Precondición más débil de ciclos

Algoritmos y Estructuras de Datos I

Repaso: Triplas de Hoare

- Consideremos la siguiente tripla de Hoare:

$$\{P\}\ \mathbf{S}\ \{Q\}.$$

- Esta tripla es válida si se cumple que:
  1. Si el programa S comienza en un estado que cumple P ...
  2. ... entonces termina luego de un número finito de pasos ...
  3. ... y además en un estado que cumple Q.

Repaso: Lenguaje SmallLang

- Definimos un lenguaje imperativo basado en variables y las siguientes instrucciones:
  1. Nada: Instrucción **skip** que no hace nada.
  2. Asignación: Instrucción **x := E**.
- Además, tenemos las siguientes estructuras de control:
  1. Secuencia: **S1; S2** es un programa, si **S1** y **S2** son dos programas.
  2. Condicional: **if B then S1 else S2 endif** es un programa, si **B** es una expresión lógica y **S1** y **S2** son dos programas.
  3. Ciclo: **while B do S endwhile** es un programa, si **B** es una expresión lógica y **S** es un programa.

Repaso: Precondición más débil

- **Definición.** La precondición más débil de un programa **S** respecto de una postcondición Q es el predicado P más débil posible tal que $\{P\}\mathbf{S}\{Q\}$.
- **Notación.** $wp(\mathtt{S}, Q)$.
- **Teorema:** Decimos que $\{P\}\ \mathbf{S}\ \{Q\}$ es válida sii $P \Rightarrow_L wp(\mathtt{S}, Q)$

Repaso: Axiomas wp

- **Axioma 1.** $wp(\texttt{x := E}, Q) \equiv \operatorname{def}(E) \wedge_L Q^x_E$.
- **Axioma 2.** $wp(\texttt{skip}, Q) \equiv Q$.
- **Axioma 3.** $wp(\texttt{S1; S2}, Q) \equiv wp(\texttt{S1}, wp(\texttt{S2}, Q))$.
- **Axioma 4.** $wp(\texttt{if B then S1 else S2 endif}, Q) \equiv$

$$def(B) \wedge_L \Big((B \wedge wp(\texttt{S1}, Q)) \vee (\neg B \wedge wp(\texttt{S2}, Q))\Big)$$

- **Observación:** $wp(\texttt{b[i]:=E}, Q) \equiv wp(\texttt{b:=setAt(b,i,E)}, Q)$

¿Cuál es la precondición más débil?

$\{???\}$

```
while (x>0) do
  x := x -1
endwhile
```

$\{x = 0\}$

$$wp(\texttt{while ...}, x = 0) \equiv x \geq 0$$

¿Cuál es la precondición más débil?

$\{???\}$

```
i := 0;
while (x<5) do
  x := x + 1;
  i := i + 1
endwhile
```

$\{x = 5 \wedge i = 5\}$

$$wp(\texttt{i:=0; while ...}, x = 5 \wedge i = 5) \equiv x = 0$$

¿Cuál es la precondición más débil?

$\{???\}$

```
while (x==5) do
  x := 5
endwhile
```

$\{x \neq 5\}$

$$wp(\texttt{while ...}, x \neq 5) \equiv x \neq 5$$

¿Es válida la siguiente tripla de Hoare?

$\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

```
while (i <= n) do
  s := s + i;
  i := i + 1
endwhile
```

$\{s = \sum_{k=1}^{n} k\}$

Precondición más débil de un ciclo

- Supongamos que tenemos el ciclo **while B do S endwhile**.
- **Definición.** Definimos $H_k(Q)$ como el predicado que define el conjunto de estados a partir de los cuales la ejecución del ciclo termina en exactamente k iteraciones:

$$H_0(Q) \equiv \text{def}(B) \wedge \neg B \wedge Q,$$
$$H_{k+1}(Q) \equiv \text{def}(B) \wedge B \wedge wp(\mathtt{S}, H_k(Q)) \quad \text{para } k \geq 0.$$

- **Propiedad:** Si el ciclo realiza a lo sumo k iteraciones, entonces

$$wp(\mathtt{while\ B\ do\ S\ endwhile}, Q) \equiv \vee_{i=0}^{k} H_i(Q)$$

Ejemplo

$\{???\}$

```
while (0<i && i<3) do
  i := i +1
endwhile
```

$\{i = 3\}$

- A lo sumo, se va a ejecutar 2 veces el cuerpo del ciclo
- ¿Cuál es la precondición más débil?

$$\begin{aligned} & wp(\mathtt{while}\ 0 < i < 3\ \mathtt{do\ i{:=}i{+}1\ endwhile}, i = 3) \\ \equiv\ & \vee_{i=0}^{2} H_i(i = 3) \\ \equiv\ & H_0(i = 3) \vee H_1(i = 3) \vee H_2(i = 3) \\ \equiv\ & i = 1 \vee i = 2 \vee i = 3 \end{aligned}$$

Otro ejemplo

$\{???\}$

```
while (0<i && i<n) do
  i := i +1
endwhile
```

$\{i \geq 0\}$

- ¿Cuántas veces se va a ejecutar el cuerpo del ciclo?
- ¿Podemos usar la propiedad anterior para conocer la precondición más débil?
- ¡No! Porque no podemos fijar a priori una cota superior a la cantidad de iteraciones que va a realizar el ciclo.

Precondición más débil de un ciclo

- **Intituivamente:** $wp(\texttt{while B do S endwhile}, Q)$ tiene que ser una fórmula lógica capaz de capturar todos los estados tales que, luego de ejecutar el ciclo una cantidad arbitraria de veces, vale Q.
- **Axioma 5:**

$$wp(\texttt{while B do S endwhile}, Q) \equiv (\exists_{i\geq 0})(H_i(Q))$$

Precondición más débil de un ciclo

- Ahora tratemos de usar el **Axioma 5**:

$$\begin{aligned} & wp(\texttt{while B do S endwhile}, Q) \\ & \quad \equiv (\exists_{i\geq 0})H_i(Q) \\ & \quad \equiv H_0(Q) \vee H_1(Q) \vee H_2(Q) \vee \ldots \\ & \quad \equiv \vee_{i=0}^{\infty}(H_i(Q)) \end{aligned}$$

¡Es una fórmula infinitaria!

- Por lo tanto, no podemos usar mecánicamente el **Axioma 5** para demostrar la corrección de un ciclo con una cantidad no acotada a priori de iteraciones :(

Recap: Teorema del Invariante

- **Teorema.** Si def(B) y existe un predicado I tal que
  1. $P_C \Rightarrow I$,
  2. $\{I \wedge B\}\ S\ \{I\}$,
  3. $I \wedge \neg B \Rightarrow Q_C$,

  ... y **el ciclo termina**, entonces la siguiente tripla de Hoare es válida:

$$\{P_C\}\ \textbf{while B do S endwhile}\ \{Q_C\}$$

- Esta observación es un teorema que se deduce de la definición anterior.
- Las condiciones 1-3 garantizan la corrección parcial del ciclo (la hipótesis de terminación es necesaria para garantizar corrección).

Ejemplo: suma de índices

- Sea la siguiente tripla de Hoare:

$\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

```
while (i <= n) do
  s = s + i;
  i = i + 1;
endwhile
```

$\{s = \sum_{k=1}^{n} k\}$

- Habíamos identificamos los predicados necesarios para aplicar el Teorema del Invariante:
  - $P_C \equiv n \geq 0 \wedge i = 1 \wedge s = 0$
  - $Q_C \equiv s = \sum_{k=1}^{n} k$
  - $B \equiv i \leq n$
  - $I \equiv 1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{i-1} k$

$P_C \Rightarrow I$

$$\begin{aligned}
P_C &\equiv n \geq 0 \land i = 1 \land s = 0 \\
&\Rightarrow 1 \leq i \leq n+1 \land s = 0 \\
&\Rightarrow 1 \leq i \leq n+1 \land s = \sum_{k=1}^{0} k \\
&\Rightarrow 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \\
&\equiv I \checkmark
\end{aligned}$$

