Specifica del linguaggio NewLang

Angelo Afeltra Mtr:0522501354
Corso di Compilatori A.A. 2022/23

January 19, 2023

Abstract

Il seguente documento contiene le specifiche del linguaggio "NewLang".
La sezione delle specifiche lessicali, contiente la lista dei token rilevati e dei pattern che vengono matchati.
Nella sezione delle specifiche sintattiche, viene riportata la grammatica del linguaggio con relativa tabella delle precedenze. Inoltre vengono riportati i costrutti del AST(Abstract Syntax Tree) generato.
Nella sezione delle spefiche sematiche vengono riportate le regole di type checking.
Mentre nell'ultima parte del documento è presente la specifica dei test effettuati.

1 Specifiche Lessicali

Table 1: Lexical Table.

Begin of Table	
Token	Pattern
MAIN	start:
VAR	var
INT	integer
FLOAT	float
STRING	string
BOOL	boolean
CHAR	char
VOID	void
DEF	def
OUT	out
FOR	for
IF	if
ELSE	else
THEN	then
WHILE	while
TO	to
LOOP	loop

Continuation of Table 5	
Token	Pattern
RETURN	return
TRUE	true
FALSE	false
AND	and
OR	or
NOT	not
ID	`[$_A-Za-z][$_A-Za-z0-9]*`
EQ	=
NE	<>, !=
LT	<
LE	<=
GT	>
GE	>=
PLUS	+
MINUS	-
TIMES	*
DIV	/
POW	^
STR_CONCAT	&
SEMI	;
COMMA	,
PIPE	|
LPAR	(
RPAR	)
LBRACK	{
RBRACK	}
READ	<--
WRITE	-->
WRITELN	-->!
COLON	:
ASSIGN	<<
INTEGER_CONST	`[0-9]+ (e-?[0-9]+)? | 0x[0-9a-f] | 0b[01]+`
REAL_CONST	`[0-9]+ . [0-9]+ (e-?[0-9]+)?`
STRING_CONST	`"[^\n\r\"\\]+"`
CHAR_CONST	`'[^\']'`
End of Table	

I blocchi di commento iniziano con `|*` e terminano con `|`, mentre i commenti in linea iniziano con `||`.

2 Specifiche Sintattiche

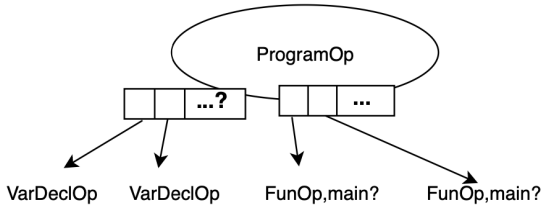
2.1 Grammatica

```
Program -> DeclList MainFunDecl DeclList
DeclList -> VarDecl DeclList | FunDecl DeclList | /* empty */
MainFunDecl -> MAIN FunDecl
VarDecl ::= Type IdInitList SEMI | VAR IdInitObblList SEMI
Type ::= INTEGER | BOOL | REAL | STRING | CHAR
IdInitList ::= ID
| IdInitList COMMA ID
| ID ASSIGN Expr
| IdInitList COMMA ID ASSIGN Expr
IdInitObblList ::= ID ASSIGN Const | IdInitObblList COMMA ID ASSIGN Const
Const ::= INTEGER_CONST | REAL_CONST | TRUE | FALSE | STRING_CONST | CHAR_CONST
FunDecl -> DEF ID LPAR ParamDeclList RPAR COLON TypeOrVoid Body
Body -> LBRACK VarDeclList StatList RBRACK
ParamDeclList ::= /*empty */ | NonEmptyParamDeclList
NonEmptyParamDeclList ::= ParDecl | NonEmptyParamDeclList PIPE ParDecl
ParDecl ::= Type IdList | OUT Type IdList
TypeOrVoid -> Type | VOID
VarDeclList -> /* empty */ | VardDecl VarDeclList
StatList ::= * empty */ | Stat StatList /*Modificato rispetto all'originale
Stat ::= IfStat | ForStat | ReadStat SEMI | WriteStat SEMI | AssignStat SEMI
| WhileStat | FunCall SEMI | RETURN Expr SEMI
| RETURN SEMI /*Modificato rispetto all'originale
IfStat ::= IF Expr THEN Body Else
Else ::= /* empty */ | ELSE Body
WhileStat ::= WHILE Expr LOOP Body
ForStat ::= FOR ID ASSIGN INTEGER_CONST TO INTEGER_CONST LOOP Body
ReadStat ::= IdList READ STRING_CONST | IdList READ
IdList ::= ID | IdList2 /*Modificato rispetto all'originale
IdList2 ::= COMMA IdList | /* empty */ /*Aggiunto rispetto all'originale
WriteStat ::= LPAR ExprList RPAR WRITE | LPAR ExprList RPAR WRITELN
AssignStat ::= IdList ASSIGN ExprList
FunCall ::= ID LPAR ExprList RPAR | ID LPAR RPAR
ExprList ::= Expr | Expr COMMA ExprList
Expr ::= TRUE | FALSE | INTEGER_CONST | REAL_CONST | STRING_CONST | CHAR_CONST
| ID | FunCall
| Expr PLUS Expr | Expr MINUS Expr | Expr TIMES Expr | Expr DIV Expr | Expr AND Expr
| Expr POW Expr | Expr STR_CONCAT Expr | Expr OR Expr | Expr GT Expr | Expr GE Expr
| Expr LT Expr | Expr LE Expr | Expr EQ Expr | Expr NE Expr | MINUS Expr | NOT Expr
| LPAR Expr RPAR
```

