مجلـــة آفاق للدراسات الإنسانية والتطبيقية كلية الآداب والعلوم الإبيار- جامعة بنغازي
Afaq Journal for Human and Applied Studies Faculty of Arts and Sciences ALAbyar University of Benghazi
ISSUE 3 March 2025 16 رمضان 1445 العـــدد: الثالث مـــارس 2025

Phases of Work Lexical Analysis

Dr.Abdulkream I. Salem [1]
1. Department of Computer Science,Faculty of Science Elmergib University.

DOI: https://doi.org/10.37376/ajhas.vi3.7250 | Publication Date: 16/03/2025 | Acceptance Date:02/10/2024 | Date of Receipt: 28/09/2024

Abstract:

The purpose of the Compiler is generally to translate the program from the source language into the target language, and the Compiler is going through several phases And one of these is a phase Lexical Analysis, and this phase is used in different types in computer applications, including processing important texts, extracting information and Identification of languages required for example C++. This paper focuses on work steps Lexical Analysis to build literal table and identifier table , and build a uniform symbol tabl .

Keywords : Lexical Analysis, Token, Pre-processing,Tokenzation,Token classification , Tken validation, Output generation, C++

مراحل عمل محلل المفردات

د.عبد الكريم إبراهيم سالم [1]
1.محاضر بقسم الحاسوب، كلية العلوم، جامعة الخمس

الملخص:

الغرض من المترجم عمومًا هو ترجمة البرنامج من لغة المصدر إلى اللغة المستهدفة، و المترجم يمر بعدة مراحل وإحدى هذه المراحل هي مرحلة محلل المفردات ، وتستخدم هذه المرحلة في أنواع مختلفة في التطبيقات الحاسوبية، بما في ذلك تجهيز النصوص الهامة واستخراج المعلومات وتحديد اللغات المطلوبة على سبيل المثال السى + . تتركز هذه الورقة على خطوات عمل محلل المفردات لبناء جداول حرفية و تعريف الجداول و بناء رموز الجداول.

الكلمات المفتاحية: محلل المفردات ,الرموز , معالجة المسبقة , المفردات , التصنيف المفردات ، التحقق من صحة المفردات، توليد المخرجات

1.Introduction

The first step in Compiler is Lexical Analyzer , this stage is called the name of the scanner Lexical Analyzer and scan the input without going back and also reads the codes more than once before processing so that the full reading is done. The main task of Lexical Analyzer is to enter character and outputs are token sequence to be used in Lexical Analyzer, (Elgobshawi & Aldawsari, 2022) Each token is a sequency consisting of letters and each letter represents information in the source program, Examples of token is :

Keywords: Consisting of fixed letters e,g "while , for ".

Identifiers : Limited string consisting of letters and numbers. (Dan, Darmi, & Yusof, 2024). Special symbols : as arithmetic symbols

lexeme is A sequency of characters in the source of the programming to match token.

pattern is Structural description Lexical Analysis and the deletion of it gives precise specifications for token and is created Lexical Analysis automatically (T, Devi2, & Aithal, 2020). Figure 1 shows input and output Lexical Analysis

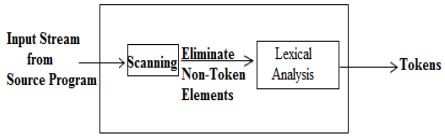


Fig 1 Lexical Analysis

Lexical Analysis scan the string from the source program identifies the of lexeme convert it into tokens. (Oo & Kin, Implementation of Lexical Analysis on Assignment Statements in C++ Programming Language, 2020)

1.1 Token

Each token is a sequence of characters that represents unit of information in the source program.

1.2 Advantage Lexical Analysis generator

It can utilize the best known pattern matching algorithms and there by create efficient Lexical Analysis for people who are not experts in pattern matching techniques . (Budiartha & Ramadhan, 2020)

The following table shows an example of token :

Token	Sample lexems	Informal description pattern
const	const	const
If	If	If
relation	<,<=,=,<>,>,>=	<or<=or=,<>or>=
Id	Pi,count,D2	Letter followed by letters and digits any numeric constant
Num	3.14,0.6.02E23	Any numeric constant
Litteral	Core dumped	Any characters between and except

Table 1 example of tokens

The following table shows an example of Lexical Analysis:

Input : x=x*(acc+123)

Token	Lexemes
Identifier	X
Equal	=
identifier	x
start	*

Left-paren	(
Identifier	acc
Plus	+
Integer	123
Right-paren	)

Table 2 example of Lexical Analysis

2. LITERATURE REVIEW

It shows an overview of how to examine Lexical Analysis and also shows the work of phases Lexical Analysis (Oo & Kyin, 2020). Research of using finite automaton in the modeling of lexical analysis.

