

Relazione del progetto del corso di Complementi di linguaggi di programmazione

Andrea Loretto[0001097539]

Luca Gruppi[0001099394]

Indice

0	Introduzione	3
0.1	Struttura relazione	3
0.2	Obiettivo della progetto	3
0.3	Ambiente di lavoro	3
1	Esercizio 1 - Analisi Lessicale	3
1.1	Esempi di codice	3
2	Esercizio 2 - Tabella dei simboli	6
2.1	Esempi di codice	6
3	Esercizio 3 - Analisi semantica	7
3.1	Esempi di codice	10
4	Esercizio 4 - Interprete	13
4.1	Esempi di codice	13

0 Introduzione

0.1 Struttura relazione

Questa relazione ha l'obiettivo di spiegare il lavoro svolto all'interno del progetto e spiegarne i concetti principali.

La relazione si divide in quattro capitoli, ognuno dei quali corrisponde a uno degli esercizi assegnati. In ogni capitolo, descriviamo il problema da risolvere, la soluzione proposta, le difficoltà incontrate e i test effettuati.

0.2 Obiettivo della progetto

L'obiettivo del progetto è sviluppare SimpLanPlus un compilatore che estende SimpLan in quattro esercizi:

1. **Analisi lessicale:** Controllo sulla presenza di errori sintattici nell'input.
2. **Symbol Table:** Creazione della tabella di simboli per la gestione degli scope.
3. **Analisi semantica:** Verifica della correttezza dei tipi, l'uso di variabili non inizializzate e utilizzo corretto della semantica.
4. **Interprete:** Estendere l'interprete SimpLan per creare l'interprete SimpLanPlus.

0.3 Ambiente di lavoro

L'IDE utilizzato per lo sviluppo è IntelliJ IDEA community dotato di plugin per il supporto ad ANTLR. Nella main-directory del progetto ci sono quattro cartelle:

- **gen:** Nella cartella gen sono presenti tutti i file generati da ANTLR per entrambe le grammatiche, SimpLanPlus.g4 e SVM.g4.
- **lib:** La cartella lib contiene il file antlr-4.12.0-complete.jar di ANTLR.
- **out:** Nella cartella out ci sono tutti i file di output della compilazione.
- **src:** Nella cartella src ci sono tutti i file sorgente del progetto e il file di input per la compilazione.

1 Esercizio 1 - Analisi Lessicale

1.1 Esempi di codice

Di seguito abbiamo eseguito alcune linee di codice con errori lessicali e i loro relativi output.

Come si può notare il codice se rileva errori lessicali non prosegue con l'esecuzione della type check e c-gen ma si interrompe.

Gli errori lessicali vengono tutti inseriti in un file di output chiamato Error.txt.