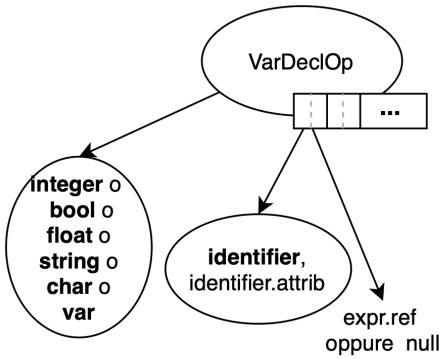
2.1.1 Tabella Precedenze

Terminale	Associatività
or	Sinistra
and	Sinistra
not	Destra
`=, <>, !=, <, <=, >, >=`	Nessuna
`+, -, *, /, ; &`	Sinistra

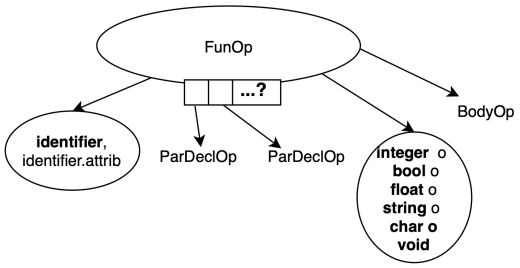
2.2 Costrutti AST (Abstract Syntax Tree)



