```
ⓐ Var x : Int;
  [$\sigma_0 : (x \mapsto 1)$]
  x := 5;
  [$\sigma_1 : (x \mapsto 5)$]
```

```
ⓑ Var x,y : Int;
  [$\sigma_0 : (x \mapsto 2, y \mapsto 5)$]
  x := x + y;
  [$\sigma_1 : (x \mapsto 7, y \mapsto 5)$]
  x := y + y;
  [$\sigma_2 : (x \mapsto 7, y \mapsto 10)$]
```

```
ⓒ Var x,y : Int;
  [$\sigma_0 : (x \mapsto 2, y \mapsto 5)$]
  y := y + y;
  [$\sigma_1 : (x \mapsto 2, y \mapsto 10)$]
  x := x + y;
  [$\sigma_2 : (x \mapsto 12, y \mapsto 10)$]
```

```
ⓓ Var x,y : Int;
  [$\sigma_0 : (x \to 2, y \mapsto 5)$]
  y,x := y+y, x+y;
  [$\sigma_1 : (y \mapsto 10, x \mapsto 7)$]
```

② ⓐ

ⓑ El ⓒ se puede escribir como el ⓓ para tener una asignacion Multiple

```
③ ⓐ Var x,y : Int;
  [$\sigma_0 : (x \mapsto 3, y \mapsto 1)$]
  if x ≥ y →
      [$\sigma_1 : (x \mapsto 3, y \mapsto 1)$]
      x := 0;
      [$\sigma_2 : (x \to 0, y \mapsto 1)$]
  ~~□ x ≤ y →~~
      ~~[$\sigma_1'$ :                ]~~
      ~~x := 2;~~
      ~~[$\sigma_2'$ :                ]~~
  fi
  [$\sigma_3 : (x \to 0, y \mapsto 1)$]
```

```
ⓑ Var x,y : Int;
  [$\sigma_0 : (x \mapsto -100, y \mapsto 1)$]
  ~~if x ≥ y →~~
      ~~[$\sigma_1$ :            ]~~
      ~~x := 0;~~
      ~~[$\sigma_2$ :            ]~~
  □ x ≤ y
      [$\sigma_1' : (x \mapsto -100, y \mapsto 1)$]
      x := 2;
      [$\sigma_2' : (x \mapsto 2, y \mapsto 1)$]
  fi
  [$\sigma_3' : (x \mapsto 2, y \mapsto 1)$]
```

```
ⓒ Var x,y : Int;
  [$\sigma_0 : (x \mapsto 1, y \mapsto 1)$]
  if x ≥ y →
      [$\sigma_1 : (x \mapsto 1, y \mapsto 1)$]
      x := 0;
      [$\sigma_2 : (x \to 0, y \mapsto 1)$]
  ~~□ x ≤ y →~~
      ~~[$\sigma_1'$ :            ]~~
      ~~x := 2;~~
      ~~[$\sigma_2'$ :            ]~~
  fi
  [$\sigma_3 : (x \to 0, y \mapsto 1)$]
```

```
  [$\sigma_0 : (x \mapsto 1, y \mapsto 1)$]
  ~~if x ≥ y →~~
      ~~[$\sigma_1$ :            ]~~
      ~~x := 0;~~
      ~~[$\sigma_2$ :            ]~~
  □ x ≤ y →
      [$\sigma_1' : (x \mapsto 1, y \mapsto 1)$]
      x := 2;
      [$\sigma_2' : (x \mapsto 2, y \mapsto 1)$]
  fi
  [$\sigma_3' : (x \mapsto 2, y \mapsto 1)$]
```

④ ⓐ El programa no podria ejecutarse ya que no hay ningun caso que cumpla x=y

```
⑤ ⓐ Var i : Int:
  [$\sigma_0 : (i \mapsto 4)$]
  do i ≠ 0 →
      [$\sigma_1^1 : (i \mapsto 4), \sigma_1^2 : (i \mapsto 3), \sigma_1^3 : (i \mapsto 2), \sigma_1^4 : (i \mapsto 1)$]
      i := i - 1
      [$\sigma_2^1 : (i \mapsto 3), \sigma_2^2 : (i \mapsto 2), \sigma_2^3 : (i \mapsto 1), \sigma_2^4 : (i \mapsto 0)$]
  od
  [$\sigma_3 : (i \mapsto 0)$]
```

```
ⓑ Var i : Int;
  [$\sigma_0 : (i \mapsto 400)$]
  do i ≠ 0 →
      [$\sigma_1^1 : (i \mapsto 400)$]
      i := 0
      [$\sigma_2^1 : (i \mapsto 0)$]
  od
  [$\sigma_3 : (i \mapsto 0)$]
```

```
ⓒ Var i : Int
  [$\sigma_0 : (i \mapsto 4)$]
  do i < 0 →
      ~~[$\sigma_1^1$ :            ]~~
      ~~i := i - 1~~
      ~~[$\sigma_2^1$ :            ]~~
  od
  [$\sigma_3 : (i \mapsto 4)$]
```

```
ⓓ Var i : Int
  [$\sigma_0 : (i \mapsto 0)$]
  do i ≤ 0 →
      [$\sigma_1^1$ : ..., ... ]
      i := i - 1;
      [$\sigma_2^1$ : ..., ... ]
  od
  ~~[$\sigma_3$:            ]~~ programa no termina nunca
```

```
ⓔ Var r : Int
[$\sigma_0 : (r \mapsto 3)$]
do r ≠ 0 →
    if r < 0 →
        ~~[$\sigma_1^1$          ]~~
        ~~r := r + 1;~~
        ~~[$\sigma_2^1$          ]~~
    □ r > 0 →
        [$\sigma_3^1 : (r \mapsto 3), \sigma_3^2 : (r \mapsto 2), \sigma_3^3 : (r \mapsto 1)$]
        r := r - 1;
        [$\sigma_4^1 : (r \mapsto 2), \sigma_4^2 : (r \mapsto 1), \sigma_4^3 : (r \mapsto 0)$]
    fi
    [$\sigma_5^1 : (r \mapsto 2), \sigma_5^2 : (r \mapsto 1), \sigma_5^3 : (r \mapsto 0)$]
od
[$\sigma_6 : (r \mapsto 0)$]
```

⑥

7 a

```
{N ≥ 0} S {r = N!}
var r, x : Int
r := 1 ;
x := N ;
do x > 0 ⟶
    r := r * x ;
    x := x - 1 ;
od
```

b

```
{N ≥ 0} S {r = N!}
var r, x : Int
r := 1 ;
x := 1 ;
do x ≤ N ⟶
    r := r · x
    x := x+1
od
```

c

```
[$\sigma_0$ : (r ↦ 1, x ↦ 5)]
do x > 0 ⟶
    [$\sigma_1^1$ : (r ↦ 1, x ↦ 5), $\sigma_1^2$ : (r ↦ 5, x ↦ 4), $\sigma_1^3$ (r ↦ 20, x ↦ 3), $\sigma_1^4$ (r ↦ 60, x ↦ 2), $\sigma_1^5$ (r ↦ 120, x ↦ 1)]
    r := r * x ;
    x := x - 1 ;
    [$\sigma_2^1$ : (r ↦ 5, x ↦ 4), $\sigma_2^2$ : (r ↦ 20, x ↦ 3), $\sigma_2^3$ (r ↦ 60, x ↦ 2), $\sigma_2^4$ (r ↦ 120, x ↦ 1), $\sigma_2^5$ (r ↦ 120, x ↦ 0)
od
[$\sigma_3$ : (r ↦ 120, x ↦ 0)]
```

```
[$\sigma_0$ : (r ↦ 1 , x ↦ 1)]
do x ≤ 5 ⟶
    [$\sigma_1^1$ : (r ↦ 1 , x ↦ 1), $\sigma_1^2$ : (r ↦ 1, x ↦ 2), $\sigma_1^3$ : (r ↦ 2, x ↦ 3), $\sigma_1^4$ (r ↦ 6 , x ↦ 4), $\sigma_1^5$ : (r ↦ 24, x ↦ 5)
    r := r · x
    x := x+1
    [$\sigma_2^1$ : (r ↦ 1 , x ↦ 2), $\sigma_2^2$ : (r ↦ 2, x ↦ 3), $\sigma_1^3$ (r ↦ 6, x ↦ 4), $\sigma_2^4$ (r ↦ 24, x ↦ 5), $\sigma_2^5$ : (r ↦ 120, x ↦ 6)
od
[$\sigma_3$ : r ↦ 120 , x ↦ 6]
```

8 a

```
Const A: array [0, 4) of Int;
Var i,s : Int;
[ $\sigma_0$ : (i ↦ -3, s ↦ 5 , A ↦ |2|10|10|-1| )]
i,s := 0,0 ;
[$\sigma_0'$]
do i < 4 ⟶
    [$\sigma_1^0$, ..., $\sigma_1^3$]
    s,i := s + A.i , i+1
    [$\sigma_2^0$, ..., $\sigma_2^3$]
od
[$\sigma_3$]
```

