```
        struct exp *ret_exp; /* espressione di cui ritornare il valore */
        /* blocco */
        struct ls_commands *block; /* blocco di comandi */
    } u;
};
```

La struttura per la memorizzazione dei comandi contiene un tag di tipo `enum command_tag` il cui valore indica il tipo del comando da memorizzare e pertanto quale componente della successiva union è significativa.

La union `u` di `struct command` viene quindi utilizzata nel seguente modo:

1. se il comando è un'assegnazione, la struct `ass` contiene nel campo `lhs` l'indice e il tipo (locale o globale) della variabile alla sinistra dell'assegnazione e nel campo `rhs` il puntatore all'espressione alla destra dell'assegnazione;

2. se il comando è un if o un if-else, la struct `ite` contiene nel campo `cond` l'espressione condizionale dell'if-else, nel campo `then_c` il comando corrispondente al ramo then e nel campo `else_c` il comando corrispondente al ramo else (il caso di comando if senza else viene gestito mantenendo pari a `NULL` il campo `else_c`);

3. se il comando è un ciclo while, la struct `loop` coontiene nel campo `cond` l'espressione condizionale che controlla il ciclo e nel campo `body` il comando corrispondente al corpo del ciclo;

4. se il comando è una lettura da input, il campo `read_var` contiene l'indice e il tipo (locale o globale) della variabile da leggere;

5. se il comando è una scrittura (di un dato, di spazi o di new-line), il campo `wr_exp` contiene l'espressione di cui stampare il valore o corrispondente agli spazi o ai new-line da stampare;

6. se il comando è un return, il campo `ret_exp` contiene l'espressione di cui ritornare il valore;

7. se il comando è un blocco, il campo `block` contiene la lista dei comandi che formano il blocco.

Alcune considerazioni sulle parentesi nei comandi e nelle espressioni

La sintassi di CCINO è molto restrittiva nell'uso delle parentesi. Ad esempio, non si può prendere una generica espressione e racchiuderla tra una coppia di parentesi tonde, ovvero, (x) non è un'espressione corretta. Invece, ogni espressione ottenuta mediante un operatore binario deve essere racchiusa tra una coppia di parentesi tonde. Ad esempio, $x > 0$ e $(x+y+z)$ non sono espressioni corrette; le corrispondenti espressioni corrette sono $(x > 0)$ e $((x + y) + z)$ (oppure $(x + (y + z))$ se si associa a destra anzichè a sinistra).

Questa definizione della sintassi semplifica l'implementazione dell'analizzatore sintattico, ma appesantsce la scrittura dei comandi CCINO. Ad esempio,

$$\textbf{if } (x > 0)\ x = (x - 1);$$

non è un comando corretto. Infatti, in base alla sintassi di CCINO, l'if deve essere seguito da un'espressione valida racchiusa tra parentesi, quindi, rimosse le parentesi richieste dalla sintassi dell'if, $x > 0$ dovrebbe essere un'espressione corretta, ma in base a quanto detto precedentemente, tale espressione sarebbe corretta solo se raccchiusa tra parentesi tonde. Concludendo, la forma coretta del precedente comando è

$$\textbf{if } ((x > 0))\ x = (x - 1);$$

dove $x > 0$ è racchiuso tra una doppia coppia di parentesi tonde: una coppia richiesta dalla sintassi dell'if ed una coppia richiesta dalla sintassi delle espressioni.

Lo stesso ragionamento vale in tutti gli altri casi in cui la sintassi del comando richiede un'espressione racchiusa tra parentesi.