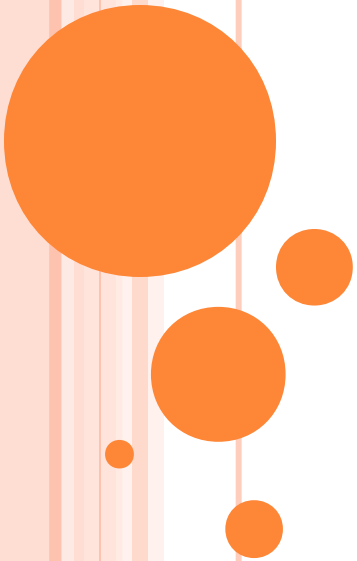

TRIPLE DI HOARE: ESEMPI ED ESERCIZI

Corso di Logica per la Programmazione

A.A. 2012/13

RICORDIAMO LA REGOLA PER IL COMANDO ITERATIVO

$$\frac{P \Rightarrow Inv \wedge def(E) \qquad Inv \wedge \sim E \Rightarrow Q \qquad Inv \Rightarrow t \geq 0 \qquad \{Inv \wedge E\}\ C\ \{Inv \wedge def(E)\} \qquad \{Inv \wedge E \wedge t = V\}\ C\ \{t < V\}}{\{P\}\ \textbf{while}\ E\ \textbf{do}\ C\ \textbf{endw}\ \{Q\}}$$

- t è chiamata **funzione di terminazione**
- Inv è chiamata **invariante**
- $Inv \Rightarrow t \geq 0$ è l'**ipotesi di terminazione**
- $\{Inv \wedge E\}\ C\ \{Inv \wedge def(E)\}$ è l'**ipotesi di invarianza**
- $\{Inv \wedge E \wedge t = V\}\ C\ \{t < V\}$ è l'**ipotesi di progresso**
- V è una **variabile di specifica**: denota un generico valore, non utilizzabile e non modificabile nel programma

ESEMPIO DI COMANDO ITERATIVO

- Usando come **invariante**

  $Inv : s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n$

e come **funzione di terminazione**

  $t : n - x$

verificare la tripla nel riquadro a destra.

$\{s = 0 \ \wedge x = 0 \wedge n \ge 0\}$
while $x < n$ **do**
    x, s := x+1, s+x
endw
$\{s = (\Sigma\, i: i \in [0, n).\ i)\}$

- Per la **Regola per il Comando Iterativo** è sufficiente mostrare:

1) $s = 0 \wedge x = 0 \wedge n \ge 0 \Rightarrow def(x<n) \wedge s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n$

2) $s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n \wedge \sim(x < n) \Rightarrow s = (\Sigma\, i: i \in [0, n).\ i)$

3) $s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n \ \Rightarrow n \dot{-} x \ge 0$

4) $\{s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n \wedge x < n\}$ x, s := x+1, s+x
   $\{s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n \wedge def(x < n)\}$

5) $\{s = (\Sigma\, i : i \in [0, x).\ i) \ \wedge 0 \le x \wedge x \le n \wedge x < n \wedge n - x = V\}$
   x, s := x+1, s+x $\{n - x < V\}$

Esercizio: completare la dimostrazione

COMANDO DI INIZIALIZZAZIONE

- Spesso la precondizione di una tripla con un **while** non è sufficiente per soddisfare la condizione $P \Longrightarrow Inv \wedge def(E)$
- In questo caso si può inserire un **comando di inizializzazione** C_I tale che $\{P\}\ C_I\ \{Inv \wedge def(E)\}$
- **Esempio. N**ella tripla vista, se la precondizione è solo $\{n \geq 0\}$, la 1) è falsa (invariante e *def*(x<n) non valgono).
- Possiamo renderla vera con un comando che inizializzi **x** e **s**.

$\{n \geq 0\}$??
while x < n **do**
x, s := x+1, s+x
endw
$\{s = (\Sigma\ i: i \in [0, n).\ i)\}$

$\{n \geq 0\}$
x, s := 0, 0 ;
$\{s = 0 \wedge x = 0 \wedge n \geq 0\}$
while x < n **do**
x, s := x+1, s+x
endw
$\{s = (\Sigma\ i: i \in [0, n).\ i)\}$

PROGRAMMI ANNOTATI

- Invece di indicare solo pre- e post-condizioni di un programma, come a destra, è utile aggiungere altre annotazioni per facilitarne la comprensione.
- Per esempio, annotiamo il programma come sotto con invariante, funzione di terminazione e altre asserzioni. Questo rende esplicito cosa bisogna dimostare:

1) $s = 0 \wedge x = 0 \wedge n \geq 0 \Rightarrow \mathit{def}(x<n) \wedge \mathit{Inv}$
2) $\mathit{Inv} \wedge \sim(x < n) \Rightarrow s = (\Sigma\ i: i \in [0, n).\ i)$
3) $\mathit{Inv} \wedge x < n \Rightarrow n - x \geq 0$
4) $\{\mathit{Inv} \wedge x<n\}$ x,s := x+1,s+x $\{\mathit{Inv} \wedge \mathrm{def}(x<n)\}$
5) $\{\mathit{Inv} \wedge x < n \wedge n-x = V\}$ x,s := x+1,s+x $\{n-x < V\}$

$\{ n \geq 0\}$
x, s := 0, 0 ;
while x < n **do**
    x, s := x+1, s+x
endw
$\{s = (\Sigma\ i: i \in [0, n).\ i)\}$

$\{n \geq 0\}$
x, s := 0, 0 ;
$\{s = 0 \wedge x = 0 \wedge n \geq 0\}$
$\{\mathit{Inv} : s = (\Sigma\ i : i \in [0, x).\ i) \wedge 0 \leq x \wedge x \leq n\}$ $\{t: n - x\}$
while x < n **do**
    x, s := x+1, s+x
endw
$\{\mathit{Inv} \wedge \sim(x < n)\}$
$\{s = (\Sigma\ i: i \in [0, n).\ i)\}$

ESERCIZIO: Calcolo MCD

- Si consideri il seguente programma annotato

$\{x = A \wedge y = B \wedge A>0 \wedge B>0\}$

$\{Inv : x>0 \wedge y>0 \wedge mcd(A,B) = mcd(x,y)\}\ \{t : x+y\}$

while x <> y **do**

    if x>y **then** x:= x-y; **else** y := y-x; **fi**

endw

$\{x=mcd(A,B)\}$

- Dimostrarne la correttezza, facendo uso delle seguenti note proprietà dell'operatore *mcd*:

$mcd\ (v,w) = v$ se v=w

$mcd\ (v,w) = mcd\ (v-w,w)$ se v>w

$mcd\ (v,w) = mcd\ (v,w-v)$ se v<w

ESERCIZIO: SCANSIONE DI SEQUENZA

- Si consideri il seguente programma annotato che conta il numero di elementi maggiori di zero in un array **a: array [0, n) of int**.

{a : array [0, n) of int }
x, c := 0, 0;
{*Inv* : c = #{j | j ∈ [0, x) ∧ a[j] > 0 } ∧ x∈ [0, n] } {*t* : n - x}
while x < n **do**
    if (a[x] > 0) **then** c := c+1 **else skip** **fi** ;
    x := x + 1
endw
{*Inv* ∧ ~(x < n)}
{ c = #{j : j ∈ [0, n) ∧ a[j] > 0 } }

- Scrivere e dimostrare la Condizione di Invarianza
- Scrivere e dimostrare la Condizione di Terminazione
- Scrivere e dimostrare la Condizione di Progresso

SPECIFICA DI PROBLEMI DI PROGRAMMAZIONE

- Sia **a: array [0,n) of int**. Fornire una specifica del seguente problema:

  " Incrementare di uno tutti gli elementi dell'array **a** "

- Dobbiamo fornire una precondizione P e una postcondizione Q tali che la tripla

  $$\{P\}\ C\ \{Q\}$$

  sia soddisfatta se e solo se il comando C realizza quanto specificato.

