Verificación de programas (clases 12 y 13)

Ejercicio 1. Se define la postcondición más fuerte de la siguiente manera:

$$post(p, S) = \{\sigma' \mid \exists \sigma : \sigma \models p \wedge val(\pi(S, \sigma)) = \sigma' \neq \bot\}$$

es decir que un estado está en post(p, S) si es el estado final de una computación finita de S que arranca desde un estado inicial que satisface p. Y se define la precondición liberal más débil de la siguiente manera:

$$pre(S, q) = \{\sigma \mid \forall \sigma': val(\pi(S, \sigma)) = \sigma' \neq \bot \longrightarrow \sigma' \models q\}$$

es decir que un estado está en pre(S, q) si es el estado inicial a partir del cual se obtiene, por la ejecución de S, si termina, un estado final que satisface q. Probar:

a) $\models \{p\}\ S\ \{q\} \leftrightarrow post(p, S) \subseteq \{\sigma \mid \sigma \models q\}$
b) $\models \{p\}\ S\ \{q\} \leftrightarrow \{\sigma \mid \sigma \models p\} \subseteq pre(S, q)$

Ejercicio 2. En la clase práctica anterior se probó usando el método H:

$$\{x \geq 0 \wedge y > 0\}\ S_{div} :: q := 0; r := x; \text{while } r \geq y \text{ do } r := r - y; q := q + 1 \text{ od } \{x = q \cdot y + r \wedge 0 \leq r < y\}$$

siendo S_{div} un programa que calcula por restas sucesivas la división entera de x sobre y en q, dejando el resto en r. Se pide ahora probar en H:

$$\{x > 0 \ \wedge \ y = 0\}\ S_{div}\ \{false\}$$

es decir que el programa S_{div} no termina a partir de la precondición ($x > 0 \ \wedge \ y = 0$).

Ejercicio 3. Probar:

$$\langle x \geq 0 \wedge y \geq 0\rangle\ S_{prod} :: prod := 0; k := y ; \text{while } k > 0 \text{ do } prod := prod + x ; k := k - 1 \text{ od } \langle true\rangle$$

<u>Ayuda</u>: S_{prod} calcula en la variable prod el producto entre x e y. Notar que k se decrementa en cada iteración y que se mantiene siempre mayor o igual que cero.

Ejercicio 4. Probar la sensatez de la regla de invariancia vista en clase:

$$\frac{\{p\}\ S\ \{q\}}{\{r \wedge p\}\ S\ \{r \wedge q\}}$$

cuando las variables libres de r son disjuntas con las variables modificables por S.
<u>Ayuda</u>: Utilizar inducción sobre la longitud de las pruebas, como hicimos en clase.

Ejercicio 5. Probar sin recurrir a la completitud de H (es decir que la prueba debe ser sintáctica) que para todo programa S y toda aserción q se cumple:

$$Tr \vdash \{false\}\ S\ \{q\}$$

<u>Ayuda</u>: Utilizar inducción estructural sobre la forma de los programas S, similar a lo visto en clase para probar sintácticamente la fórmula {true} S {true}.