$I \land \neg B \Rightarrow Q_C$

$$\begin{aligned}
I \land \neg B &\equiv 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \land \neg(i \leq n) \\
&\equiv 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \land i > n \\
&\Rightarrow 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \land i = n+1 \\
&\Rightarrow 1 \leq i \leq n+1 \land s = \sum_{k=1}^{n+1-1} k \land i = n+1 \\
&\Rightarrow s = \sum_{k=1}^{n} k \equiv Q_C \checkmark
\end{aligned}$$

$\{I \land B\}\ S\ \{I\}$

Para demostrar $\{I \land B\}\ S\ \{I\}$ tenemos que probar que:

$$I \land B \Rightarrow wp(\mathtt{S}, I)$$

$$\begin{aligned}
& wp(\mathtt{s:=s+i;i:=i+1}, 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k) \\
\equiv\ & wp(\mathtt{s:=s+i}, wp(\mathtt{i:=i+1}, 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k)) \\
\equiv\ & wp(\mathtt{s:=s+i}, def(i+1) \land_L (1 \leq i+1 \leq n+1 \land s = \sum_{k=1}^{i+1-1} k)) \\
\equiv\ & wp(\mathtt{s:=s+i}, 1 \leq i+1 \leq n+1 \land s = \sum_{k=1}^{i+1-1} k) \\
\equiv\ & def(s+i) \land_L (1 \leq i+1 \leq n+1 \land s+i = \sum_{k=1}^{i+1-1} k)
\end{aligned}$$

$\{I \land B\}\ S\ \{I\}$

$$\begin{aligned}
&\equiv 0 \leq i \leq n \land s+i = \sum_{k=1}^{i} k \\
&\equiv 0 \leq i \leq n \land s = (\sum_{k=1}^{i} k) - i \\
&\equiv 0 \leq i \leq n \land s = \sum_{k=1}^{i-1} k
\end{aligned}$$

- Luego de simplificar, nos falta probar que:

$$\underbrace{(1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k}_{I} \land \underbrace{i \leq n}_{B}) \Rightarrow \underbrace{(0 \leq i \leq n \land s = \sum_{k=1}^{i-1} k)}_{wp(\mathtt{S}, I)}$$

- Lo cual es trivialmente cierto.
- Por lo tanto podemos concluir que $\{I \land B\}\ S\ \{I\}$ es una tripla de Hoare válida (i.e., verdadera)

Ejemplo: suma de índices

- Habiendo probado las hipótesis del Teorema del Invariante podemos decir que si el ciclo siempre termina, entonces la siguiente tripla de Hoare es válida:

$\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

```
while (i <= n) do
  s = s + i;
  i = i + 1;
endwhile
```

$\{s = \sum_{k=1}^{n} k\}$

- **Pero** ..., ¡no probamos que el ciclo termina!
- ¿Cómo podemos probar si dada una precondición, un ciclo siempre termina?
  - Para eso tenemos el Teorema de terminación.

Teorema de terminación de un ciclo

- **Teorema.** Sea $\mathbb{V}$ el producto cartesiano de los dominios de las variables del programa y sea I un invariante del ciclo **while B do S endwhile**. Si existe una función $fv : \mathbb{V} \to \mathbb{Z}$ tal que

  1. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
  2. $I \wedge fv \leq 0 \Rightarrow \neg B$,

  ... entonces la ejecución del ciclo **while B do S endwhile** siempre termina.

- La función fv se llama función variante del ciclo.

- El Teorema de terminación nos permite demostrar que un ciclo termina (i.e. no se cuelga).

Ejemplo: Suma de índices

- Sea la siguiente tripla de Hoare:

$\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

```
while (i <= n) do
  s = s + i;
  i = i + 1;
endwhile
```

$\{s = \sum_{k=1}^{n} k\}$

- Ya probamos que el siguiente predicado es un invariante de este ciclo.

$$I \equiv 1 \leq i \leq n + 1 \wedge s = \sum_{k=1}^{i-1} k$$

- ¿Cúal sería una buena función variante para este ciclo?

Ejemplo: Suma de índices

- Ejecutemos el ciclo con $n = 6$.