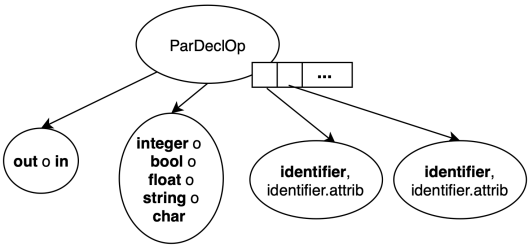
(a) Nodo ProgramOp



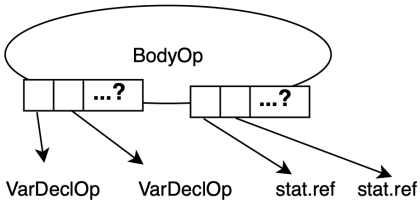
(b) Nodo VarDeclOp



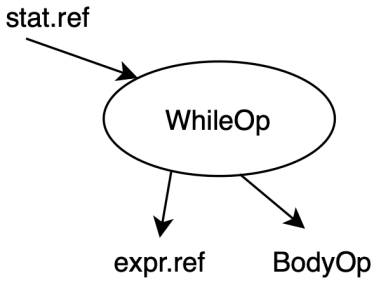
(c) Nodo FunOp



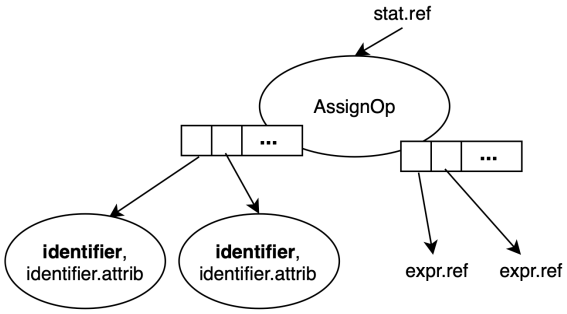
(d) Nodo ParDeclOp



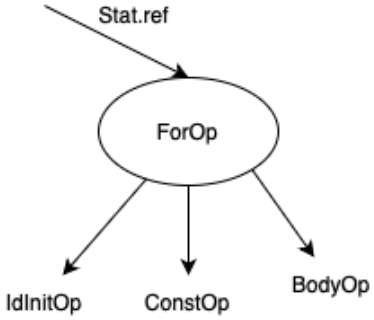
(e) Nodo BodyOp



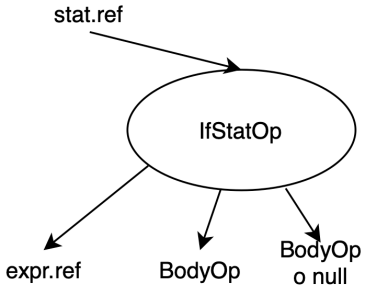
(f) Nodo WhileOp



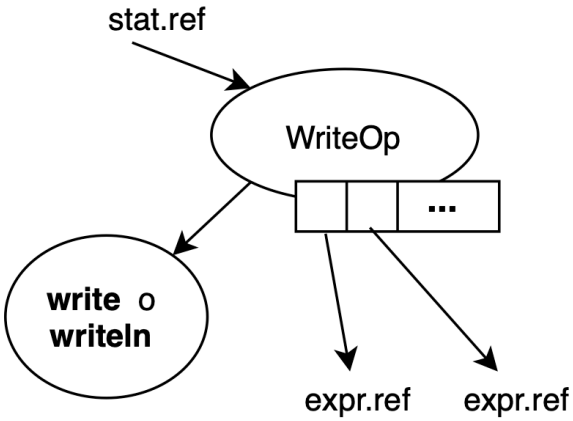
(g) Nodo AssignOp



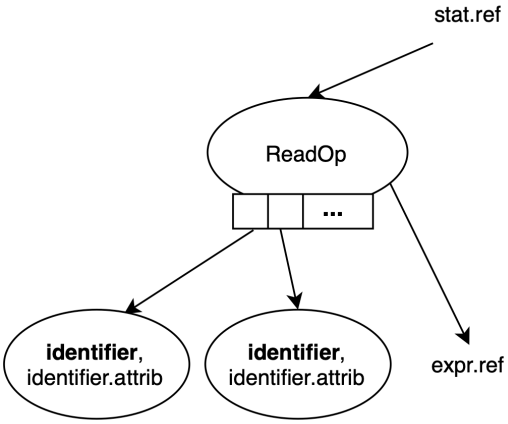
(h) Nodo ForOp



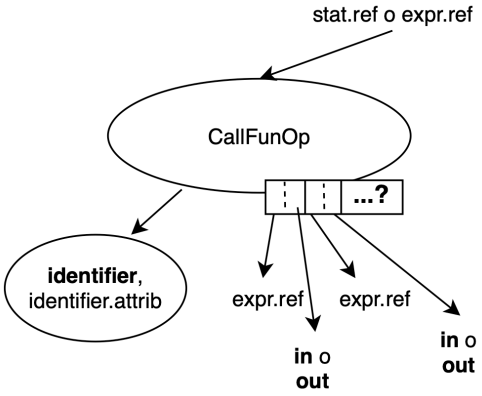
(i) Nodo IfStatOp



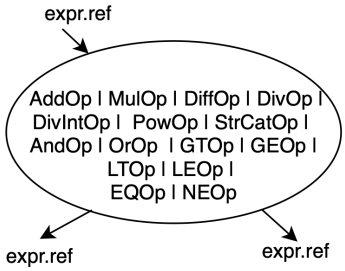
(j) Nodo WriteOp



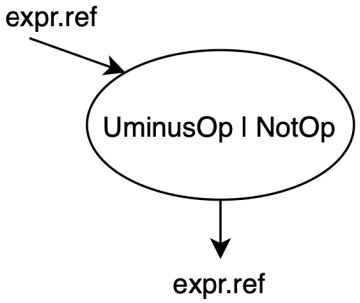
(k) Nodo ReadOp



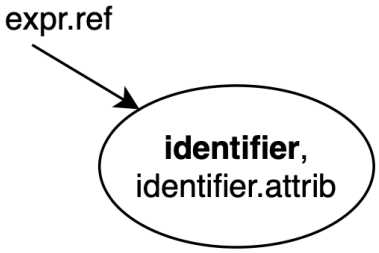
(l) Nodo CallFunOp



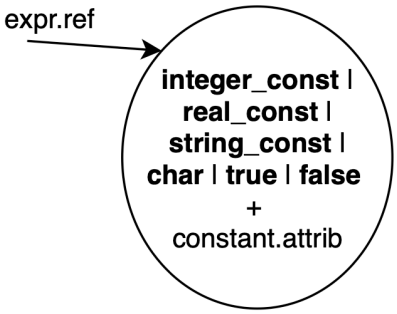
(m) Nodo BinaryOp



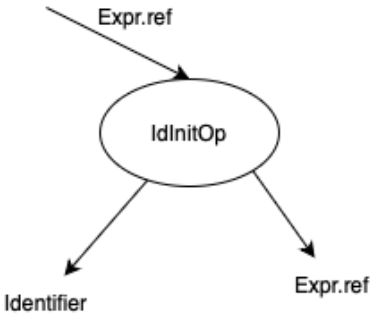
(n) Nodo UnaryOp



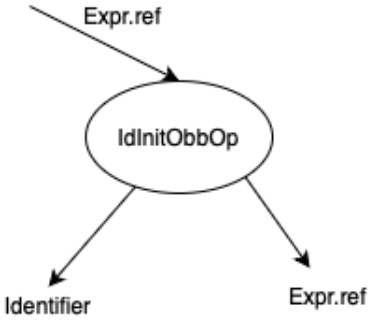
(o) Nodo Identifier



(p) Nodo ConstOp



(q) Nodo IdInitOp



(r) Nodo IdInitObbOp

3 Specifiche Semantiche

Identificatori

$$\frac{\Gamma(id) = \tau}{\Gamma \vdash id : \tau}$$

Constanti

$\Gamma \vdash integer\_const : integer$
$\Gamma \vdash real\_const : float$
$\Gamma \vdash char\_const : char$
$\Gamma \vdash string\_const : string$
$\Gamma \vdash true : boolean$
$\Gamma \vdash false : boolean$

Dichiarazione variabili

$$\frac{\Gamma \vdash e : \tau_1 \quad compatibilita(\tau, \tau_1) = true}{\Gamma \vdash \tau\ id << e}$$

tipo1	tipo2	risultato
integer	integer	true
float	float	true
float	integer	true
char	char	true
string	string	true
boolean	boolean	true
void	void	true

Table 2: Tabella per compatibilita

Dichiarazione Funzione

$$\frac{\Gamma \vdash p_i \neq p_j\ con\ i, j \in 1..n}{\Gamma \vdash def\ \ id\ \ (p_1, ..., p_n) : \tau\ \{bodyOp\}}$$