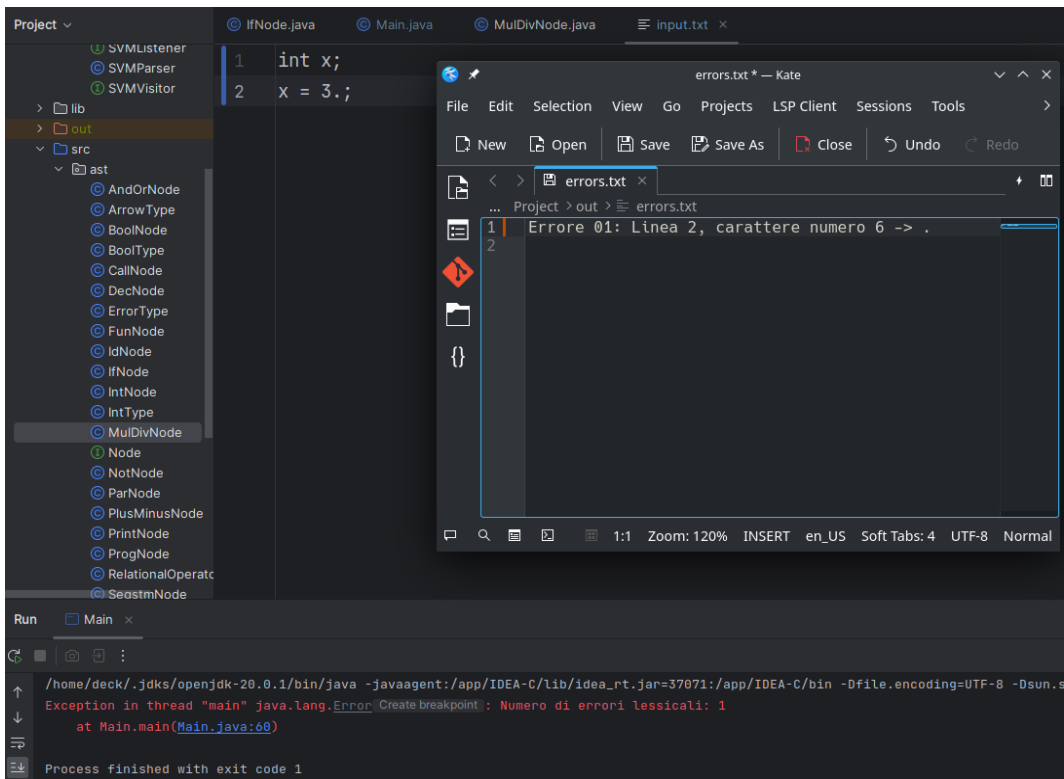


Figura 1: Analisi lessicale esempio 1

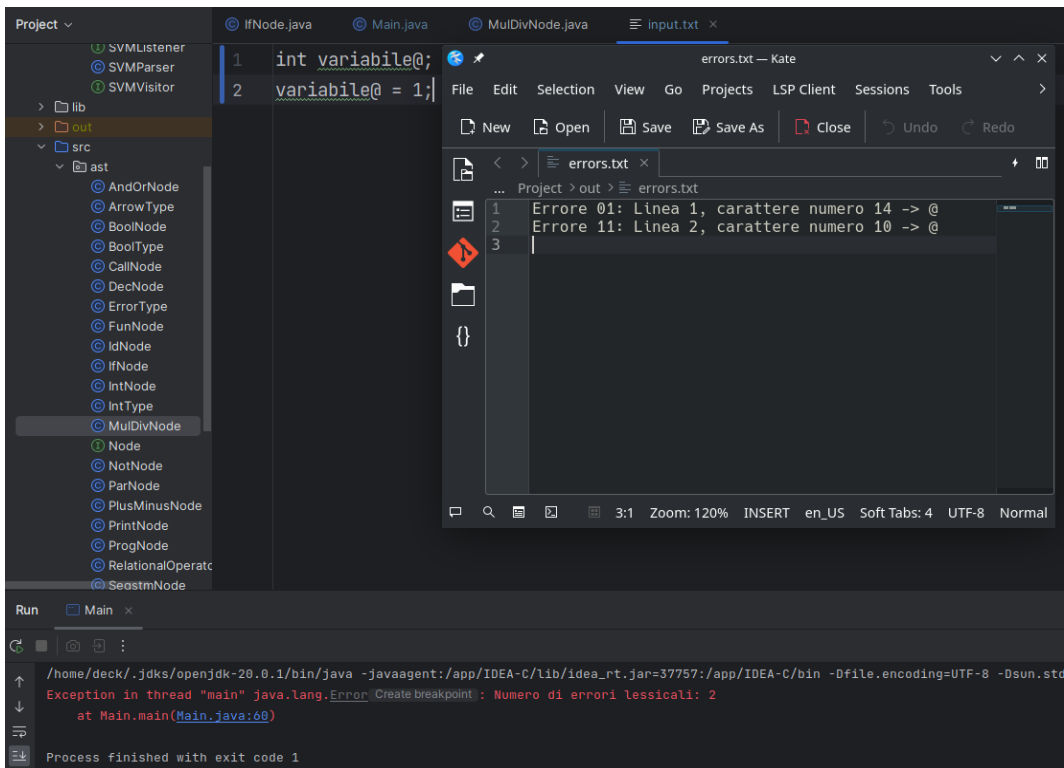


Figura 2: Analisi lessicale esempio 2

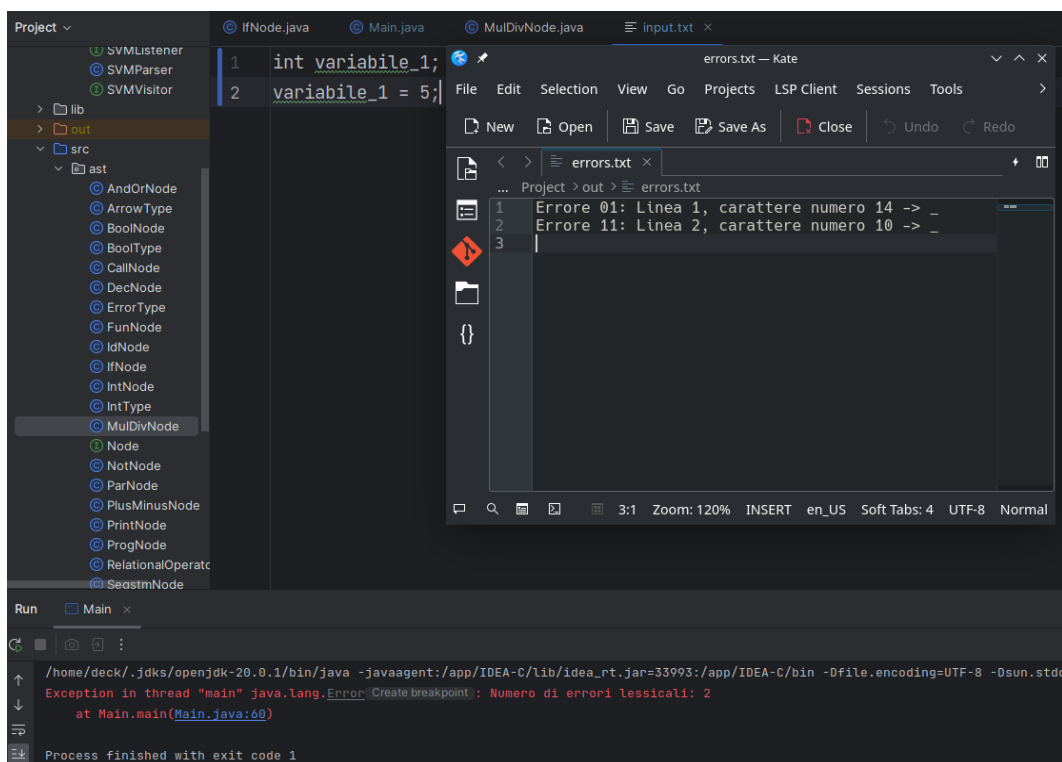


Figura 3: Analisi lessicale esempio 3

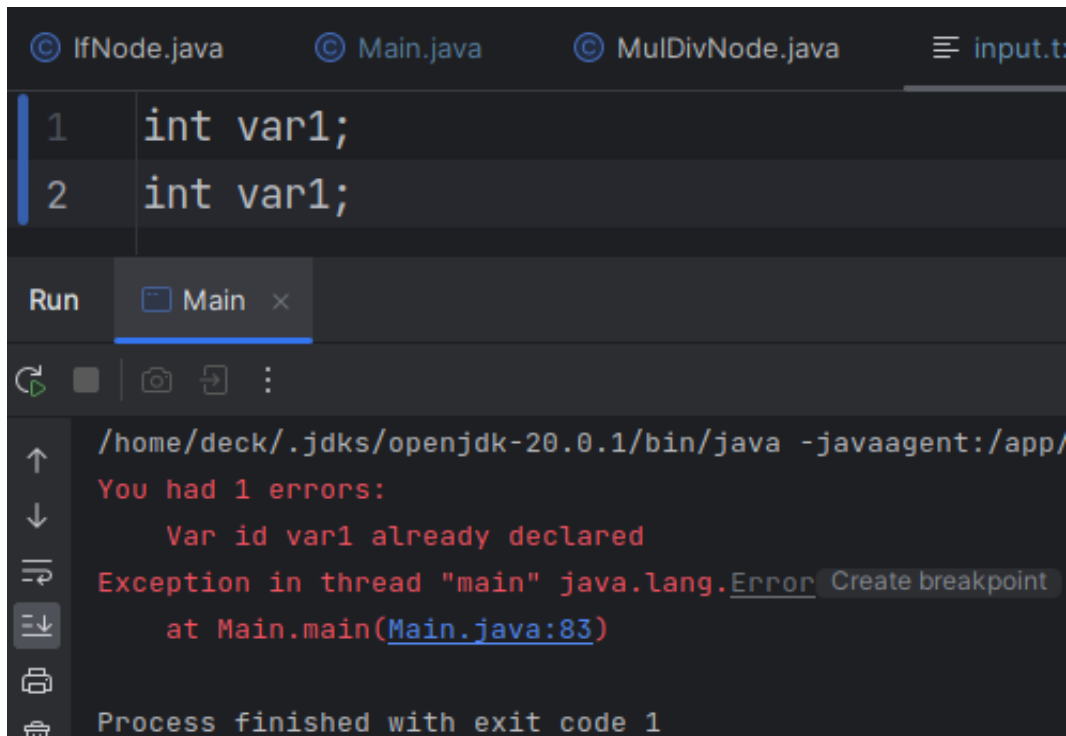
Da come si può notare dagli esempi gli errori lessicali si hanno quando ci sono caratteri non riconosciuti dalla grammatica. Questa è la prima fase, serve per eliminare tutti quegli errori di "battitura".