- Soluzione (si noti l'uso di una variabile di specifica per il valore iniziale dell'array):

  $\{n \geq 0 \ \wedge a : \text{array } [0, n) \text{ of int } \wedge (\forall k: k \in [0, n) \Rightarrow a[k] = V[k])\}$

  C

  $\{ (\forall k.\ k \in [0, n) \Rightarrow a[k] = V[k] + 1) \}$

ESERCIZI DI SPECIFICA DI PROBLEMI DI PROGRAMMAZIONE

- Siano **a,b: array [0,n) of int**. Fornire una specifica dei seguenti problemi:
  - Azzerare gli elementi di un array **a** con indice dispari
  - Verificare che tutti gli elementi pari di un array **a** sono seguiti da un elemento dispari
  - Calcolare la somma degli elementi dell'array **a** che sono uguali agli elementi di **b** nella medesima posizione
  - Calcolare nella variabile **p** il numero degli elementi di **a** che non compaiono in **b**

ESERCIZIO: MODIFICA DI SEQUENZA

- Si consideri il seguente programma annotato che incrementa tutti gli elementi di un array **a: array [0, n) of int**.

$\{n \geq 0 \ \wedge a : \text{array } [0, n) \text{ of int } \wedge (\forall k: k \in [0, n) \Rightarrow a[k] = V[k])\}$

x := 0;

$\{Inv : x \in [0, n] \wedge (\forall k: k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])\}\ \{t : n - x\}$

while x < n **do**

a[x] := a[x] + 1; x := x + 1

endw

$\{Inv \ \wedge \sim(x < n)\}$

$\{\ (\forall k.\ k \in [0, n) \Rightarrow a[k] = V[k] + 1)\ \}$

- Scrivere e dimostrare la Condizione di Invarianza
- Scrivere e dimostrare la Condizione di Terminazione
- Scrivere e dimostrare la Condizione di Progresso

- SOLUZIONE: Condizione di invarianza

$\{x \in [0, n] \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k]) \wedge x < n\}$

a[x] := a[x] + 1; **x := x + 1**

$\{x \in [0, n] \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k]) \wedge def(x < n)\ \}$

- Per la regola della sequenza dobbiamo determinare **R** in modo che

$\{x \in [0, n) \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])\ \}$

a[x] := a[x] + 1;

$\{\mathbf{R}\}$

x := x + 1

$\{x \in [0, n] \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])\ \}$

- In questo caso **R** viene determinata dall'assioma per l'assegnamento

$\{x \in [0, n) \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])\}$

a[x] := a[x] + 1;

$\{def(x+1) \wedge x+1 \in [0, n] \wedge (\forall k.\ k \in [0, x+1) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x+1, n) \Rightarrow a[k] = V[k])\}$

x := x + 1

$\{x \in [0, n] \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])\}$

- Ci resta da dimostrare la tripla

$\{x \in [0, n) \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])\}$

a[x] := a[x] + 1;

$\{x+1 \in [0, n] \wedge (\forall k.\ k \in [0, x] \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in (x, n) \Rightarrow a[k] = V[k])\}$

- Avendo a che fare con un aggiornamento selettivo, dobbiamo mostrare

$x \in [0, n) \wedge (\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])$

$\Rightarrow$

$x+1 \in [0, n] \wedge (\forall k.\ k \in [0, x] \Rightarrow b[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in (x, n) \Rightarrow b[k] = V[k]) \wedge x \in [0, n) \wedge def(a[x]+1)$

con $b = a[a[x]+1 \,/\, x]$

Alcune osservazioni:

