Progetto LC2, Parte 3
Gruppo 1

Assunzioni

- Non è possibile dichiarare più di una funzione con lo stesso identificativo.
- non è consientito dichiarare delle variabili con lo stesso identificativo nello stesso blocco, mentre è possibile se avviene su blocchi (e sotto-blocchi) diversi;
- è stato usato l'operatore '!' nelle condizioni degli 'if per veicolare correttamente i salti condizionati;
- nel corpo delle procedure non deve presentare l'istruzione return, sempre richiesta invece nei corpi delle funzioni;
- sono stati mantenuti solamente gli usuali comandi per il controllo della sequenza (*condizionali semplici*, *iterazione indeterminata*);
- le guardie booleane dei controlli di sequenza vengono gestite tramite short-cut mentre le altre espressioni booleani vengono gestite senza short-cut.

Scelte implementative

- Abbiamo generato *lexer* e *parser* tramite il tool BNFC, partendo da una gramatica iniziale;
- la gestione del *Type Checker* e del *Three Address Code* vengono fatte all'interno del parser;
- le funzioni *write* sono trattate come statements che prendendono come argomento una right expression, mentre le funzioni *read* vengono viste come right expressions e hanno una lista di parametri vuota.

Type Checker

Il Type Checker viene gestito all'interno del parser appoggiandosi ad un modulo esterno (*nome-modulo*) contenente una serie di funzioni di controllo sull'environment. Il Type Checker tiene conto in modo automatico di una conversione dal tipo int al tipo float.

Three Address Code

Il Three Address Code viene generato interamente nel parser, utilizzando un modulo esterno (*nome-modulo*) contenente la struttura dati per gestirne la generazione ed una funzione addetta alla stampa. Il Three Address Code viene rappresentato dalla

Test Case

Sono stati preparati dei test-case significativi che possono essere eseguiti in sequenza attraverso il comando *make demo*, oppure singolarmente attraverso il comando *make demo1* ... *make demo3*.