Nel primo esempio dopo il 3 è presente un ' '.

Nel secondo e nel terzo esempio si usano i caratteri @ e _ all'interno di nomi di variabili.

2 Esercizio 2 - Tabella dei simboli

2.1 Esempi di codice



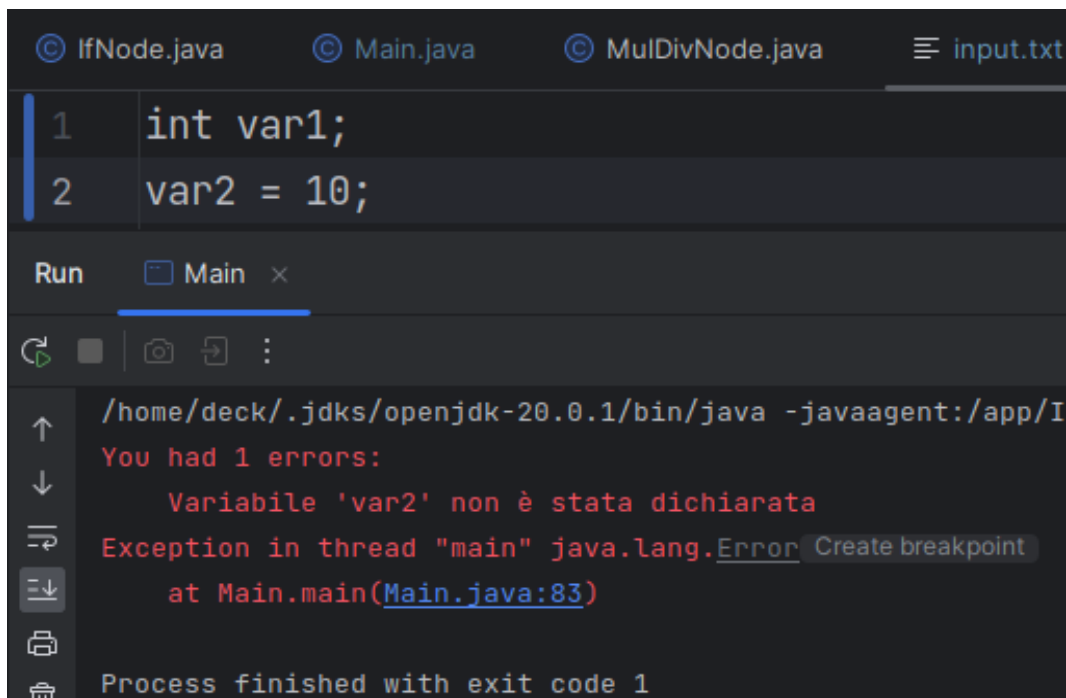
The screenshot shows an IDE with three tabs: IfNode.java, Main.java, and MulDivNode.java. The Main.java tab is active, showing the following code:

```
1 int var1;  
2 int var1;
```

Below the code editor, the Run button is visible, and the Main window is open. The console output shows the following error:

```
/home/deck/.jdk/openjdk-20.0.1/bin/java -javaagent:/app/...  
You had 1 errors:  
    Var id var1 already declared  
Exception in thread "main" java.lang.Error Create breakpoint  
    at Main.main(Main.java:83)  
Process finished with exit code 1
```

Figura 4:



The screenshot shows the same IDE with the Main.java tab active. The code is now:

```
1 int var1;  
2 var2 = 10;
```

The console output shows the following error:

```
/home/deck/.jdk/openjdk-20.0.1/bin/java -javaagent:/app/I...  
You had 1 errors:  
    Variabile 'var2' non è stata dichiarata  
Exception in thread "main" java.lang.Error Create breakpoint  
    at Main.main(Main.java:83)  
Process finished with exit code 1
```

Figura 5:



Figura 6:

Da come si può notare dagli esempi gli errori si hanno quando c'è un incongruenza tra le variabili usate nel codice e la symbol table.

Non si possono usare variabili non dichiarate e non si possono assegnare variabili non inizializzate.

Quindi per esempio anche il caso

```
int var1;
var1 = var1;
```

da errore, dato che la variabile var1 nel momento dell'assegnamento non è ancora stata inizializzata.

```
int var1;
int var2;
bool controllo;
controllo = true;
if (controllo == true)
    {var1 = 5;}
else
    {var2 = 5;}
```

Questo è un altro caso di errore.

Non si può sapere che ramo dell'if si ha percorso per cui nel caso in cui le symbol table siano diverse per evitare errori successivi si dà errore.

Nel caso i due rami dell'if modifichino la symbol table nello stesso modo non è necessario dare errore.

Nel primo esempio la variabile var1 viene dichiarata 2 volte.

Nel secondo e nel terzo esempio la variabile var2 non è stata dichiarata.