Iteración	i	s	n	n+1-i
0	1	0	6	6
1	2	1	6	5
2	3	3	6	4
3	4	6	6	3
4	5	10	6	2
5	6	15	6	1
6	7	21	6	0

- Una función variante representa una cantidad que se va reduciendo a lo largo de las iteraciones. En este caso es la cantidad de índices que falta sumar.

- Proponemos entonces $fv = n + 1 - i$

Ejemplo: Suma de índices

- Veamos que se cumplen las dos condiciones del teorema.

1. Para verificar que $\{I \wedge B \wedge fv = v_0\}$ S $\{fv < v_0\}$ para todo v_0, calculamos $wp(\mathtt{S}, fv < v_0)$.

$$\begin{aligned}
& wp(\mathtt{s:=s+1;i:=i+1}, fv < v_0) \\
& \equiv wp(\mathtt{s:=s+1;i:=i+1}, (n+1-i) < v_0) \\
& \equiv wp(\mathtt{s:=s+1}, wp(\mathtt{i:=i+1}, (n+1-i) < v_0)) \\
& \equiv wp(\mathtt{s:=s+1}, def(i+1) \wedge_L (n+1-(i+1)) < v_0)) \\
& \equiv wp(\mathtt{s:=s+1}, (n+1-(i+1)) < v_0)) \\
& \equiv def(s+1) \wedge_L n-i < v_0 \\
& \equiv n-i < v_0 \\
& \equiv n-i < n+1-i \\
& \equiv n-i < n-i+1 \checkmark
\end{aligned}$$

Ejemplo: Suma de índices

- Veamos que se cumplen las dos condiciones del teorema.

2. Verifiquemos que $I \wedge fv \leq 0 \Rightarrow \neg B$

$$\begin{aligned}
I \wedge fv \leq 0 \quad & \equiv \quad \overbrace{1 \leq i \leq n+1 \wedge \mathrm{s} = \sum_{k=1}^{i-1} k}^{I} \wedge \overbrace{n+1-i \leq 0}^{fv \leq 0} \\
& \Rightarrow \quad i \leq n+1 \wedge n+1-i \leq 0 \\
& \Rightarrow \quad i \leq n+1 \wedge n+1 \leq i \\
& \Rightarrow \quad i = n+1 \\
& \Rightarrow \quad \neg(i \leq n) \\
& \Rightarrow \quad \neg B \checkmark
\end{aligned}$$

Ejemplo: Suma de índices

Recapitulando, sean

- $I \equiv 1 \leq i \leq n+1 \wedge \mathrm{s} = \sum_{k=1}^{i-1} k$
- $fv \;=\; n+1-i$

Ya habíamos probado que el ciclo es **parcialmente** correcto dado que:

1. $P_C \Rightarrow I$
2. $\{I \wedge B\}$ S $\{I\}$
3. $I \wedge \neg B \Rightarrow Q_C$

Y además también acabamos de probar que el ciclo siempre termina ya que:

4. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
5. $I \wedge fv \leq 0 \Rightarrow \neg B$,

Por lo tanto, por (1)-(5) tenemos (finalmente) que ...

Ejemplo: Suma de índices

- Que la siguiente tripla de Hoare:

  $\{P_C : n \geq 0 \wedge i = 1 \wedge s = 0\}$
  ```
  while (i <= n) do
    s = s + i;
    i = i + 1;
  endwhile
  ```
  $\{Q_C : s = \sum_{k=1}^{n} k\}$

  ¡es una tripla de Hoare válida!

- Esto significa que:
  1. Si el ciclo comienza en un estado que cumple P_C
  2. ... entonces termina luego de un número finito de pasos
  3. y además en un estado que cumple Q_C

Otro ejemplo: Chequeo de paridad

Sea una secuencia de booleanos s, contar la cantidad de posiciones de la secuencia iguales a `true`.