The goal of execution structurally programming language EI, for yield Lexical analysis is translation and description lexical and syntactic rules for algorithm and programming description Lexical analysis. (Farooq & Abid , 2016).

The aim of this study is to build phase Lexical Analysis, demonstrate the development of a case Lexical Analysis and also show the construction token recognizer by use DFA by audit suffix. (Ingale, Vayadande, & Verma, 2022)

The computer programming process is done using one of the programming languages and you are done using certain instructions. Therefore, the text of the software must be translated into a series of instructions before the processing process is done by the computer. This leads to the programming itself and the prgraamming calls a compiler, and the text calls the source code. (NUGROHO, 2022). Using the complier phases the translator converts the program written in the Arabic language into the language of the target and translation takes more time than the interpreter. (Gupta, 2021).

This study is focused on the work of phases Lexical analyzer and there are different programs of development Lexical analyzer that have been developed in the past and are in a seamless form of exhaustion with the emergence of multi-core architecture systems. (Kadam, 2023)

The purpose of the algorithm records the risk in the source of the software and in the text file. The processes of the processor and central processing have also been used and improved the efficiency of the algorithm through processed processes and the purpose of this method is to avoid and wait time. (A & Joshi, 2012)

3.Research of Problem

The search problem is focused on :

1- Ascertain the user's information without verifying its identity.

2- The detection program token and send Lexical Analysis without Error.

3- Reduces time during program execution

4.Objective Of Research

Designing a software that shows action steps Lexical Analysis.

5.Methodology

Figure 1 shows work steps Lexical Analysis

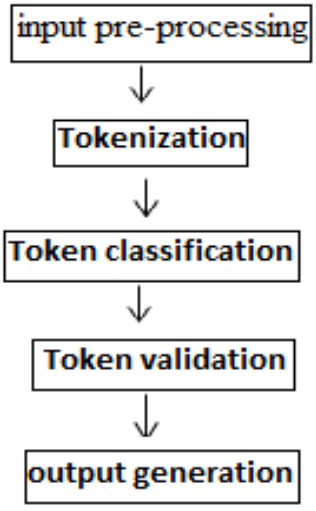


Fig 2 the proposed of system

5.1 Input Pre-processing

Includes removing distances, space and other insignificant letters from the input text.

5.2 Tokenization

In this divides the input text to Series and the letters are re-matched in the input text.

5.3 Token classification

Determine the type lexical for each token for example do symbol, operator , identifier.

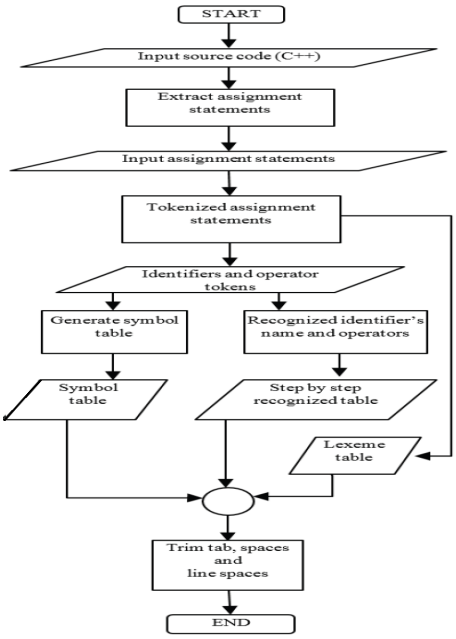
5.4 Token validation

Checks all token is it valid in accordance with programming rules.

5.5 Output Generation

Convert to complier and interpreter.

Figure 3 shows steps of work Lexical Analysis in language C++.



Flow chart Lexical Analysis in C++.

6.Algorithm for Lexical Analysis

The basis of the language he needs is token codes, and token classification and need to design an algorithm and translate the program in order to be done lexical analysis and a language c has been used, to describe symbols and regular expressions, and an algorithm has been applied to clarify Token and is written below.

Algorithm: Token (A)

Where A = Input string.

Output: tokens

Step 1: begin S.

Step 2: Define symbol table.

Step 3: Repeat while scanning (left to right) A

is not completed

i. If blank (empty space)

a. neglect and crack it.

ii. If operator op // arithmetic, relational, etc.

a. Find its type.

b. Write op.

iii. If keyword key // break, if, switch, etc.

a. Write keyword key.

iv. If identifier id // a, b, c, etc

a. Write identifier.

v. If special character sc // (,), etc.

a. Write special character sc.

Step 4: Exit

7. Result

The algorithm token will be applied and it is the next in C++ Version visual studio 2015 , Clarifies the program's inputs are legal or illegal.

surveillance 1:

legal input: for(a=1; a<=10; a++);

Output Analysis:

for : Keyword

(: Special character

a : Identifier

= : Assignment operator

1 : Constant

; : private character

x1 : Identifier

<= : Relational operator

10 : Constant

; : private character

a : Identifier

+ : Operator

+ : Operator

) : private character

; : private character

Tokens generated.

Observation 2:

Invalid input: for(a=0; a<=20a; a++);

Output Analysis:

for : Keyword

(: private character

a : Identifier

= : Assignment operator

0 : Constant

; : private character

x1 : Identifier

<= : Relational operator

Token cannot be generated.

8. CONCLUSION

The focus of this research shows the work Lexical Analysis using the language C++ and through the app it proved that it reduces the time and also clarifies whether the program inputs are legal or illegal and the program is read from left to right. And through the use Lexical Analysis it can't continue processing and that's why you have to use error recovery. Errors are identified when they match token with character sequence.

Refernce

1.A, B., & Joshi, B. K. (2012, December). Parallel lexical analysis on multi-core machines using divide and conquer. University International Conference on Engineering (NUiCONE)

(pp. 1-5). IEEE., p. 5.

2.Budiartha, C. I., & Ramadhan, Y. P. (2020, August 19). LEXICAL ANALYSIS OF THE MOST FREQUENT ACADEMIC WORD IN STUDENTS' ACADEMIC ESSAY. English Language Education Study Program Association, Indonesia, p. 18.

3.Dan, C., Darmi, R. H., & Yusof, M. A. (2024, March 1). The Use Lexical Bundles in English Lanuage Academic Writing Among University Learners :A systematic Literature Review. Faculty of Modern Languages and Communication, p. 15.

4.Elgobshawi, A. E., & Aldawsari, M. (2022, 2 October). LEXICAL DENSITY AS IMPROVEMENT INDICATOR IN THE WRITTEN PERFORMANCE OF EFL MAJORS. 1Department of English, College of Science and Humanities Prince Sattam bin Abdulaziz University, Saudi Arabia, p. 10.

5.Farooq, M. S., & Abid , R. K. (2016, 4 2). A Formal Design for the Lexical and Syntax Analyzer of a Pedagogically Effective Subset of C++. IEEE International Conference on Machine Learning and Applications (ICMLA), 2016, pp. 420-425, doi: 10.1109/ICMLA.2016.0074., p. 8.

6.Gupta, B. (2021, December). Review of Compiler Structure and Processing in Compiler Design. International Journal of Advanced Research in Computer and Communication Engineering, p. 9.

7.Ingale, V., Vayadande, K., & Verma, V. (2022). Lexical analyzer using DFA. International Journal of Advance Research, Ideas and Innovations in Technology, p. 4.

8.Kadam, P. (Composer). (2023). Lexical analyzer. India.

9.NUGROHO, A. (2022, 3). AN ANALYSIS OF LEXICAL AND CONTEXTUAL MEANING IN "CHELSEA FC – THE 5th STAND" APPLICATION. TARBIYAH AND TEACHER TRAINING FACULTY RADEN INTAN STATE ISLAMIC UNIVERSITY LAMPUNG, p. 25.

10.Oo , Z. L., & Kin, M. S. (2020, March 25). Implementation of Lexical Analysis on Assignment Statements in C++ Programming Language. International Journal of Scientific Research in Science, Engineering and Technology, p. 5.

11.Oo, Z. L., & Kyin, M. S. (2020, March 25). Implementation of Lexical Analysis on Assignment Statements in C++ Programming Language. International Journal of Scientific Research in Science, Engineering and Technology (www.ijsrset.com), p. 5.

12.T, V. P., Devi2, A. J., & Aithal, P. S. (2020, December 2). A Systematic Literature Review of Lexical Analyzer Implementation Techniques in Compiler Design. International Journal of Applied Engineering and Management Letters (IJAEML), ISSN: 2581-7000, Vol. 4, p. 17.