3 Esercizio 3 - Analisi semantica

Qui sotto presentiamo tutte le regole di inferenza relative alla grammatica:

- Prog (multipleExp, singleExp)

$$\frac{\frac{\frac{\emptyset \cdot [], 0 \vdash dec : \Gamma, n \quad \Gamma, n \vdash stm : \Gamma', n \quad \Gamma', n \vdash exp : T, init}{\emptyset, 0 \vdash dec \quad stm \quad exp : T}}{\emptyset \cdot [], 0 \vdash exp : T, init}}{\emptyset, 0 \vdash exp : T}$$

- varDec

$$\frac{ID \notin \text{dom}(\text{top}(\Gamma))}{\Gamma, n \vdash T \quad ID : \Gamma[ID \mapsto T, dec], n + 1}$$

- DecSeq

$$\frac{}{\Gamma, n \vdash d : \Gamma', n' \quad \Gamma', n' \vdash D : \Gamma'', n''} \quad \Gamma, n \vdash d D : \Gamma'', n''$$

- funDec

$$\frac{\Gamma \cdot [f \mapsto (T_1 \dots T_n) \rightarrow T, x_1 \mapsto T_1, \dots, x_n \mapsto T_n], n \vdash \text{body} : T' \quad T = T', \quad f \notin \text{dom}(\text{top}(\Gamma))}{\Gamma, n \vdash T f (T_1 x_1 \dots T_n x_n) = \text{body}; \quad \Gamma[f \mapsto (T_1 \dots T_n) \rightarrow T], n}$$

- funBody

$$\frac{\Gamma, n \vdash \text{dec} : \Gamma', n' \quad \Gamma', n' \vdash \text{stm} : \Gamma'', n' \quad \Gamma'', n' \vdash \text{exp} : T, \text{init}}{\Gamma, n \vdash \text{dec stm exp} : T}$$

$$\frac{\Gamma, n \vdash \text{dec} : \Gamma', n' \quad \Gamma', n' \vdash \text{stm} : \Gamma'', n' \quad \dots}{\Gamma, n \vdash \text{dec stm} : \text{void}}$$

- StmAsg

$$\frac{\Gamma, n \vdash e : T, \text{init} \quad \Gamma, n \vdash \text{ID} : T', S \quad T = T'}{\Gamma, n \vdash \text{ID} = e : \Gamma[\text{ID} \rightarrow T, \text{init}], n}$$

Dove lo stato $S \in \{\text{dec}, \text{init}\}$

- stmCallFun

$$\frac{\Gamma, n \vdash f : T_1 \times \dots \times T_n \rightarrow T \quad (\Gamma, n \vdash e_i : T'_i)_{i \in 1 \dots n} \quad (T_i = T'_i)_{i \in 1 \dots n}}{\Gamma, n \vdash f(e_1 \dots e_n) : \Gamma, n}$$

- StmSeq

$$\frac{\Gamma, n \vdash \text{stm1} : \Gamma', n \quad \Gamma', n \vdash \text{stm2} : \Gamma'', n}{\Gamma, n \vdash \text{stm1 stm2} : \Gamma'', n}$$

- stmIf

$$\frac{\Gamma, n \vdash e1 : \text{bool}, \text{init} \quad \Gamma, n \vdash \text{Stm1} : \Gamma', n \quad \Gamma, n \vdash \text{Stm2} : \Gamma'', n \quad \Gamma''' = \text{choose}(\Gamma', \Gamma'')}{\Gamma, n \vdash \text{if } e1 \{ \text{Stm1} \} \text{ else } \{ \text{Stm2} \} : \Gamma'', n}$$

- Int

$$\frac{}{\Gamma, n \vdash \text{NUM} : \text{int}, \text{init}}$$

- Bool

$$\frac{}{\Gamma, n \vdash \text{true} : \text{bool}, \text{init}}$$

$$\frac{}{\Gamma, n \vdash \text{false} : \text{bool}, \text{init}}$$

- expId

$$\frac{\Gamma(\text{ID}) : T, S}{\Gamma, n \vdash \text{ID} : T, S}$$

Dove lo stato $S \in \{\text{dec}, \text{init}\}$

- expNotId

$$\frac{\Gamma, n \vdash e : \text{bool}, \text{init} \quad ! : \text{bool} \rightarrow \text{bool}}{\Gamma, n \vdash !e : \text{bool}, \text{init}}$$

- expMulDiv

$$\begin{array}{c}
- \frac{\Gamma, n \vdash e1 : T, \text{init} \quad \Gamma, n \vdash e2 : T', \text{init} \quad T = \text{int} = T' \quad * : \text{int} \times \text{int} \rightarrow \text{int}}{\Gamma, n \vdash e1 * e2 : \text{int}, \text{init}} \\
- \frac{\Gamma, n \vdash e1 : T, \text{init} \quad \Gamma, n \vdash e2 : T', \text{init} \quad T = \text{int} = T' \quad / : \text{int} \times \text{int} \rightarrow \text{int}}{\Gamma, n \vdash e1 / e2 : \text{int}, \text{init}}
\end{array}$$

- expPlusMinus

$$\begin{array}{c}
- \frac{\Gamma, n \vdash e1 : T, \text{init} \quad \Gamma, n \vdash e2 : T', \text{init} \quad T = \text{int} = T' + : \text{int} \times \text{int} \rightarrow \text{int}}{\Gamma, n \vdash e1 + e2 : \text{int}, \text{init}} \\
- \frac{\Gamma, n \vdash e1 : T, \text{init} \quad \Gamma, n \vdash e2 : T', \text{init} \quad T = \text{int} = T' - : \text{int} \times \text{int} \rightarrow \text{int}}{\Gamma, n \vdash e1 - e2 : \text{int}, \text{init}}
\end{array}$$

- expReop

$$\begin{array}{c}
- \frac{\Gamma, n \vdash e2 : T, \text{init} \quad \Gamma, n \vdash e1 : T', \text{init} \quad T = T' == : T \times T \rightarrow \text{bool}}{\Gamma, n \vdash e1 == e2 : \text{bool}, \text{init}} \\
- \frac{\Gamma, n \vdash e2 : T, \text{init} \quad \Gamma, n \vdash e1 : T', \text{init} \quad T = \text{int} = T' \geq : \text{int} \times \text{int} \rightarrow \text{bool}}{\Gamma, n \vdash e1 \geq e2 : \text{bool}, \text{init}} \\
- \frac{\Gamma, n \vdash e2 : T, \text{init} \quad \Gamma, n \vdash e1 : T', \text{init} \quad T = \text{int} = T' \leq : \text{int} \times \text{int} \rightarrow \text{bool}}{\Gamma, n \vdash e1 \leq e2 : \text{bool}, \text{init}} \\
- \frac{\Gamma, n \vdash e2 : T, \text{init} \quad \Gamma, n \vdash e1 : T', \text{init} \quad T = \text{int} = T' > : \text{int} \times \text{int} \rightarrow \text{bool}}{\Gamma, n \vdash e1 > e2 : \text{bool}, \text{init}} \\
- \frac{\Gamma, n \vdash e2 : T, \text{init} \quad \Gamma, n \vdash e1 : T', \text{init} \quad T = \text{int} = T' < : \text{int} \times \text{int} \rightarrow \text{bool}}{\Gamma, n \vdash e1 < e2 : \text{bool}, \text{init}}
\end{array}$$