(1) $def(a[x]+1) \equiv x \in [0, n)$ che quindi possiamo omettere

(2) $(\forall k.\ k \in [0, x) \vee k \in (x, n) \Rightarrow b[k] = a[k])$

(3) $b[x] = a[x] + 1$

$x+1 \in [0, n] \wedge (\forall k.\ k \in [0, x] \Rightarrow b[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in (x, n) \Rightarrow b[k] = V[k]) \wedge x \in [0, n)$

$\equiv$ { **Ip:** $x \in [0, n)$ }

$(\forall k.\ k \in [0, x] \Rightarrow b[k] = V[k] + 1) \wedge$

$(\forall k.\ k \in (x, n) \Rightarrow b[k] = V[k])$

$\equiv$ { Intervallo, $x \in [0, x]$ }

$(\forall k.\ k \in [0, x) \Rightarrow b[k] = V[k] + 1) \wedge\ b[x] = V[x] + 1 \wedge$

$(\forall k.\ k \in (x, n) \Rightarrow b[k] = V[k])$

$\equiv$ { Osservazioni (2) e (3) precedenti }

$(\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1) \wedge\ a[x] + 1 = V[x] + 1 \wedge$

$(\forall k.\ k \in (x, n) \Rightarrow a[k] = V[k])$

$\equiv$ { **Ip:** $(\forall k.\ k \in [0, x) \Rightarrow a[k] = V[k] + 1)$ }

$a[x] + 1 = V[x] + 1 \wedge (\forall k.\ k \in (x, n) \Rightarrow a[k] = V[k])$

$\equiv$ { calcolo, Intervallo }

$(\forall k.\ k \in [x, n) \Rightarrow a[k] = V[k])$

ESERCIZIO: Calcolo del fattoriale

- Si consideri la seguente specifica

  {n>0} C {f = n!}

```
{n>0}
   f , x:= 1, 1;
{Inv : ?} {t : ?}
while x <= n do
    f, x :=  f * x,  x + 1;
endw
{Inv  ∧ ~(x <= n)}
{f = n!}
```

- Come determinare l'invariante e la funzione di terminazione?

- Eseguiamo manualmente il programma (es: per n = 5)**:**

{n>0}	x	f
f , x:= 1, 1;	1	1
{*Inv* : ?} {*t* : ?}	**2**	**1**
while x <= n **do**	3	2
f, x := f * x, x + 1;	4	6
endw	5	24
{*Inv* ∧ ~(x <= n)}	**6**	**120**
{f = n!}		

- Osserviamo che, ad ogni iterazione, i valori di x e f sono legati dalla seguente relazione

  f = (x - 1)!

- Scegliamo questa formula come invariante candidata: non riusciamo però a dimostrare

  f = (x - 1)! ∧ ~(x <= n) ⇒ f = n!

```
{n>0}
   f , x:= 1, 1;
{Inv : f = (x - 1)! ∧ x∈ [0, n+1]} {t : ?}
while x <= n do
    f, x :=  f * x, x + 1;
endw
{Inv  ∧ ~(x <= n)}
{f = n!}
```

- In questo modo

$$Inv \wedge \sim(x <= n) \Rightarrow f = n!$$

- Dimostrazione per esercizio

Ipotesi di invarianza

$\{f = (x\text{-}1)! \wedge x \in [0, n+1)\}$ f, x := f * x, x + 1 $\{f = (x\text{-}1)! \wedge x \in [0, n+1]\}$

$f * x = x! \wedge x+1 \in [0, n+1]$

$\equiv$ { **Ip:** $f = (x\text{-}1)!$ }

$(x\text{-}1)! * x = x! \wedge x+1 \in [0, n+1]$

$\equiv$ { def. fattoriale}

$x+1 \in [0, n+1]$

$\Leftarrow$

$x \in [0, n+1)$

- Abbiamo dimostrato

$$x \in [0, n+1) \Rightarrow ((f = (x\text{-}1)! \Rightarrow f * x = x! \wedge x+1 \in [0, n+1]))$$

che sappiamo essere equivalente a quanto richiesto dalla tripla (perché?)

- La funzione di terminazione è lasciata per esercizio.