt{i}} ???.$$

Achtung: r_k ist keine Programm-Variable. Der Wert r_k dient nur zur Überlegung. In der Invariante darf r_k nicht vorkommen.