Estado	i	s
σ_0	-3	5
σ_0'	0	0
σ_1^0	0	0
σ_2^0	2	1
σ_1^1	2	1
σ_2^1	12	2
σ_1^2	12	2
σ_2^2	22	3
σ_1^3	22	3
σ_2^3	21	4
σ_3	21	4

b

```
Const A: array [0, 4) of Int ;
Var i, c : Int;
[ $\sigma_0$ : (i ↦ 3 , c ↦ 12, A ↦ |12|-9|10|-1| )]
i, c := 0, 0;
[$\sigma_0'$]
do i < 4 ⟶
    [$\sigma_1^0$, ..., $\sigma_1^3$]
    if A.i > 0 ⟶
        c := c+1
    [] A.i ≤ 0 ⟶
        skip.
    fi
    [$\sigma_2^0$, ..., $\sigma_2^3$]
    i := i+1
    [$\sigma_3^0$, ..., $\sigma_3^3$]
od
[$\sigma^4$]
```

Estado	i	A.i	c
σ_0	3	-1	12
σ_0'	0	12	0
σ_0^1	0	12	0
σ_0^2	0	12	1
σ_0^3	1	-9	1
σ_1^1	1	-9	1
σ_1^2	1	-9	1
σ_1^3	2	10	1
σ_2^1	2	10	1
σ_2^2	2	10	2
σ_2^3	3	-1	2
σ_3^1	3	-1	2
σ_3^2	3	-1	2
σ_3^3	4	/	2
σ_4	4	/	2

9 a agregoria un p:= s/i cuando termine el do

b agregria una condicion en el do i % 2 ==0

10 a

```
Const A: array [0, N);
Var i, s, sm3;
i, s, sm3 := 0,0,0
[$\sigma_0$]
do i < N
    [$\sigma_1^0$, ... $\sigma_1^4$]
    if A.i > 0
        s = s + A.i
    fi
    [$\sigma_2^0$, ... , $\sigma_2^4$]
    if A.i % 3 == 0
        sm3 = sm3 + A.i
    fi
    i = i + 1
    [ $\sigma_3^0$, ... , $\sigma_3^4$ ]
od
[$\sigma_4$]
```

b N = 5, A = |12|-9|10|0|-1|

Estados	i	A.i	s	sm3
σ_0	0	12	0	0
σ_1^0	0	12	0	0
σ_2^0	0	12	12	0
σ_3^0	1	-9	12	12
σ_1^1	1	-9	12	12
σ_2^1	1	-9	12	12
σ_3^1	2	10	12	3
σ_1^2	2	10	12	3
σ_2^2	2	10	22	3
σ_3^2	3	0	22	3
σ_4^3	3	0	22	3
σ_2^3	3	0	22	3
σ_3^3	4	-1	22	3
σ_1^4	4	-1	22	3
σ_2^4	4	-1	22	3
σ_3^4	5	-1	22	3
σ_4	5	/	22	3

11 a

b

12 a Var x: Int;

$\{x > 0\}$

$x := x \cdot x$

$\{True\}$

$\{x>0\}\ S\ \{True\}$

$\equiv \{TH\ y\ WP\}$

$x>0 \Rightarrow wp.(x := x * x).\ True$

$\equiv \{Asignacion\}$

$x>0 \Rightarrow True$

$\equiv \{Logica\}$

True

b Var x: Int ;

$\{x \neq 100\}$

$x := x * x$

$\{x \geq 0\}$

$\{x \neq 100\}\ S\ \{x \geq 0\}$

$\equiv \{TH\ y\ WP\}$

$(x \neq 100) \Rightarrow wp.(x := x * x).(x \geq 0)$

$\equiv \{Asignacion\}$

$(x \neq 100) \Rightarrow (x * x \geq 0)$

$\equiv \{Logica\}$

$(x \neq 100) \Rightarrow True$

$\equiv \{Logica\}$

True

c Var x: Int;

$\{x>0 \wedge x < 100\}$

$x := x * x$

$\{x \geq 0\}$

$\{x>0 \wedge x<100\}\ S\ \{x \geq 0\}$

$\equiv \{TH\ y\ WP\}$

$(x>0 \wedge x<100) \Rightarrow wp(x := x * x).(x \geq 0)$

$\equiv \{Asignacion\}$

$(x>0 \wedge x<100) \Rightarrow x * x \geq 0$

$\equiv \{Logica\}$

$(x>0 \wedge x<100) \Rightarrow True$

$\equiv \{Logica\}$

True

d Var x: Int :

