Lab Terminal

Group Members
Ahmad Raza (FA20-BCS-001)
M.Muzammil Hayat (FA20-BCS-055)

Subject
Compiler Construction

Instructor
Mr. Bilal Haider Bukhari

Question # 1

Introduction

Our Compiler project comprises three integral components: the Lexical Analyzer, Syntax Analyzer, and Semantic Analyzer. The Lexical Analyzer, or scanner, initiates the process by converting the input program into a sequence of tokens using regular expressions. Following this, the Syntax Analyzer, or parser, scrutinizes the syntactical structure of the program to ensure adherence to the language's syntax. It constructs a parse tree based on a predefined grammar and the token sequence provided by the Lexical Analyzer. Finally, the Semantic Analyzer evaluates the program's declarations and statements, verifying their semantic correctness in terms of meaning, control structures, and data types. Together, these phases ensure a comprehensive analysis of input programs, encompassing both syntactic and semantic aspects for accurate compilation.

LEXICAL ANALYZER

Lexical Analysis is the first phase of compiler also known as scanner. It converts the input program into a sequence of Tokens.It can be implemented with the regular expressions.

SYNTAX ANALYZER

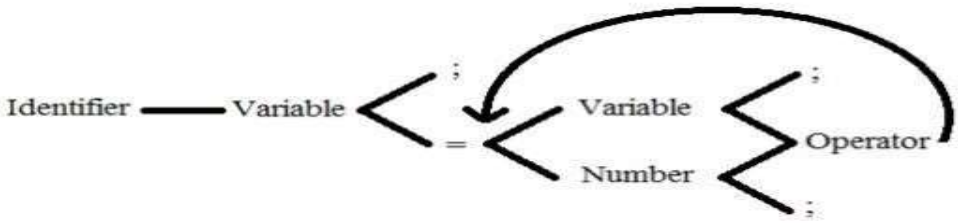
Syntax Analysis or Parsing is the second phase,i.e. after lexical analysis. It checks the syntactical structure of the given input,i.e. whether the given input is in the correct syntax (of the language in which the input has been written) or not. The parse tree is constructed by using the pre-defined Grammar of the language and the input string.

Identifiers	Symbols		Reserved Words
int	+	Operators	for
float	-		while
string	/		if
double	%		do
bool	*		return
char	(	Open Bracket	break
	)	Close Bracket	continue
	{	Open Curly Bracket	end
	}	Close Curly Bracket	
	,	Comma	
	;	Semicolon	
	&&	And	
	||	Or	
	<	Less than	
	>	Greater than	
	=	Equal	
	!	Not	
Variables			

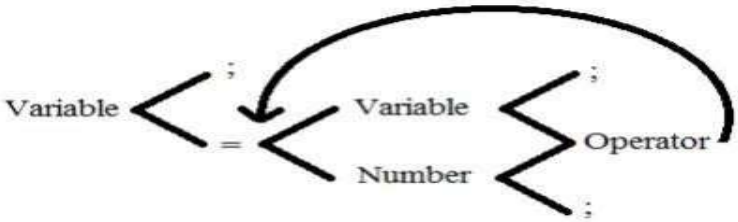
SEMANTIC ANALYZER

Semantic analysis is the task of ensuring that the declarations and statements of a program are semantically correct, i.e,that their meaning is clear and consistent with the way in which control structures and data types are supposed to be used.

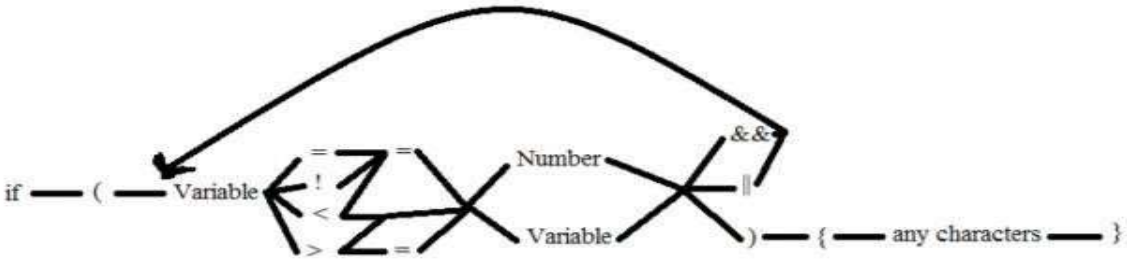
1.


Identifier
Variable
;
=
Variable
Number
Operator
;
;

2.


Variable
;
=
Variable
Number
Operator
;
;

3.


if
(
Variable
=
!
<
>
=
=
Number
Variable
&&
||
)
{
any characters
}