Read

$$\frac{(id_i^{i \in 1..N}) \in \Gamma}{\Gamma \vdash id_1, ..., id_n\texttt{<--} : notype}$$

Write

$\Gamma \vdash (e)\texttt{-->} : notype$

If-Then

$$\frac{\Gamma \vdash e : boolean \quad \Gamma \vdash body : notype}{\Gamma \vdash if\ e\ then\ body : notype}$$

For

$$\frac{\Gamma \vdash e_1 : int\_const \quad \Gamma \vdash e_2 : int\_const \quad \Gamma[id \mapsto integer] \vdash body : noType}{\Gamma \vdash for\ \ id << e_1\ \ to\ \ e_2\ \ loop\ \ body : noType}$$

While

$$\frac{\Gamma \vdash e : boolean \quad \Gamma \vdash body : notype}{\Gamma \vdash while\ e\ loop\ body : notype}$$

Assegnazione

$$\frac{\Gamma(id_i) : \tau_i^{i \in 1..n} \quad \Gamma \vdash e_i : \tau_i^{i \in 1..n}}{\Gamma \vdash id_1, ..., id_n << e_1, ..., e_n : noType}$$

Chiamata a funzione (espressione o istruzione)

$$\frac{\Gamma \vdash f : \tau_1 X ... X \tau_n \mapsto \tau \quad \Gamma \vdash e_i : \tau_1^{i \in 1..n}}{\Gamma \vdash f(e_1, ..., e_n) : \tau}$$

Lista di istruzioni

$$\frac{\Gamma \vdash stmt_1 : notype \quad \Gamma \vdash stmt_2 : notype}{\Gamma \vdash stmt_1; stmt_2 : notype}$$

Return

$$\frac{\Gamma \vdash e : \tau \quad \Gamma \vdash funScope : \tau_1 \quad compatibilita(\tau_1, \tau) = true}{\Gamma \vdash return\ \ e}$$

Operatori Unari

$$\frac{\Gamma \vdash e : \tau_1 \quad optype1(op1, \tau_1) = \tau}{\Gamma \vdash \texttt{op1 e} : \tau}$$

op1	operando	risultato
MINUS	integer	integer
MINUS	float	float
NOT	boolean	boolean

Table 3: Tabella per optype1

Operatori Binari

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2 \quad optype2(op2, \tau_1, \tau_2) = \tau}{\Gamma \vdash e_1 \; op2 \; e_2 : \tau}$$

op	operando1	operando2	risultato
PLUS, MINUS, TIMES, DIV, POW	integer	integer	integer
PLUS, MINUS, TIMES, DIV, POW	integer	float	float
PLUS, MINUS, TIMES, DIV, POW	float	integer	float
PLUS, MINUS, TIMES, DIV, POW	float	float	float
STR_CONCAT	string	string	string
STR_CONCAT	string	integer	string
STR_CONCAT	string	float	string
STR_CONCAT	string	char	string
STR_CONCAT	integer	string	string
STR_CONCAT	float	string	string
STR_CONCAT	char	string	string
AND, OR	boolean	boolean	boolean
GT, GE, LT, LE, EQ, NE	integer	integer	boolean
GT, GE, LT, LE, EQ, NE	integer	float	boolean
GT, GE, LT, LE, EQ, NE	float	integer	boolean
GT, GE, LT, LE, EQ, NE	float	float	boolean
GT, GE, LT, LE, EQ, NE	string	string	boolean
GT, GE, LT, LE, EQ, NE	char	char	boolean

Table 4: Tabella per optype2

4 Test

Table 5: Test Table.

Begin of Table			
TestCase	Descrizione	Input	Oracolo
tc_correct1	Test in cui viene passato un programma NewLang semanticamente corretto	NewLangExemple	Nessuna Eccezione Catturata
tc_correct2	Test in cui viene passato un programma NewLang semanticamente corretto	ExempleProgram	Nessuna Eccezione Catturata
tc_var1	Test dichiearazione multipla di una variabile nello stesso scope.	Test_Var1	Eccezione MultipleDeclaration Catturata
tc_var2	Test in cui viene dichiarato un integer inizializzato con una stringa.	Test_Var2	Eccezione InizializationError Catturata

Continuation of Table 5			
TestCase	Descrizione	Input	Oracolo
tc_var3	Test in cui viene dichiarata una variabile assegnandogli una variabile non dichiarata nel programma.	Test_Var3	Eccezione NoDeclarationError Catturata
tec_decFun1	Test in cui viene dichiarata una funzione più volte	Test_Fun1	Eccezione MultipleDeclaration Catturata
tec_decFun2	Test in cui una funzione di tipo float ritorna una stringa	Tes_Fun2	Eccezione ReturnError Catturata
tec_decFun3	Test in cui è presente una funzione di tipo void che ritorna un float	Test_Fun3	Eccezione ReturnErrorCatturata
tec_decFun4	Test in cui una funzione di tipo float non ritorna nulla	Test_Fun4	Eccezione ReturnErrorCatturata
tec_decFun5	Test in cui è presente una funzione di tipo float che non possiede un return	Test_Fun5	Eccezione ReturnErrorCatturata
tec_decFun6	Test in cui è presente una funzione che nel body possiede una variabile col nome della funzione	Test_Fun6	Eccezione MultipleDeclaration Catturata
tec_decFun7	Test di una funzione che dichiara due volte lo stesso parametro	Test_Fun7	Eccezione MultipleDeclaration Catturata
tec_decFun8	Test di una funzione che possiede un parametro e una variabile nel body con lo stesso identificatore	Test_Fun8	Eccezione MultipleDeclaration Catturata
tc_aritOp1	Test in cui viene passato un programma NewLang in cui ad una stringa viene sommato un intero	Test_AritOp1	Eccezione ArithmeticOpError Catturata
tc_aritOp2	Test in cui viene passato un programma NewLang in cui ad una stringa viene sommato un float	Test_AritOp2	Eccezione ArithmeticOpError Catturata
tc_aritOp3	Test in cui viene passato un programma NewLang in cui ad una char viene sommato un integer	Test_AritOp3	Eccezione ArithmeticOpError Catturata
tc_aritOp4	Test in cui viene passato un programma NewLang in cui ad una char viene sommato un float	Test_AritOp4	Eccezione ArithmeticOpError Catturata
tc_aritOp5	Test in cui viene passato un programma NewLang in cui ad intero viene sommato un booleano	Test_AritOp5	Eccezione ArithmeticOpError Catturata
tc_aritOp6	Test in cui viene passato un programma NewLang in cui ad char viene sommato un booleano	Test_AritOp6	Eccezione ArithmeticOpError Catturata

Continuation of Table 5			
TestCase	Descrizione	Input	Oracolo
tc_assOp1	Test in cui viene passato un programma NewLang in cui nell'assegnazione è presente un identificatore non dichiarato	Test_AssOp3	Eccezione NoDeclarationError Catturata
tc_assOp2	Test in cui viene passato un programma NewLang in cui il numero di identificatori non coincide con il numero di espressioni	Test_AssOp1	Eccezione AssignError Catturata
tc_assOp3	Test in cui viene passato un programma NewLang in cui il numero di identificatori non coincide con il numero di espressioni	Test_AssOp2	Eccezione TypeAssignError Catturata
tc_booleanOp1	Test in cui viene passato un programma NewLang in cui si esegue l'and tra int e bool	Test_BooleanOp1	Eccezione BooleanOpError Catturata
tc_booleanOp2	Test in cui viene passato un programma NewLang in cui si esegue l'and tra float e bool	Test_BooleanOp2	Eccezione BooleanOpError Catturata
tc_booleanOp3	Test in cui viene passato un programma NewLang in cui si esegue l'and tra char e bool	Test_BooleanOp3	Eccezione BooleanOpErrorCatturata
tc_booleanOp4	Test in cui viene passato un programma NewLang in cui si esegue l'and tra string e bool	Test_BooleanOp4	Eccezione BooleanOpErrorCatturata
tc_callFun1	Test in cui viene passato un programma NewLang in cui si chiama una funzione con un numero di parametri errato	Test_callFun1	Eccezione CallFunNumParamError Catturata
tc_callFun2	Test in cui viene passato un programma NewLang in cui si chiama una funzione, passando dei parametri errati	Test_callFun2	Eccezione CallFunTypeParamError Catturata
tc_callFun3	Test in cui viene passato un programma NewLang in chi si chiama una funzione, non dichiarata	Test_callFun3	Eccezione NoDeclarationError Catturata
tc_if1	Test in cui è presente un if che come condizione ha una somma aritmetica	Test_If1	Eccezione ConditionNotValid Catturata
tc_if2	Test in cui è presente un if che ha nella condizione una variabile non dichiarata	Test_If2	Eccezione NoDeclarationError Catturata
tc_readOp1	Test in cui si esegue un operazione di read con una variabile non dichiarata	Test_ReadOp1	Eccezione NoDeclarationError Catturata

