educational engineering lab
Department for Information Technology
University of Zurich

Programmentwicklung

Edsger Wybe Dijkstra (1930-2002)

Entwickeln von Iterationen

```
// Q ... Precondition
Initialisierung;
// P ... Invariant
while (B) { // P and B
      S;
      // P
}
// (P and not B) ⇒ R ... Postcondition
```

Beispiel: Potenzieren (1)

geg: $x \in R$, $n \in Z$

ges: $y = x^n$

Axiome zur Spezifikation von x^n:

$x^n = 1$	falls $n=0$
$x^n = x*x^{n-1}$	falls $n>0$
$x^n = (1/x)^{-n}$	falls $n<0$
$x^n = (x^2)^{n/2}$	falls n gerade

Beispiel: Potenzieren (2)

Q: true — Precondition

R: $y = x^n$ — Postcondition

P: $y*w^i = x^n$ — Invariant

```
y = 1; w = x; i = n;
// y*w^i = x^n
while (i ≠ 0) { // (y*w^i = x^n) and (i ≠ 0), t: i
    S; // verkleinere i unter Invarianz von P
    // y*w^i = x^n
}
// ((y*w^i = x^n) and (i = 0) ) ⇒ y = x^n
```

Beispiel: Potenzieren (3)

```
// n ≥ 0
y = 1; w = x; i = n;
// (y*w^i = x^n) and (i ≥ 0)
while (i ≠ 0) { // (y*w^i = x^n) and (i > 0), t: i
   i = i-1; y = y*w;
   // (y*w^i = x^n) and (i ≥ 0)
}
// ((y*w^i = x^n) and (i = 0)) ⇒ y = x^n
```

Beispiel: Potenzieren (4)

```
if (n ≥ 0) {
   y = 1; w = x; i = n;
} else {
   y = 1; w = 1/x; i = -n;
}
// (y*w^i = x^n) and (i ≥ 0)
while (i ≠ 0) { // (y*w^i = x^n) and (i > 0), t: i
   i = i-1; y = y*w;
   // (y*w^i = x^n) and (i ≥ 0)
}
// ((y*w^i = x^n) and (i = 0) ) ⇒ y = x^n
```

Beispiel: Potenzieren (5)

```
if (n ≥ 0) {
   y = 1; w = x; i = n;
} else {
   y = 1; w = 1/x; i = -n;
}
// (y*w^i = x^n) and (i ≥ 0)
while (i ≠ 0) { // (y*w^i = x^n) and (i > 0), t: i
   if (i%2 == 0) { // i gerade
      i = i/2; w = w*w;
   } else {
      i = i-1; y = y*w;
   }
   // (y*w^i = x^n) and (i ≥ 0)
}
// ((y*w^i = x^n) and (i = 0) ) ⇒ y = x^n
```

Beispiel: Potenzieren (6)

```
if (n ≥ 0) {
   y = 1; w = x; i = n;
} else {
   y = 1; w = 1/x; i = -n;
}
// (y*w^i = x^n) and (i ≥ 0)
while (i ≠ 0) { // (y*w^i = x^n) and (i > 0), t: i
   while (i%2 == 0) { // i gerade
      i = i/2; w = w*w;
   }
   // (y*w^i = x^n) and (i > 0)
   i = i-1; y = y*w;
   // (y*w^i = x^n) and (i ≥ 0)
}
// ((y*w^i = x^n) and (i = 0) ) ⇒ y = x^n
```

Beispiel: Binäres Suchen

geg: $N \geq 0$, a[0..N-1] steigend sortiert, x

ges: Ein Index i so dass alle Elemente mit Indizes kleiner oder gleich i nicht grösser als x sind und alle Elemente mit Indizes grösser als i grösser als x sind.

Q: All k: $1 \leq k < N$: $a[k-1] \leq a[k]$ and $(N \geq 0)$

R: (All k: $0 \leq k \leq i$: $a[k] \leq x$) and (All k: $i+1 \leq k < N$: $a[k] > x$)

P: (All k: $0 \leq k \leq i$: $a[k] \leq x$) and (All k: $j \leq k < N$: $a[k] > x$) and $(i < j \leq N)$
invariant

```
i = -1; j = N; // P
while (i+1 ≠ j) { // t: j-i-1
    m = (i+j)/2; // i<m<j
    if (a[m]≤x) i=m else j=m; // P
}
```

Beispiel: Maximaler Kursgewinn

geg: N≥2, a[0..N-1] Aktienkurse der letzten N Tage

ges: Der maximale Kursgewinn g = a[j]-a[i] für j>i.

Q: N≥2

R: g = (Max i,j: 0≤i<j<N: a[j]-a[i])

P: (g = (Max i,j: 0≤i<j<n: a[j]-a[i])) and (2≤n≤N) and (min = (Min i: 0≤i<n: a[i]))

```
n = 2; g = a[1]-a[0]; min = min(a[0],a[1]); // P
while (n ≠ N) { // t: N-n
    g = max(g, a[min]);
    min = min(min,a[n]);
    n = n+1; // P
}
```

Weakest Precondition wp(S,R)

wp(S,R) ist die schwächste Vorbedingung die garantiert, dass das Programmstück S in einer endlichen Anzahl von Schritten die Endbedingung R sicherstellt.

Es gilt

$$(Q \Rightarrow wp(S,R)) \Leftrightarrow \{Q,S,R\}$$

aber auch

$$(Q \notin wp(S,R)) \Rightarrow \text{not } \{Q,S,R\}$$