$\{x>0\}$

$x := x * x$

$\{x<0\}$

$\{x>0\}\ S\ \{x<0\}$

$\equiv \{TH\ y\ WP\}$

$(x>0) \Rightarrow wp.(x := x * x).(x<0)$

$\equiv \{Asignacion\}$

$(x>0) \Rightarrow (x * x < 0)$

$\equiv \{Logica\}$

$(x>0) \Rightarrow False$

No es valida la TH, $(x>0) \Rightarrow False$ no es valida, si $x=1$ la formula es falsa.

e Var x: Int;

$\{x<0\}$

$x := x * x$

$\{x<0\}$

$\{x<0\}\ S\ \{x<0\}$

$\equiv \{TH\ y\ WP\}$

$(x<0) \Rightarrow wp.(x := x * x).(x<0)$

$\equiv \{Asignacion\}$

$(x<0) \Rightarrow (x * x < 0)$

f $\{True\}\ S\ \{x \geq 0\}$

$\equiv \{T.H\ y\ W.P\}$

$True \Rightarrow wp.(x := x * x).(x \geq 0)$

$\equiv \{Asignacion\}$

$True \Rightarrow (x * x \geq 0)$

$\equiv \{Logica\}$

$True \Rightarrow True$

$\equiv \{Logica\}$

True

13

(14) (a) Var x, y : Num ;

$\{ \quad \}$

$x := x+y$

$\{x = 6 \wedge y=5\}$

$\{P\}\ S\ \{x=6 \wedge y=5\}$

$\equiv$ {TH y WP}

$P \Rightarrow wp.(x:=x+y).(x=6 \wedge y=5)$

$\equiv$ {Asignacion}

$P \Rightarrow (x+y=6 \wedge y=5)$

$\equiv$ {Leibniz}

$P \Rightarrow (x+5=6 \wedge y=5)$

$\equiv$ {Aritmetrica}

$P \Rightarrow (x=1 \wedge y=5)$

WP es $x=1 \wedge x=5$

$\{x=1 \wedge y=5\}\ S\ \{x=6 \wedge y=5\}$

(b) $\{ \quad \}$

$x := 8$

$\{x = 8\}$

$\{P\}\ S\ \{x=8\}$

$\equiv$ {TH y WP}

$P \Rightarrow wp.(x:=8).(x=8)$

$\equiv$ {Asignacion}

$P \Rightarrow 8=8$

$\equiv$ {Logica}

$P \Rightarrow True$

WP es True

$\{True\}\ S\ \{x=8\}$

(c) $\{ \quad \}$

$x := 8$

$\{x = 7\}$

$\{P\}\ S\ \{x=7\}$

$\equiv$ {TH y WP}

$P \Rightarrow wp(x:=8).(x=7)$

$\equiv$ {Asignacion}

$P \Rightarrow (8=7)$

$\equiv$ {Logica}

$P \Rightarrow False$

La WP es False

$\{False\}\ S\ \{x=8\}$

(d) Var x, y : Num ;

$\{ \quad \}$

$x, y := y, x$

$\{x=6 \wedge y=5\}$

$\{P\}\ S\ \{x=B \wedge y=A\}$

$\equiv$ {TH y WP}

$P \Rightarrow wp.(x, y := y, x)(x=B \wedge y=A)$

$\equiv$ {Asignacion}

$P \Rightarrow (y=B \wedge x=A)$

La WP es $y=B \wedge x=A$

$\{y=B \wedge x=A\}\ S\ \{x=B \wedge y=A\}$

(e) Var x, y, a : Num ;

$\{ \quad \} \rightarrow \{y=B \wedge x=A\}$

$a, x := x, y$;