Algunas propiedades de $\#apariciones$:

- $\#apariciones(\langle\rangle, true) = 0$
- $\#apariciones(concat(s, \langle e\rangle), true) =$ $\#apariciones(s, true) + (\text{if } e = true \text{ then } 1 \text{ else } 0 \text{ fi})$

Otro ejemplo: Chequeo de paridad

- ¿Es válida la siguiente tripla de Hoare?
- $\{i = 0 \land c = 0\}$

```
while( i < |s| ) do
   if s[i]=true then
     c := c + 1
   else
     skip
   endif;
   i := i +1
endwhile
```

$\{c = \#apariciones(s, true)\}$

Otro ejemplo: Chequeo de paridad

- Para probar que se cumplen las condiciones del Teorema del Invariante tenemos que demostrar formalmente que se cumple:
  1. $P_C \Rightarrow I$
  2. $\{I \land B\}$ S $\{I\}$
  3. $I \land \neg B \Rightarrow Q_C$
- ¿Cuál es el predicado P_C, Q_C, I y B?
  - $P_C \equiv i = 0 \land c = 0$
  - $Q_C \equiv c = \#apariciones(s, true)$
  - $B \equiv i < |s|$
  - $I \equiv 0 \leq i \leq |s| \land_L c = \#apariciones(subseq(s, 0, i), true)$

Recap #1: Teorema del invariante

- **Teorema.** Si def(B) y existe un predicado I tal que
  1. $P_C \Rightarrow I$,
  2. $\{I \land B\}\ S\ \{I\}$,
  3. $I \land \neg B \Rightarrow Q_C$,

  ... y **el ciclo termina**, entonces la siguiente tripla de Hoare es válida:

  $$\{P_C\}\ \textbf{while B do S endwhile}\ \{Q_C\}$$

Recap #2: Teorema de terminación de un ciclo

- **Teorema.** Sea $\mathbb{V}$ el producto cartesiano de los dominios de las variables del programa y sea I un invariante del ciclo **while B do S endwhile**. Si existe una función $fv : \mathbb{V} \to \mathbb{Z}$ tal que

  1. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
  2. $I \wedge fv \leq 0 \Rightarrow \neg B$,

  ... entonces la ejecución del ciclo **while B do S endwhile** siempre termina.

- La función fv se llama función variante del ciclo.

Teorema de corrección de un ciclo

- **Teorema.** Sean un predicado I y una función $fv : \mathbb{V} \to \mathbb{Z}$ (donde $\mathbb{V}$ es el producto cartesiano de los dominios de las variables del programa), y supongamos que $I \Rightarrow \text{def}(B)$. Si

  1. $P_C \Rightarrow I$,
  2. $\{I \wedge B\}\ S\ \{I\}$,
  3. $I \wedge \neg B \Rightarrow Q_C$,
  4. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
  5. $I \wedge fv \leq 0 \Rightarrow \neg B$,

  ... entonces la siguiente tripla de Hoare es válida:

  $$\{P_C\}\ \textbf{while B do S endwhile}\ \{Q_C\}$$

Teorema de corrección de un ciclo

- El **teorema de corrección de un ciclo** nos permite demostrar la validez de una tripla de Hoare cuando el programa es un ciclo.

- Por definición, si probamos que:

  $$\{P_C\}\ \text{while B do S endwhile}\ \{Q_C\}$$

  ... entonces probamos que:

  $$P_C \Rightarrow wp(\texttt{while B do S endwhile}, Q_C)$$

- **¡Cuidado!** Probar lo anterior no significa haber obtenido un predicado que caracteriza a la precondición más débil del ciclo:

  $$wp(\text{while B do S endwhile}, Q_C)$$

Programas con ciclos

- En general, no se puede definir un mecanismo efectivo para obtener una fórmula cerrada que represente la precondición más débil de un ciclo.

- Entonces, ¿cómo hacemos para probar la corrección y terminación de un programa que incluye ciclos intercalados con otras instrucciones?

Programas con ciclos

- Supongamos que tenemos la siguiente tripla de Hoare:

$\{Pre : n \geq 0\}$

```
s := 0;
i := 1;
while i<=n do
  s := s + i;
  i := i + 1
endwhile;
result := s
```

$\{Post : result = \sum_{k=1}^{n} k\}$

- Para demostrar que es válida necesitamos probar que es válida la fórmula:

$$\underbrace{n \geq 0}_{Pre} \Rightarrow wp(\texttt{s:=0;i:=1;while...;result:=s}, \overbrace{result = \sum_{k=1}^{n} k}^{Post})$$