- expAndOr

$$\begin{array}{c}
- \frac{\Gamma, n \vdash e1 : \text{bool}, \text{init} \quad \Gamma, n \vdash e2 : \text{bool}, \text{init} \quad \&\& : \text{bool} \times \text{bool} \rightarrow \text{bool}}{\Gamma, n \vdash e1 \&\& e2 : \text{bool}, \text{init}} \\
- \frac{\Gamma, n \vdash e1 : \text{bool}, \text{init} \quad \Gamma, n \vdash e2 : \text{bool}, \text{init} \quad || : \text{bool} \times \text{bool} \rightarrow \text{bool}}{\Gamma, n \vdash e1 || e2 : \text{bool}, \text{init}}
\end{array}$$

- expIf

$$\frac{\Gamma, n \vdash e1 : \text{bool}, \text{init} \quad \Gamma, n \vdash \text{Stm1} : \Gamma', n \quad \Gamma', n \vdash \text{Exp1} : \Gamma'', T' \quad \Gamma, n \vdash \text{Stm2} : \Gamma''', n \quad \Gamma''', n \vdash \text{Exp2} : \Gamma'''' , T'' \quad T' = T'' \quad \Gamma'''' = \text{choose}(\Gamma'', \Gamma''')}{\Gamma, n \vdash \text{If } e1 \{ \text{Stm1 Exp1} \} \text{ else } \{ \text{Stm2 Exp2} \} : T, \text{init}}$$

- expBracket

$$- \frac{\Gamma, n \vdash e : T, \text{init}}{\Gamma, n \vdash (e) : T, \text{init}}$$

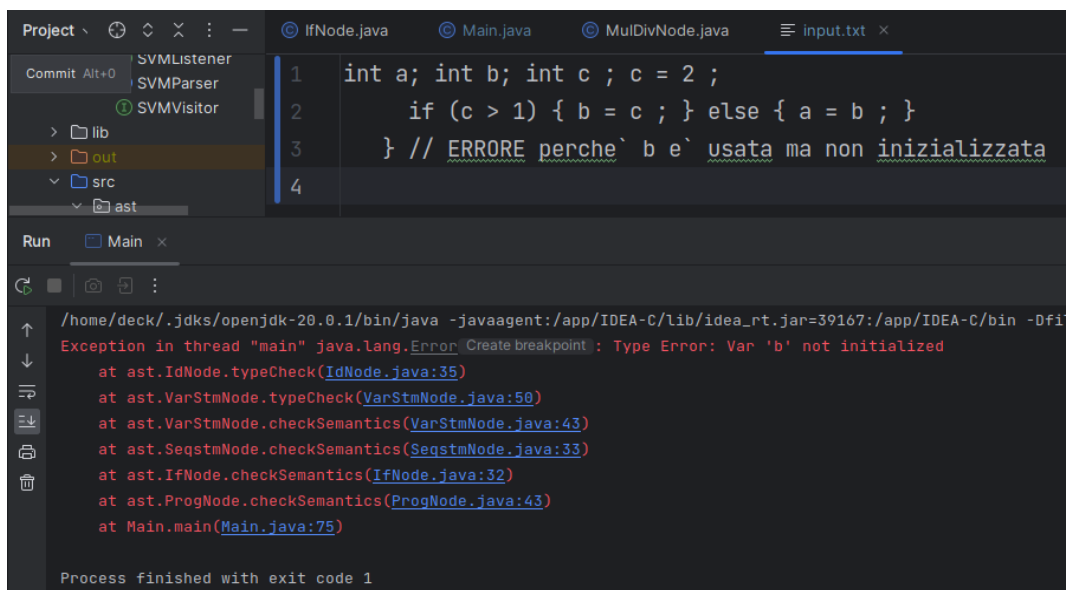
- expCallFun

$$- \frac{\Gamma, n \vdash f : T_1 \times \dots \times T_n \rightarrow T \quad (\Gamma, n \vdash e_i : T'_i)_{i \in 1 \dots n} \quad (T_i = T'_i)_{i \in 1 \dots n}}{\Gamma, n \vdash f(e_1 \dots e_n) : T, \text{init}}$$

- expThenBranch/expElseBranch

$$- \frac{\Gamma, n \vdash \text{stm} : \Gamma', n \quad \Gamma', n \vdash e : T, \text{init}}{\Gamma, n \vdash \text{stm } e : \Gamma', T}$$

3.1 Esempi di codice



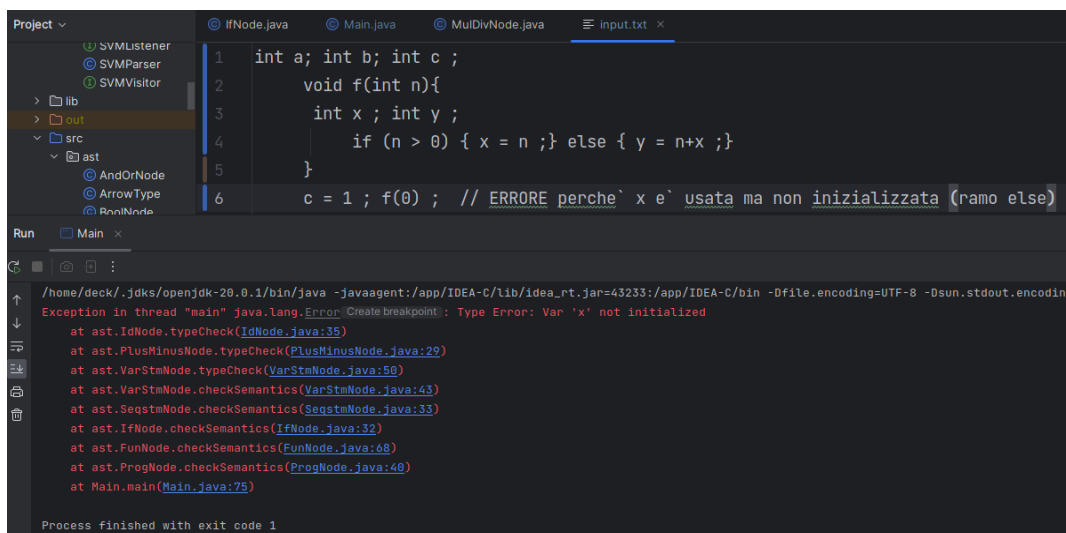
The screenshot shows an IDE with a project named 'SVM'. The 'src' directory contains 'ast' and 'out'. The 'ast' directory contains 'IdNode.java', 'VarStmNode.java', 'SeqstmNode.java', 'IfNode.java', and 'ProgNode.java'. The 'out' directory contains 'Main.java'. The 'Main.java' file is open, showing the following code:

```
1 int a; int b; int c ; c = 2 ;
2     if (c > 1) { b = c ; } else { a = b ; }
3     } // ERRORE perche' b e' usata ma non inizializzata
4
```

