Contents

1 Esercizio 3.2

1.1 Calcolo di NULL, FIRST e FOLLOW

	NULL	FIRST	FOLLOW
⟨prog⟩		{ `assign`, `print`, `read`, `for`, `if`, `{` }	{ `$` }
⟨statlist⟩		{ `assign`, `print`, `read`, `for`, `if`, `{` }	{ `EOF`, `}` }
⟨statlistp⟩	x	{ `;` }	{ `EOF`, `}` }
⟨stat⟩		{ `assign`, `print`, `read`, `for`, `if`, `{` }	{ `;`, `else`, `end`, `EOF`, `}` }
⟨assignlist⟩		{ `[` }	{ `;`, `else`, `end`, `EOF`, `}` }
⟨assignlistp⟩	x	{ `[` }	{ `;`, `else`, `end`, `EOF`, `}` }
⟨idlist⟩		{ `ID` }	{ `)`, `]` }
⟨idlistp⟩	x	{ `,` }	{ `)`, `]` }
⟨bexpr⟩		{ `<`, `>`, `<=`, `>=`, `==`, `<>` }	{ `)` }
⟨expr⟩		{ `+`, `-`, `*`, `/`, `NUM`, `ID` }	{ `,`, `;`, `to`, `+`, `-`, `*`, `/`, `)`, `NUM`, `ID` }
⟨exprlist⟩		{ `+ -`, `*`, `/`, `NUM`, `ID` }	{ `)` }
⟨exprlistp⟩	x	{ `,` }	{ `)` }

1.2 Calcolo degli insiemi GUIDA

GUIDA($\langle prog \rangle \to \langle statlist \rangle$ EOF)	{ assign, print, read, for, if, { }
GUIDA($\langle statlist \rangle \to \langle stat \rangle \langle statlistp \rangle$)	{ assign, print, read, for, if, { }
GUIDA($\langle statlistp \rangle \to$; $\langle stat \rangle \langle statlistp \rangle$)	{ ; }
GUIDA($\langle statlistp \rangle \to \varepsilon$)	{ EOF, } }
GUIDA($\langle stat \rangle \to$ assign $\langle assignlist \rangle$)	{ assign }
GUIDA($\langle stat \rangle \to$ print ($\langle exprlist \rangle$))	{ print }
GUIDA($\langle stat \rangle \to$ read ($\langle idlist \rangle$))	{ read }
GUIDA($\langle stat \rangle \to$ for (ID := $\langle expr \rangle$; $\langle bexpr \rangle$) do $\langle stat \rangle$)	{ for }
GUIDA($\langle stat \rangle \to$ for ($\langle bexpr \rangle$) do $\langle stat \rangle$)	{ for }
GUIDA($\langle stat \rangle \to$ if ($\langle bexpr \rangle$) $\langle stat \rangle$ else $\langle stat \rangle$ end)	{ if }
GUIDA($\langle stat \rangle \to$ if ($\langle bexpr \rangle$) $\langle stat \rangle$ end)	{ if }
GUIDA($\langle stat \rangle \to$ { $\langle statlist \rangle$ })	{ { }
GUIDA($\langle assignlist \rangle \to$ [$\langle expr \rangle$ to $\langle idlist \rangle$] $\langle assignlistp \rangle$)	{ [}
GUIDA($\langle assignlistp \rangle \to$ [$\langle expr \rangle$ to $\langle idlist \rangle$] $\langle assignlistp \rangle$)	{ [}
GUIDA($\langle assignlistp \rangle \to \varepsilon$)	{ ;, else, end, EOF, } }
GUIDA($\langle idlist \rangle \to$ ID $\langle idlistp \rangle$)	{ ID }
GUIDA($\langle idlistp \rangle \to$, ID $\langle idlistp \rangle$)	{ , }
GUIDA($\langle idlistp \rangle \to \varepsilon$)	{),] }
GUIDA($\langle bexpr \rangle \to$ < $\langle expr \rangle \langle expr \rangle$)	{ < }
GUIDA($\langle bexpr \rangle \to$ > $\langle expr \rangle \langle expr \rangle$)	{ > }
GUIDA($\langle bexpr \rangle \to$ <= $\langle expr \rangle \langle expr \rangle$)	{ <= }
GUIDA($\langle bexpr \rangle \to$ <= $\langle expr \rangle \langle expr \rangle$)	{ <= }
GUIDA($\langle bexpr \rangle \to$ == $\langle expr \rangle \langle expr \rangle$)	{ == }
GUIDA($\langle bexpr \rangle \to$ <> $\langle expr \rangle \langle expr \rangle$)	{ <> }
GUIDA($\langle expr \rangle \to$ + ($\langle exprlist \rangle$))	{ + }
GUIDA($\langle expr \rangle \to$ - $\langle expr \rangle \langle expr \rangle$)	{ - }
GUIDA($\langle expr \rangle \to$ * ($\langle exprlist \rangle$))	{ * }
GUIDA($\langle expr \rangle \to$ / $\langle expr \rangle \langle expr \rangle$)	{ / }
GUIDA($\langle expr \rangle \to$ NUM)	{ NUM }
GUIDA($\langle expr \rangle \to$ ID)	{ ID }
GUIDA($\langle exprlist \rangle \to \langle expr \rangle \langle exprlistp \rangle$)	{ +, -, *, /, NUM, ID }
GUIDA($\langle exprlistp \rangle \to$, $\langle expr \rangle \to \langle exprlistp \rangle$)	{ , }
GUIDA($\langle exprlistp \rangle \to \varepsilon$)	{) }