$\{ \quad \} \longrightarrow \{x=B \wedge a=A\}$

$y := a$

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

$\{P\}\ S;T\ \{x=B \wedge y=A\}$

$\equiv$ {Sec, Comp o Concat}

$P \Rightarrow wp(a, x := x,y\ ;\ y:=a).(x=B \wedge y=A)$

$\equiv$ {Def}

$P \Rightarrow wp(a, x := x, y).wp(y:=a).(x=B \wedge y=A)$

$\equiv$ {Asignacion}

$P \Rightarrow wp(a, x := x, y).(x=B \wedge a=A)$

$\equiv$ {Asignacion}

$P \Rightarrow y=B \wedge x=A$

La WP es $y=B \wedge x=A$

$\{y=B \wedge x=A\}\ S;T\ \{x=B \wedge y=A\}$

(f) $\{y=1\}$

if $x \geq y \longrightarrow$

$\{②\}$

$x := 0$

$\{③\}$

$\square\ x \leq y \longrightarrow$

$\{④\}$

$x := 2$

$\{⑤\}$

fi

$\{(x=0 \vee x=2) \wedge y=1\}$

$\{P\}\ S;T\ \{(x=0 \vee x=2) \wedge y=1\}$

$((x \geq y) \vee (x \leq y)) \wedge ((x \geq y) \Rightarrow wp.(x:=0).((x=0 \vee x=2) \wedge y=1)$

$\wedge ((x \leq y) \Rightarrow wp.(x:=2).((x=0 \vee x=2) \wedge y=1)$

$\equiv$ {WP}

$((x \geq y) \vee (x \leq y)) \wedge ((x \geq y) \Rightarrow ((0=0 \vee 0=2) \wedge y=1))$

$\wedge ((x \leq y) \Rightarrow ((2=0 \vee 2=2) \wedge y=1))$

$\equiv$ {Logica}

$((x \geq y) \vee (x \leq y) \wedge ((x \geq y) \Rightarrow True \wedge y=1) \wedge$

$((x \leq y) \Rightarrow True \wedge y=1)$

$\equiv$ {Logica}

$((x \geq y) \vee (x \leq y)) \wedge ((x \geq y) \Rightarrow y=1) \wedge ((x \leq y) \Rightarrow y=1)$

$\equiv$ {Caracterizacion}

$((x \geq y) \vee (x \leq y)) \wedge (\neg(x \geq y) \vee y=1) \wedge (\neg(x \leq y) \vee y=1)$

$\equiv$ {distributibidad

$((x \geq y) \vee (x \leq y)) \wedge (\neg(x \geq y) \wedge \neg(x \leq y)) \vee y=1$

$\equiv$ {Logica}

$True \wedge ((x \leq y) \wedge (x \geq y)) \vee y=1$

$\equiv$ {Logica} $\longrightarrow$ $y=1$

(15) (a) $\{True\}$

```
if x≥1 ⟶ x := x+1
□ x≤1 ⟶ x:= x-1
fi
```

$\{x \neq 1\}$

$$True \Rightarrow ((x \geq 1) \vee (x \leq 1)) \wedge \quad \{x \geq 1 \wedge True\}.(x:=x+1).\{x \neq 1\}$$
$$\wedge \quad \{x \leq 1 \wedge True\}.(x:= x-1).\{x \neq 1\}$$

$x \geq 1 \Rightarrow wp.(x:x+1).(x \neq 1)$
$\equiv \{WP \text{ y } Asig\}$
$x \geq 1 \Rightarrow x+1 \neq 1$

$x \leq 1 \Rightarrow wp.(x:x-1).(x \neq 1)$
$\equiv \{WP \text{ y } Asig\}$
$x \leq 1 \Rightarrow x \dot{-} 1 \neq 1$

$$((x \geq 1) \vee (x \leq 1)) \wedge (x \geq 1 \Rightarrow x+1 \neq 1) \wedge (x \leq 1 \Rightarrow x \dot{-} 1 \neq 1)$$
$\equiv \{logica\}$
$$(x \geq 1 \Rightarrow x+1 \neq 1) \wedge (x \leq 1 \Rightarrow x \dot{-} 1 \neq 1)$$
$\equiv \{Logica\}$
$$True \wedge True$$
$\equiv \{Conj\}$
$$True$$

⑯ ⓐ La post condicion indica que la poscicion k del arreglo es la sumatoria de todos los elementos previos a K, hata N-1.
El programo no es valido ya que nunca se incrementa i.

ⓑ La post condicion indica que la poscicion k del arreglo es la sumatoria de todos los elementos previos a K, hasta N-1.
Valida las anotaciones

ⓒ La post condicion indica que la poscicion k del arreglo es la sumatoria de todos los elementos previos a K, hasta N-1.
No valida ya que la poscicion -1 del array no existe (numero aleatorio)

ⓓ La postcondicion comprueba que existe un elemento en la poscicion k del arreglo y que esto implica que la poscicion i=K+1
tambien es igual a E
La anotacion no es valida

Faltan Extras (17, 18, 19, 20) y 14 f, 15