Demostrando programas con ciclos

$$\begin{aligned} n \geq 0 \quad \Rightarrow & \quad wp(\texttt{s:=0;i:=1;while...;result:=s}, result = \sum_{k=1}^{n} k) \\ \equiv & \quad wp(\texttt{s:=0;i:=1;while...}, wp(\texttt{result:=s}, result = \sum_{k=1}^{n} k)) \\ \equiv & \quad wp(\texttt{s:=0;i:=1}, wp(\texttt{while...}, s = \sum_{k=1}^{n} k)) \end{aligned}$$

Como no podemos aplicar el Axioma 5 (no está acotado el número de iteraciones), ¿qué podemos hacer entonces?

Recap: Suma de índices

- Antes probamos que la siguiente tripla es válida:

$\{P_C : n \geq 0 \land i = 1 \land s = 0\}$

```
while (i <= n) do
  s = s + i;
  i = i + 1;
endwhile
```

$\{Q_C : s = \sum_{k=1}^{n} k\}$

- Por lo tanto, sabemos que se cumple que:

$$\underbrace{(n \geq 0 \land i = 1 \land s = 0)}_{P_C} \Rightarrow wp(\texttt{while...}, \underbrace{s = \sum_{k=1}^{n} k}_{Q_C})$$

Demostrando programas con ciclos

- Para poder usar que:

$$\underbrace{(n \geq 0 \land i = 1 \land s = 0)}_{P_C} \Rightarrow wp(\texttt{while...}, \underbrace{s = \sum_{k=1}^{n} k}_{Q_C})$$

- ... necesitamos probar que efectivamente el programa cumple $n \geq 0 \land i = 1 \land s = 0$ antes de que comience la ejecución del `while`.
- En otras palabras, necesitamos probar que:

$$\underbrace{n \geq 0}_{Pre} \Rightarrow wp(\texttt{s:=0;i:=1}, \underbrace{n \geq 0 \land i = 1 \land s = 0}_{P_C})$$

- ¿Es esto verdadero?

Demostrando programas con ciclos

$$\begin{array}{lcl} n \geq 0 & \Rightarrow & wp(\texttt{s:=0;i:=1}, n \geq 0 \wedge i = 1 \wedge s = 0) \\ & \equiv & wp(\texttt{s:=0}, wp(\texttt{i:=1}, n \geq 0 \wedge i = 1 \wedge s = 0)) \\ & \equiv & wp(\texttt{s:=0}, n \geq 0 \wedge 1 = 1 \wedge s = 0) \\ & \equiv & wp(\texttt{s:=0}, n \geq 0 \wedge s = 0) \\ & \equiv & n \geq 0 \wedge 0 = 0 \\ & \equiv & n \geq 0 \\ & \text{Verdadero} & \end{array}$$

Demostrando programas con ciclos

Por lo tanto, ya demostramos que:

$$\underbrace{n \geq 0}_{Pre} \Rightarrow wp(\texttt{s:=0;i:=1}, \underbrace{n \geq 0 \wedge i = 1 \wedge s = 0}_{P_C})$$

- A esto lo llamaremos **Lema 1**

Y además probamos que:

$$\underbrace{(n \geq 0 \wedge i = 1 \wedge s = 0)}_{P_C} \Rightarrow wp(\texttt{while...}, \underbrace{s = \sum_{k=1}^{n} k}_{Q_C})$$

- A esto otro lo llamaremos **Lema 2**

Entonces, ¿qué propiedad que vimos la clase pasada podemos usar?

¡Corolario de la monotonía de la wp!

Recap: Corolario de la monotonía de la wp

- **Corolario:** Si
  - $P \Rightarrow wp(\texttt{S1}, Q)$,
  - $Q \Rightarrow wp(\texttt{S2}, R)$,

  entonces
  - $P \Rightarrow wp(\texttt{S1;S2}, R)$.