1.3 Trasformazione in una grammatica LL(1) equivalente

La grammatica data non è LL(1) per via dei seguenti insiemi guida per la variabile $\langle \mathit{stat} \rangle$.

1.3.1 Produzione per for

Dati gli insiemi guida non LL(1) della grammatica per `for`:

GUIDA($\langle \mathit{stat} \rangle \rightarrow$ `for ( ID :=` $\langle \mathit{expr} \rangle$ `;` $\langle \mathit{bexpr} \rangle$ `) do` $\langle \mathit{stat} \rangle$)	{ `for` }
GUIDA($\langle \mathit{stat} \rangle \rightarrow$ `for (` $\langle \mathit{bexpr} \rangle$ `) do` $\langle \mathit{stat} \rangle$)	{ `for` }

Fattorizzando la parte <u>non</u> comune ad ambo le produzioni introducendo una variabile $\langle \mathit{statc} \rangle$,
otteniamo un nuovo insieme guida per $\langle \mathit{stat} \rangle$ che risulta essere LL(1):

GUIDA($\langle \mathit{stat} \rangle \rightarrow$ `for (` $\langle \mathit{statc} \rangle$ $\langle \mathit{bexpr} \rangle$ `) do` $\langle \mathit{stat} \rangle$)	{ `for` }
GUIDA($\langle \mathit{statc} \rangle \rightarrow$ `ID :=` $\langle \mathit{expr} \rangle$ `;`)	{ `ID` }
GUIDA($\langle \mathit{statc} \rangle \rightarrow \varepsilon$	{ `<`, `>`, `<=`, `>=`, `==`, `<>` }

1.3.2 Produzione per if

Dati gli insiemi guida non LL(1) della grammatica per `if`:

GUIDA($\langle \mathit{stat} \rangle \rightarrow$ `if (` $\langle \mathit{bexpr} \rangle$ `)` $\langle \mathit{stat} \rangle$ `else` $\langle \mathit{stat} \rangle$ `end`)	{ `if` }
GUIDA($\langle \mathit{stat} \rangle \rightarrow$ `if (` $\langle \mathit{bexpr} \rangle$ `)` $\langle \mathit{stat} \rangle$ `end`)	{ `if` }

Fattorizzando la parte <u>non</u> comune ad ambo le produzioni introducendo una variabile $\langle \mathit{statc} \rangle$,
otteniamo un nuovo insieme guida per $\langle \mathit{stat} \rangle$ che risulta essere LL(1):

GUIDA($\langle \mathit{stat} \rangle \rightarrow$ `if (` $\langle \mathit{bexpr} \rangle$ `)` $\langle \mathit{stat} \rangle \langle \mathit{statp} \rangle$ `end`)	{ `if` }
GUIDA($\langle \mathit{statp} \rangle \rightarrow$ `else` $\langle \mathit{stat} \rangle$)	{ `else` }
GUIDA($\langle \mathit{statp} \rangle \rightarrow \varepsilon$)	{ `end` }

