2.

Para probar que -> es determinista, es decir, si t->t’ y t->t’’, entonces t’=t’’; usamos inducción sobre la derivación t->t’.

\*Si la última regla utilizada es ASS, luego t tiene la forma ‹v:=e,σ› . Pero entonces, esta es la única regla que se puede aplicar y, por lo tanto, t’=[ σ|v:[[e]]intExp σ]=t’’. ( [[e]]intExp σ es el valor de una función)

\* Si la última regla utilizada es SKIP, luego t tiene la forma ‹**skip**,σ› . Pero entonces, esta es la única regla que se puede aplicar y, por lo tanto, t’= σ = t’’.

\*\*Si la última regla utilizada es SEQ1, luego t tiene la forma ‹c0;c1,σ›. Por lo tanto, la única otra regla que se podría aplicar es SEQ2, pero como ‹c0,σ› -> σ’ tenemos que esta es la única regla que se puede aplicar. Ahora, t’=‹c1,σ’› (‹c0,σ› -> σ’) y t’’=‹c1,σ’’› (‹c0,σ› -> σ’’), pero por HI si ‹c0,σ› -> σ’ y ‹c0,σ› -> σ’’ entonces σ’= σ’’, y por lo tanto, t’=t’’.

\*Si la última regla utilizada es IF1, luego t tiene la forma ‹**if** b **then** c0 **else** c1,σ›. Por lo tanto, la única otra regla que se podría aplicar es IF2, pero como [[b]]boolexp -> **true** tenemos que esta es la única regla que se puede aplicar. Luego, t’=‹c0,σ›=t’’.

\* Si la última regla utilizada es WHILE1, luego t tiene la forma ‹**while** b  **do** c,σ›. Por lo tanto, la única otra regla que se podría aplicar es WHILE2, pero como [[b]]boolexp -> **true** tenemos que esta es la única regla que se puede aplicar. Luego, t’=‹c;**while** b  **do** c,σ›=t’’.

\*Si la última regla utilizada es SEQ2, IF2 o WHILE2, el razonamiento es análogo.

GRAMATICA ABSTRACTA: