Undefined Language

M.Usman Akram CS191078 **Sameem Abbas** CS191082 **Sundus Saleem** CS191055

I. ABSTRACT

Undefined is a general purpose systems language that uses the idea of having simple, easy to remember keywords and syntax to be developer friendly. It is a statically typed language that has a single keyword for data types, functions and operators that basically perform the same action but have many syntaxes.

e.g: In C++, a string and charArray basically performs the same action but it is the parsing that differentiates the both.

II. Lexical Analyzer

The lexical analysis is independent of the syntax parsing and the semantic analysis. The lexical analyzer splits the source text up into tokens. The lexical grammar describes the syntax of those tokens. The grammar is designed to be suitable for high speed scanning and to make it easy to write a correct scanner for it. It has a minimum of special case rules and there is only one phase of translation.

III. Lexical Unit

Tokens	Regex
main	globlPrnt
loop	untilSatisfied
bool	isValid
input	store
output	console
if	check
else	ifNot
elseif	checkAgain
return	out
import	in
true	Yes
false	No
comment	#([^\\\"|\\.])*\#\
number	[0-9]
charStr	\"([^\\\"|\\.])*\"
arithmetic operator	(-|\+|*|\/|\%)
relational operator	(==|\<|\>|\>=|\<=|\!=)
unary operator	((\+|-|*|\\|\^)(\+|\-))
assignment	\=
letter	[a-zA-Z]
identifier	{letter}+
terminator	\;
other	[0-9][_a-zA-Z]*

IV. Grammar Rules

undefined:

MAIN undefined | LOOP undefined | BOOL undefined | INPUT undefined | OUTPUT undefined | IF undefined | ELSE undefined | ELSEIF undefined | IMPORT undefined | RETURN undefined | YES undefined | NO undefined | NUM undefined | STRING undefined | OPERATOR_A undefined | OPERATOR_R undefined | OPERATOR_U undefined | OPERATOR_E undefined | IDENTIFIER undefined | TERMINATOR undefined | COMMENT undefined | OPEN_BR undefined | OPEN_PR undefined | CLOSE_BR undefined | CLOSE_PR undefined | HEADER_FILE | OTHER | MAIN | LOOP | BOOL | INPUT | OUTPUT | IF | ELSE | ELSEIF | IMPORT | RETURN | YES | NO | NUM | STRING | OPERATOR_A | OPERATOR_R | OPERATOR_U | OPERATOR_E | IDENTIFIER | TERMINATOR | COMMENT | OPEN_BR | OPEN_PR | CLOSE_BR | CLOSE_PR | HEADER_FILE | OTHER

V. Parser Details

The techniques and approach of Recursive Descent Parsing has been used for the language . Recursive-descent parsing is based on the notion of associating each non-terminal with a method. It's kind of a Top-Down Parser. The purpose of each of these procedures is to read a sequence of input characters.

VI. Sample Program

```
in undefinedlib.h
num globlPrnt ( ){
        #start of program#
        console “Hello World” ;
        out 0;
}
```