Continuation of Table 5			
TestCase	Descrizione	Input	Oracolo
tc_RelOp1	Test in cui si esegue un operazione relazionale tra integer e char	Test_RelOp1	Eccezione RelationalOpError Catturata
tc_RelOp2	Test in cui si esegue un operazione relazionale tra integer e string	Test_RelOp2	Eccezione RelationalOpError Catturata
tc_RelOp3	Test in cui si esegue un operazione relazionale tra integer e boolean	Test_RelOp3	Eccezione RelationalOpError Catturata
tc_RelOp4	Test in cui si esegue un operazione relazionale tra string e char	Test_RelOp4	Eccezione RelationalOpError Catturata
tc_RelOp5	Test in cui si esegue un operazione relazionale tra char e float	Test_RelOp5	Eccezione RelationalOpError Catturata
tc_RelOp6	Test in cui si esegue un operazione relazionale tra char e boolean	Test_RelOp6	Eccezione RelationalOpError Catturata
tc_RelOp7	Test in cui si esegue un operazione relazionale tra boolean e boolean	Test_RelOp7	Eccezione RelationalOpError Catturata
tc_StrOp1	Test in cui si eseguita la concanetazione tra due integer	Test_StrOp1	Eccezione StringOpError Catturata
tc_StrOp2	Test in cui si eseguita la concanetazione tra due float	Test_StrOp2	Eccezione StringOpError Catturata
tc_StrOp3	Test in cui si eseguita la concanetazione tra due char	Test_StrOp3	Eccezione StringOpError Catturata
tc_StrOp4	Test in cui si eseguita la concanetazione tra due boolean	Test_StrOp4	Eccezione StringOpError Catturata
tc_MinusOp1	Test in cui si nega una stringa	Test_MinusOp1	Eccezione MinusOpError Catturata
tc_MinusOp2	Test in cui si nega un char	Test_MinusOp2	Eccezione MinusOpError Catturata
tc_MinusOp3	Test in cui si nega un boolean	Test_MinusOp3	Eccezione MinusOpError Catturata
tc_NotOp1	Test in cui viene applicato il not ad un intero	Test_NotOp1	Eccezione NotOpError Catturata
tc_NotO2	Test in cui viene applicato il not ad un float	Test_NotOp2	Eccezione NotOpError Catturata
tc_NotOp3	Test in cui viene applicato il not ad un char	Test_NotOp3	Eccezione NotOpError Catturata
tc_NotOp4	Test in cui viene applicato il not ad una stringa	Test_NotOp4	Eccezione NotOpError Catturata
tc_while1	Test in cui è presente un while che come condizione ha una somma aritmetica	Test_WhileOp1	Eccezione ConditionNotValid Catturata
tc_while2	Test in cui è presente un whle che ha nella condizione una variabile non dichiarata	Test_WhileOp2	Eccezione NoDeclarationError Catturata
End of Table			