Skip

- obligación de prueba: { P } **skip** { Q } si y sólo sí $P \Rightarrow Q$

- wp.**skip**.Q = Q

Asignación

- Obligación de prueba: { P } x≔E { Q } si y sólo sí $P \Rightarrow Q(x \rightarrow E)$

- wp.(x≔E).Q = Q(x→E)

Composición Secuencial

- Obligación de prueba: { P } c_1 ; c_2 { Q }
- proponer R tal que
1. { P } c_1 { R }
2. { R } c_2 { Q }

- wp.(c_1 ; c_2).Q = wp.c_1.(wp.c_2.Q)

Condicional

- Obligaciones de prueba { P } **if** $b_1 \rightarrow c_1$ [] $b_2 \rightarrow c_2$ **fi** { Q }
1. $P \Rightarrow b_1 \vee b_2$
2. $\{ P \wedge b_1 \}\ c_1\ \{Q\}$ y $\{ P \wedge b_2 \}\ c_2\ \{Q\}$

- wp.(**if** $b_1 \rightarrow c_1$ [] $b_2 \rightarrow c_2$ **fi**).Q = $(b_1 \vee b_2) \wedge (b_1 \Rightarrow$ wp.c_1.Q$) \wedge (b_2 \Rightarrow$ wp.c_2.Q$)$

Ciclos

- Obligaciones de prueba { P } **do** b → c **od** { Q }
- proponer *I* (invariante) tal que

0. $P \Rightarrow I$
1. $I \wedge \neg b \Rightarrow Q$
2. $\{ I \wedge b \}\ c\ \{ I \}$
3. *El ciclo termina*: existe una función de cota t : Estados → Int tal que:
   (i) $I \wedge b \Rightarrow t \geq 0$
   (ii) $\{ I \wedge b \wedge t = T \}\ c\ \{ t < T \}$

Estrategias para calcular Invariantes

1. Tomar términos de una conjunción

Ejemplo: buscar un elemento en un arreglo

Const N:Int; A : Array[0,N) of Int; e: Int;
Var k : Int;
{ P : $\langle \exists i : 0 \leq i < N : A.i = e \rangle$ }
S
{ Q : k = $\langle$ Min i : $0 \leq i < N \wedge A.i = e : i \rangle$ }

Para aplicar esta técnica, buscamos volver a escribir la postcondición de tal forma que sea una conjunción **(Q=Q')**

Q' : $(A.k = e) \wedge \langle \forall i : 0 \leq i < k : A.i \neq e \rangle \wedge (0 \leq k < N)$

A partir de esta Q', buscamos lo siguiente

- Invariantes posibles:
    - "**0≤k<N**", no es lo suficientemente fuerte. Sólo nos garantiza que no indexamos mal el arreglo.
    - "$\langle \forall$ **i : 0≤i<k : A.i ≠ e**$\rangle \wedge$ **0≤k<N**" , es lo suficientemente fuerte-
- Posibles guardas:
    - **A.k≠e**

Una vez que ya elegimos la guarda, el invariante es el resto de la conjunción.

2. Reemplazar constante por variable (cambiar una variable)

Ejemplo: Todos los elementos de un arreglo son iguales?

Const N:Int, A : Array [0,N) of Int;
Var todosIguales : Bool ;
{ **P** : True }
S_0 ;
{ I }
do B →
    S_1
od
{ **Q** : todosIguales = $\langle \forall i : 0 \leq i < N : A.i = A.0 \rangle$}

Esta técnica se usa cuando tenemos un Array, la técnica se basa en cambiar la variable y especificar el rango de la nueva variable tal que :

{ **Q** : todosIguales = $\langle \forall i : 0 \leq i < N : A.i = A.0 \rangle$}   ***de aca sacamos el invariante***

{ **I** : todosIguales = $\langle \forall i : 0 \leq i < n : A.i = A.0 \rangle \wedge 0 \leq n < N$}

Del Invariante deducimos que la **guarda** es n≠N

Estrategias para calcular Invariantes

3. Fortalecimiento de Invariantes (problema de bordes)

Esta técnica se utiliza en las derivaciones en las cuales nos encontramos algo que no sabemos qué es, en este caso, creamos una variable con el valor desconocido tal que se verifique el programa

Forma final de una derivación con bucle (plantear al final del ejercicio)

I = (...)
b = (...)
t = (...)

Declaración de variables	Const, Array, Var
{ P }	Pre-condición
S1	Inicialización
{ I]	Invariante
do b ->	Guarda
{ I ∧ b ∧ t=T}	Cota
S2	Asignación / Cuerpo del bucle
{ I ∧ t<T }	Cota decrece
{ Q }	Post-condición

Indicios para saber que necesitamos un ciclo

- La post-condición tiene una cuantificación (explícitamente o implícitamente).
- La post-condición menciona un arreglo.
- La post-condición menciona (implícita o explícitamente) una función definida por recursión.