1.4 Grammatica LL(1) completa

GUIDA(⟨*prog*⟩ → ⟨*statlist*⟩ `EOF`)	{ `assign`, `print`, `read`, `for`, `if`, `{` }
GUIDA(⟨*statlist*⟩ → ⟨*stat*⟩⟨*statlistp*⟩)	{ `assign`, `print`, `read`, `for`, `if`, `{` }
GUIDA(⟨*statlistp*⟩ → `;` ⟨*stat*⟩⟨*statlistp*⟩)	{ `;` }
GUIDA(⟨*statlistp*⟩ → ε)	{ `EOF`, `}` }
GUIDA(⟨*stat*⟩ → `assign` ⟨*assignlist*⟩)	{ `assign` }
GUIDA(⟨*stat*⟩ → `print` `(` ⟨*exprlist*⟩ `)`)	{ `print` }
GUIDA(⟨*stat*⟩ → `read` `(` ⟨*idlist*⟩ `)`)	{ `read` }
GUIDA(⟨*stat*⟩ → `for` `(` ⟨*statc*⟩ ⟨*bexpr*⟩ `)` `do` ⟨*stat*⟩)	{ `for` }
GUIDA(⟨*stat*⟩ → `if` `(` ⟨*bexpr*⟩ `)` ⟨*stat*⟩⟨*statp*⟩ `end`)	{ `if` }
GUIDA(⟨*stat*⟩ → `{` ⟨*statlist*⟩ `}`)	{ `{` }
GUIDA(⟨*statc*⟩ → `ID` `:=` ⟨*expr*⟩ `;`)	{ `ID` }
GUIDA(⟨*statc*⟩ → ε	{ `<`, `>`, `<=`, `>=`, `==`, `<>` }
GUIDA(⟨*statp*⟩ → `else` ⟨*stat*⟩)	{ `else` }
GUIDA(⟨*statp*⟩ → ε)	{ `end` }
GUIDA(⟨*assignlist*⟩ → `[` ⟨*expr*⟩ `to` ⟨*idlist*⟩ `]` ⟨*assignlistp*⟩)	{ `[` }
GUIDA(⟨*assignlistp*⟩ → `[` ⟨*expr*⟩ `to` ⟨*idlist*⟩ `]` ⟨*assignlistp*⟩)	{ `[` }
GUIDA(⟨*assignlistp*⟩ → ε)	{ `;`, `else`, `end`, `EOF`, `}` }
GUIDA(⟨*idlist*⟩ → `ID` ⟨*idlistp*⟩)	{ `ID` }
GUIDA(⟨*idlistp*⟩ → `,` `ID` ⟨*idlistp*⟩)	{ `,` }
GUIDA(⟨*idlistp*⟩ → ε)	{ `)`, `]` }
GUIDA(⟨*bexpr*⟩ → `<` ⟨*expr*⟩⟨*expr*⟩)	{ `<` }
GUIDA(⟨*bexpr*⟩ → `>` ⟨*expr*⟩⟨*expr*⟩)	{ `>` }
GUIDA(⟨*bexpr*⟩ → `<=` ⟨*expr*⟩⟨*expr*⟩)	{ `<=` }
GUIDA(⟨*bexpr*⟩ → `<=` ⟨*expr*⟩⟨*expr*⟩)	{ `<=` }
GUIDA(⟨*bexpr*⟩ → `==` ⟨*expr*⟩⟨*expr*⟩)	{ `==` }
GUIDA(⟨*bexpr*⟩ → `<>` ⟨*expr*⟩⟨*expr*⟩)	{ `<>` }
GUIDA(⟨*expr*⟩ → `+` `(` ⟨*exprlist*⟩ `)`)	{ `+` }
GUIDA(⟨*expr*⟩ → `-` ⟨*expr*⟩⟨*expr*⟩)	{ `-` }
GUIDA(⟨*expr*⟩ → `*` `(` ⟨*exprlist*⟩ `)`)	{ `*` }
GUIDA(⟨*expr*⟩ → `/` ⟨*expr*⟩⟨*expr*⟩)	{ `/` }
GUIDA(⟨*expr*⟩ → `NUM`)	{ `NUM` }
GUIDA(⟨*expr*⟩ → `ID`)	{ `ID` }
GUIDA(⟨*exprlist*⟩ → ⟨*expr*⟩⟨*exprlistp*⟩)	{ `+`, `-`, `*`, `/`, `NUM`, `ID` }
GUIDA(⟨*exprlistp*⟩ → `,` ⟨*expr*⟩⟨*exprlistp*⟩)	{ `,` }
GUIDA(⟨*exprlistp*⟩ → ε)	{ `)` }