The 'Run' button is clicked, and the output window shows the following error:

```
Exception in thread "main" java.lang.Error: Create breakpoint : Type Error: Var 'b' not initialized
    at ast.IdNode.typeCheck(IdNode.java:35)
    at ast.VarStmNode.typeCheck(VarStmNode.java:50)
    at ast.VarStmNode.checkSemantics(VarStmNode.java:43)
    at ast.SeqstmNode.checkSemantics(SeqstmNode.java:33)
    at ast.IfNode.checkSemantics(IfNode.java:32)
    at ast.ProgNode.checkSemantics(ProgNode.java:43)
    at Main.main(Main.java:75)
Process finished with exit code 1
```

Figura 7:



The screenshot shows the same IDE with the 'Main.java' file open, showing the following code:

```
1 int a; int b; int c ;
2 void f(int n){
3     int x ; int y ;
4     if (n > 0) { x = n ; } else { y = n+x ; }
5 }
6 c = 1 ; f(0) ; // ERRORE perche' x e' usata ma non inizializzata (ramo else)
```

The 'Run' button is clicked, and the output window shows the following error:

```
Exception in thread "main" java.lang.Error: Create breakpoint : Type Error: Var 'x' not initialized
    at ast.IdNode.typeCheck(IdNode.java:35)
    at ast.PlusMinusNode.typeCheck(PlusMinusNode.java:20)
    at ast.VarStmNode.typeCheck(VarStmNode.java:50)
    at ast.VarStmNode.checkSemantics(VarStmNode.java:43)
    at ast.SeqstmNode.checkSemantics(SeqstmNode.java:33)
    at ast.IfNode.checkSemantics(IfNode.java:32)
    at ast.FunNode.checkSemantics(FunNode.java:68)
    at ast.ProgNode.checkSemantics(ProgNode.java:40)
    at Main.main(Main.java:75)
Process finished with exit code 1
```

Figura 8:

The screenshot shows an IDE with a project named 'Project'. The file explorer on the left shows a directory structure with 'lib', 'out', and 'src' folders. The 'src' folder contains an 'ast' directory with files like 'AndOrNode', 'ArrowType', and 'BinNode'. The main editor displays a Java file named 'Main.java' with the following code:

```

1 void h(int n){
2     int x ; int y ;
3     if (n==0){ x = n+1 ;} else { h(n-1) ; x = n ; y = x ;}
4 }
5 h(5) ; // CORRETTA

```

The 'Run' button is highlighted, and the output console shows the execution results. The output consists of a series of lines representing the execution of the program, including the final result and the exit code.

```

290: 7 999 998 16 999 998 5 15 0 0 -----SP = 990, FP = 990, AL = 995, RA = 52, A0 = 4, T1 = 989
297: 11 999 998 16 999 998 5 15 0 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 4, T1 = 995
298: 5 999 998 16 999 998 5 15 0 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 4, T1 = 994
299: 7 999 998 16 999 998 5 15 0 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 994
300: 11 999 998 16 999 998 5 15 0 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 995
301: 4 999 998 16 999 998 5 15 0 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 992
302: 7 999 998 16 999 998 5 15 5 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 992
303: 11 999 998 16 999 998 5 15 5 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 995
304: 5 999 998 16 999 998 5 15 5 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 992
305: 7 999 998 16 999 998 5 15 5 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 992
306: 11 999 998 16 999 998 5 15 5 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 995
307: 4 999 998 16 999 998 5 15 5 0 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 991
308: 20 999 998 16 999 998 5 15 5 5 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 991
309: 9 999 998 16 999 998 5 15 5 5 -----SP = 990, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 991
310: 19 999 998 16 999 998 5 15 -----SP = 992, FP = 996, AL = 995, RA = 52, A0 = 5, T1 = 991
311: 9 999 998 16 999 998 5 -----SP = 993, FP = 996, AL = 995, RA = 15, A0 = 5, T1 = 991
312: 18 999 998 16 999 998 -----SP = 994, FP = 996, AL = 995, RA = 15, A0 = 5, T1 = 991
313: 5 999 998 16 999 -----SP = 995, FP = 996, AL = 995, RA = 15, A0 = 5, T1 = 991
314: 7 999 998 16 999 -----SP = 995, FP = 999, AL = 995, RA = 15, A0 = 5, T1 = 991
315: 11 999 998 16 999 -----SP = 995, FP = 999, AL = 999, RA = 15, A0 = 5, T1 = 991
316: 18 999 998 16 999 -----SP = 995, FP = 999, AL = 998, RA = 15, A0 = 5, T1 = 991
317: 24 999 998 16 -----SP = 996, FP = 999, AL = 998, RA = 15, A0 = 5, T1 = 991
318: 25 999 998 16 -----SP = 996, FP = 999, AL = 998, RA = 15, A0 = 5, T1 = 991

Result: 5

Process finished with exit code 0

```

Figura 9:

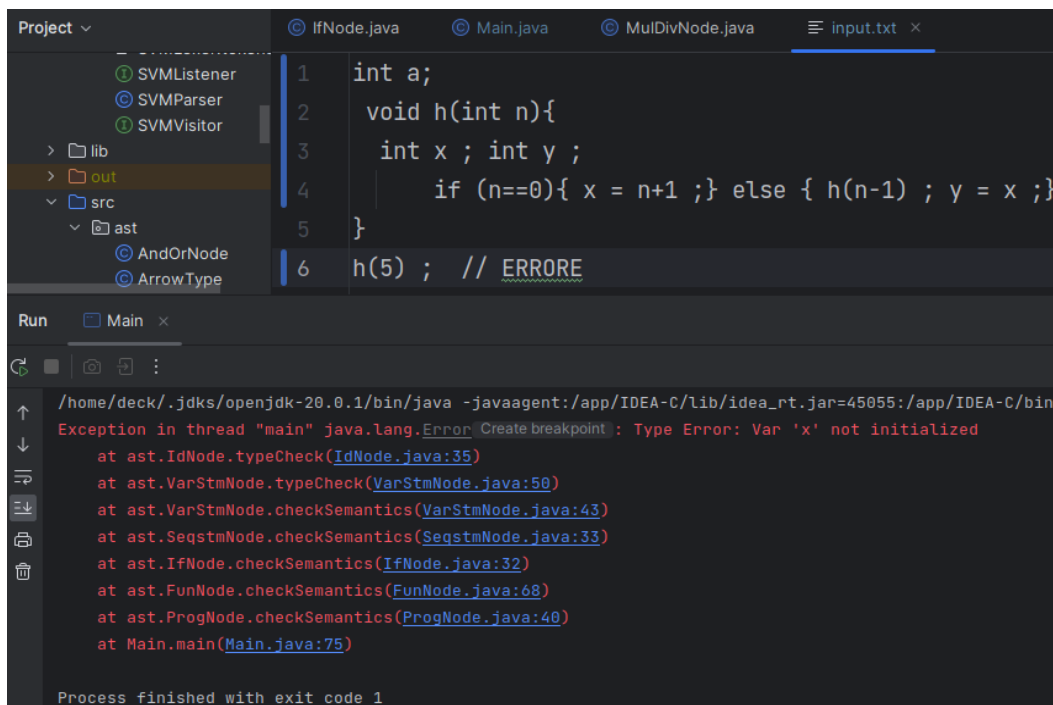


Figura 10:

Per quanto riguarda il primo esempio l'errore è dato dal fatto che 'b' non è inizializzato mentre lo si prova ad assegnare nel ramo dell'else.

Il secondo esempio è analogo al primo, la variabile 'x' non può essere usata nel ramo else se è stata inizializzata nel ramo then.

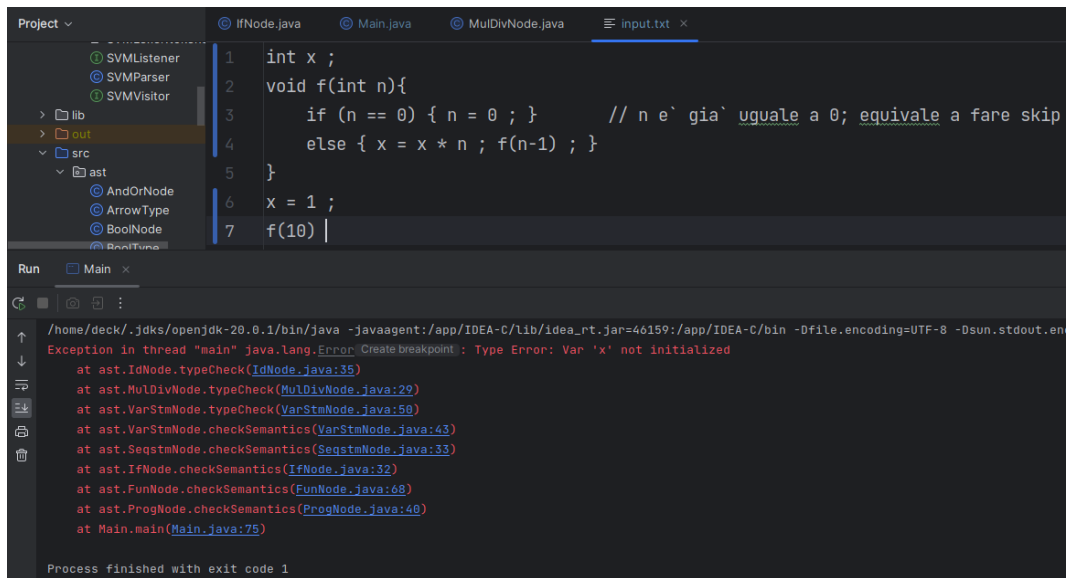
Questo perchè vengono create due symbol table identiche una volta che si ha un if.

I rami else e then modificano le due symbol table nuove e poi se sono uguali aggiornano la symbol table da cui sono state create, nel caso siano diverse da un errore.

Non in tutti i casi questo porterebbe ad un errore effettivo ma per prevenire alcun errore a runtime, si preferisce bloccare l'esecuzione del programma.

4 Esercizio 4 - Interprete

4.1 Esempi di codice



The screenshot shows an IDE with a project named 'SVM'. The left sidebar shows a file tree with 'lib', 'out', and 'src' folders. The 'src' folder contains an 'ast' package with several classes: 'AndOrNode', 'ArrowType', 'BoolNode', and 'RealType'. The main editor shows a file named 'input.txt' with the following Java code:

```
1 int x ;
2 void f(int n){
3     if (n == 0) { n = 0 ; }      // n e' gia' uguale a 0; equivale a fare skip
4     else { x = x * n ; f(n-1) ; }
5 }
6 x = 1 ;
7 f(10) ;
```

Below the code editor, the 'Run' tab is active, showing the output of the program. The output indicates a runtime error:

```
Exception in thread "main" java.lang.Error: Create breakpoint : Type Error: Var 'x' not initialized
    at ast.IdNode.typeCheck(IdNode.java:35)
    at ast.MulDivNode.typeCheck(MulDivNode.java:29)
    at ast.VarStmNode.typeCheck(VarStmNode.java:50)
    at ast.VarStmNode.checkSemantics(VarStmNode.java:43)
    at ast.SeqstmNode.checkSemantics(SeqstmNode.java:33)
    at ast.IfNode.checkSemantics(IfNode.java:32)
    at ast.FunNode.checkSemantics(FunNode.java:68)
    at ast.ProgNode.checkSemantics(ProgNode.java:40)
    at Main.main(Main.java:75)

Process finished with exit code 1
```

Figura 11:

The screenshot shows an IDE with a project structure on the left and a main editor window. The project structure includes a package 'ast' with classes like 'AndOrNode', 'ArrowType', 'BoolNode', 'BoolType', 'CallNode', and 'DecNode'. The main editor window displays a Java file 'Main.java' with the following code:

```

1  int u ;
2  int f(int n){
3      int y ;
4      y = 1 ;
5      if (n == 0) { y }
6      else { y = f(n-1) ; y*n }
7  }
8  u = 6 ;
9  f(u)

```

The 'Run' window at the bottom shows the execution output, which is a detailed stack trace of the recursive function calls. The output shows the state of the program at various points, including the values of 'u', 'n', and 'y'. The final result is 720.

```

381: 17 999 998 6 23 999 998 6 22 120 -----SP = 990, FP = 995, AL = 994, RA = 62, A0 = 120, T1 = 991
382: 7 999 998 6 23 999 998 6 22 120 120 -----SP = 989, FP = 995, AL = 994, RA = 62, A0 = 120, T1 = 991
383: 11 999 998 6 23 999 998 6 22 120 120 -----SP = 989, FP = 995, AL = 994, RA = 62, A0 = 120, T1 = 994
384: 5 999 998 6 23 999 998 6 22 120 120 -----SP = 989, FP = 995, AL = 994, RA = 62, A0 = 120, T1 = 993
385: 19 999 998 6 23 999 998 6 22 120 120 -----SP = 989, FP = 995, AL = 994, RA = 62, A0 = 6, T1 = 993
386: 12 999 998 6 23 999 998 6 22 120 -----SP = 990, FP = 995, AL = 994, RA = 62, A0 = 6, T1 = 120
387: 19 999 998 6 23 999 998 6 22 120 720 -----SP = 989, FP = 995, AL = 994, RA = 62, A0 = 6, T1 = 120
388: 20 999 998 6 23 999 998 6 22 120 -----SP = 990, FP = 995, AL = 994, RA = 62, A0 = 720, T1 = 120
389: 9 999 998 6 23 999 998 6 22 120 -----SP = 990, FP = 995, AL = 994, RA = 62, A0 = 720, T1 = 120
390: 19 999 998 6 23 999 998 6 22 -----SP = 991, FP = 995, AL = 994, RA = 62, A0 = 720, T1 = 120
391: 9 999 998 6 23 999 998 6 -----SP = 992, FP = 995, AL = 994, RA = 22, A0 = 720, T1 = 120
392: 18 999 998 6 23 999 998 -----SP = 993, FP = 995, AL = 994, RA = 22, A0 = 720, T1 = 120
393: 5 999 998 6 23 999 -----SP = 994, FP = 995, AL = 994, RA = 22, A0 = 720, T1 = 120
394: 7 999 998 6 23 999 -----SP = 994, FP = 999, AL = 994, RA = 22, A0 = 720, T1 = 120
395: 11 999 998 6 23 999 -----SP = 994, FP = 999, AL = 999, RA = 22, A0 = 720, T1 = 120
396: 18 999 998 6 23 999 -----SP = 994, FP = 999, AL = 998, RA = 22, A0 = 720, T1 = 120
397: 24 999 998 6 23 -----SP = 995, FP = 999, AL = 998, RA = 22, A0 = 720, T1 = 120
398: 25 999 998 6 23 -----SP = 995, FP = 999, AL = 998, RA = 22, A0 = 720, T1 = 120

Result: 720

Process finished with exit code 0

```

Figura 12:

The screenshot shows an IDE with a project structure on the left and a main editor window. The project structure includes a package 'ast' with classes like 'AndOrNode', 'ArrowType', 'BoolNode', 'BoolType', 'CallNode', and 'DecNode'. The main editor window displays a Java file 'Main.java' with the following code:

```

1  int u ;
2  void f(int m, int n){
3      if (m>n) { u = m+n ;}
4      else { int x ; x = 1 ; f(m+1,n+1) ; }
5  }
6  f(5,4) ; u

```

The 'Run' window at the bottom shows the execution output, which includes an exception message. The exception is a 'NullPointerException' in the 'main' thread, caused by a null value being passed to the 'checkSemantics' method. The output also shows the stack trace of the exception.


```

/home/deck/.jids/openjdk-20.0.1/bin/java -javaagent:/app/IDEA-C/lib/idea_rt.jar=40071:/app/IDEA-C/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
Line 4:8 extraneous input 'int' expecting {'if', 10}
Line 4:14 no viable alternative at input 'x'
Exception in thread "main" java.lang.NullPointerException: Create breakpoint: Cannot invoke "ast.Node.checkSemantics(semanticanalysis.SymbolTable, int)" because "stm" is null
    at ast.SegstmNode.checkSemantics(SegstmNode.java:33)
    at ast.IfNode.checkSemantics(IfNode.java:32)
    at ast.FunNode.checkSemantics(FunNode.java:68)
    at ast.ProgNode.checkSemantics(ProgNode.java:40)
    at Main.main(Main.java:75)

Process finished with exit code 1

```

Figura 13:



```
1 int u ;
2 void f(int m, int n){
3     if (m>n) { u = m+n ;}
4     else { int x ; x = 1 ; f(m+1,n+1) ; }
5 }
6 f(4,5) ; u

Run Main x
/home/deck/.jdk/openjdk-20.0.1/bin/java -javaagent:/app/IDEA-C/lib/idea_rt.jar=34617:/app/IDEA-C/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
line 4:8 extraneous input 'int' expecting {'if', 'ID'}
line 4:14 no viable alternative at input 'x;'
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "ast.Node.checkSemantics(semanticanalysis.SymbolTable, int)" because "sta" is null
    at ast.SeqstNode.checkSemantics(SeqstNode.java:33)
    at ast.IfNode.checkSemantics(IfNode.java:32)
    at ast.FunNode.checkSemantics(FunNode.java:68)
    at ast.ProgNode.checkSemantics(ProgNode.java:40)
    at Main.main(Main.java:75)
Process finished with exit code 1
```

Figura 14:

Nelle figure 3 e 4 l'errore è dato da 'int x;' all'interno del ramo else dell'if.

Questo non è possibile perchè la grammatica non lo permette. Nel caso 'int x;' venga scritto prima dell' if, tra riga 2 e riga 3, il codica non da alcun errore.

Infatti la grammatica permette le dichiarazione all'interno del body delle funzioni.

Per quanto riguarda il primo esempio l'errore è dato dal fatto che 'x' non è inizializzato mentre lo si prova ad assegnare.