



Informatik II Schleifeninvarianten


G. Zachmann
Clausthal University, Germany
zach@in.tu-clausthal.de



Technik zum Korrektheitsbeweis eines Algo

- Früher: Spezifikation = Zusicherungen, Vorbedingungen, Nachbedingungen
- Wie beweist man die Nachbedingungen, ausgehend von den Vorbedingungen?
- Wichtige Technik: Schleifeninvariante (*loop invariant*)
- Ein erfahrener und guter Programmierer hat zu jeder Schleife die Schleifeninvariante im Kopf!
 - Zu Beginn sollte man Schleifeninvarianten im Code als Kommentar hinschreiben
 - Hilft auch Programmier-Kollegen später


G. Zachmann Informatik 2 - SS 06 Schleifeninvarianten 2



Definition

- Invariante** = Zusicherung (prädikatenlogische Formel), die immer erfüllt bleibt
- Schleifeninvariante** := Zusicherung, die
 - unmittelbar vor Eintritt in eine Schleife gilt, und
 - am Ende des Schleifenrumpfes wieder gilt, und
 - eine "sinnvolle" Eigenschaft der Schleife wiedergibt.
- Später: damit kann man die formale Korrektheit eines Programms beweisen
 - Nach der Schleife gilt Schleifeninvariante und negierte Schleifenbedingung
 - Diese Zusicherung ist dann (hoffentlich) "näher" an der Nachbedingung dran.

G. Zachmann Informatik 2 - SS 06 Schleifeninvarianten 3



Beispiel

- Berechnung der Fakultät
- Schleifeninvariante
 - Am Schluß wollen wir haben: $f = n!$
 - Invariante: $f = (i - 1)!$
- Nachweis:
 - Vor der Schleife: ist gültig, da $f=i=1$
 - Durch die Schleife hindurch:
 - Annahme: Invariante gilt für i (hergestellt durch vorigen Schleifendurchlauf)
 - Schritt: zeige Invariante für f' und $i'=i+1$

$$f' = f \cdot i = (i - 1)! \cdot i = i! = (i' - 1)!$$
- Nach Ende der Schleife gilt:

$$f = (i - 1)! \wedge i = n + 1 \Rightarrow f = n!$$

```

i = 1
f = 1
# Invariante gültig
while i <= n:
    f *= i
    i += 1
# Invariante gü.

```

G. Zachmann Informatik 2 - SS 06 Schleifeninvarianten 4

Weiteres Beispiel

- Maximales Element in einem Array bestimmen:

```

# Vorbedingung:
# A = Liste; n = len(A)
m = 0
# Schleifeninvariante muß hier wahr sein
for j in range( 1, len(A) ):
    if A[j] > A[m]:
        m = j
    # Schleifeninvariante muß hier wahr sein
# Schleifeninvariante ist hier also auch wahr
    
```

- Nachbedingung: $A[m]$ soll maximales Element in A sein
- Sinnvolle Schleifeninvariante:
 $\forall k, 0 \leq k \leq j : A_k \leq A_m \wedge 0 \leq m < \text{len}(A)$

G. Zachmann Informatik 2 - SS 06 Schleifenvarianten 5

- Nachweis der Eigenschaften:

```

m = 0
for j in range( 1, len(A) ):
    if A[j] > A[m]:
        m = j
    
```

- Vor der Schleife:
 - Konkrete Werte für die Variablen einsetzen
 - $\forall k, 0 \leq k \leq 0 : A_k \leq A_0 \wedge 0 \leq k < \text{len}(A)$ ist wahr (nur $k=0$ ist möglich)
- Am Ende der Schleife: Beweisschema wie bei vollst. Induktion
 - Induktionsannahme: Invariante gilt für $j-1$
 - Induktionsschritt: zeige Invariante für j
 - Fall 1: $A_j \leq A_m$
 Der if -Body wird nicht ausgeführt, m wird nicht verändert, also
 $\forall k, 0 \leq k \leq j-1 : A_k \leq A_m \wedge 0 \leq m < \text{len}(A) \wedge A_j \leq A_m \Rightarrow$
 $\forall k, 0 \leq k \leq j : A_k \leq A_m \wedge 0 \leq m < \text{len}(A)$

G. Zachmann Informatik 2 - SS 06 Schleifenvarianten 6

- Fall 2: $A_j > A_m$
 Der if -Body wird ausgeführt, m wird gleich j gesetzt, also
 $\forall k, 0 \leq k \leq j-1 : A_k \leq A_m \wedge 0 \leq m < \text{len}(A) \wedge A_m = A_j \wedge j < \text{len}(A) \Rightarrow$
 $\forall k, 0 \leq k \leq j : A_k \leq A_m \wedge 0 \leq m < \text{len}(A)$

- Am Ende der Schleife gilt also in beiden Fällen wieder die Invariante
- Bemerkung: so können wir jetzt auch beweisen, daß am Ende das gewünschte Ergebnis herauskommt
 $\forall k, 0 \leq k \leq j : A_k \leq A_m \wedge 0 \leq m < \text{len}(A) \wedge j = \text{len}(A) - 1 \Rightarrow$
 $\forall k, 0 \leq k < \text{len}(A) : A_k \leq A_m \wedge 0 \leq m < \text{len}(A)$

```

m = 0
for j in range( 1, len(A) ):
    if A[j] > A[m]:
        m = j
    
```

G. Zachmann Informatik 2 - SS 06 Schleifenvarianten 7

- Später (beim Sortieren): noch mehr Beispiele (nicht so formal)
- Noch später (wahrscheinlich): Kalkül zur formalen, exakten Verifikation

G. Zachmann Informatik 2 - SS 06 Schleifenvarianten 8