Corolario de monotonía

- Tenemos que:
  1. $n \geq 0 \Rightarrow wp(\texttt{s:=0;i:=1}, n \geq 0 \wedge i = 1 \wedge s = 0)$,
  2. $(n \geq 0 \wedge i = 1 \wedge s = 0) \Rightarrow wp(\texttt{while...}, s = \sum_{k=1}^{n} k)$.
- Entonces, por el corolario de la monotonía de la wp podemos concluir que:

$$n \geq 0 \Rightarrow wp(\texttt{s:=0;i:=1;while...}, s = \sum_{k=1}^{n} k)$$

- A esta fórmula que es verdadera la llamaremos **Lema 3**:

Demostrando programas con ciclos

- Volviendo a lo que habíamos desarrollado anteriormente:

$$n \geq 0 \quad \Rightarrow \quad wp(\texttt{s:=0;i:=1;while...;result:=s}, result = \sum_{k=1}^{n} k)$$

$$n \geq 0 \quad \Rightarrow \quad wp(\texttt{s:=0;i:=1;while...}wp(\texttt{result :=s}, result = \sum_{k=1}^{n} k))$$

$$n \geq 0 \quad \Rightarrow \quad wp(\texttt{s:=0;i:=1}, wp(\texttt{while...}, s = \sum_{k=1}^{n} k))$$

- Ahora, por **Lema 3**, sabemos que:

$$\begin{array}{ccl} n \geq 0 & \Rightarrow & wp(\texttt{s:=0;i:=1}, wp(\texttt{while...}, s = \sum_{k=1}^{n} k)) \\ & \equiv & \text{Verdadero} \end{array}$$

Programas con ciclos

- Ya que probamos que es verdadero que:

$$\underbrace{n \geq 0}_{Pre} \Rightarrow wp(\texttt{s:=0;i:=1;while...;result:=s}, \overbrace{result = \sum_{k=1}^{n} k}^{Post})$$

- Entonces probamos que la siguiente tripla de Hoare es válida:

$\{Pre : n \geq 0\}$

```
s := 0;
i := 1;
while i<=n do
  s := s + i;
  i := i + 1
endwhile;
result := s
```

$\{Post : result = \sum_{k=1}^{n} k\}$

Guía para demostrar programas con ciclos

Cuando tenemos programas con ciclos: `S1;while...;S3`

1. Aplicamos los axiomas de *wp* hasta que que no podemos aplicar ninguno y obtenemos Q_C.
2. Probamos que la tripla de Hoare que contiene al ciclo es verdadera. Esto nos permite concluir que $P_C \Rightarrow wp(\texttt{while...}, Q_C)$.
3. Utilizamos el corolario de la monotonía de wp para concluir que $Pre \Rightarrow wp(\texttt{S1;while...}, Q_C)$ es verdadero.
4. Esto finalmente nos permite demostrar que $Pre \Rightarrow wp(\texttt{S1;while...;S3}, Post)$ es verdadera.

Recap: SmallLang

- Para las demostraciones de corrección, introdujimos un lenguaje sencillo y con menos opciones (mucho más simple que C++). Llamemos **SmallLang** a este lenguaje.
- SmallLang tiene únicamente:
  - Nada: `skip`
  - Asignación: `x := E`
  - Secuencia: `S1;S2`
  - Condicional: `if B then S1 else S2 endif`
  - Ciclo: `while B do S endwhile`
- No posee memoria dinámica (punteros), aliasing, llamados a función, estructura `for`, etc.

C++ → SmallLang

Pero dado un programa en C++ podemos traducirlo a SmallLang preservando su semántica (comportamiento).
Por ejemplo:

Versión C++

```
for (int i =0; i < s.size(); i++) {
  if (s[i]==0) {
    s[i]++;
  }
}
```

Versión SmallLang

```
i:=0;
while (i < s.size()) do
  if (s[i]==0)
    s[i]:=s[i]+1
  else
    skip
  endif;
  i:=i+1
endwhile
```

Ambos programas tienen el mismo comportamiento.

Corrección de programas en C++

Para demostrar la corrección de un programa en C++ con respecto a una especificación, podemos:

1. Traducir el programa C++ a SmallLang preservando su comportamiento.
2. Demostrar la corrección del programa en SmallLang con respecto a la especificación.
3. Entonces, probamos la corrección del comportamiento del programa original.

Bibliografía

- David Gries - The Science of Programming
  - Part II - The Semantics of a Small Language
    - Chapter 11 - The Iterative Command