Bulletin of Artificial Intelligence
Volume 1, No 1, April 2022 Page: 13−16
ISSN 2962-3944 (media online)
https://journal.grahamitra.id/index.php/buai

Perancangan Perangkat Lunak Teks Editor Bahasa C Menggunakan Metode Lexical Analyzer

Mahmudin Harahap

Fakultas, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: Mahmudinharahap981@gamail.com

Abstrak−Komputer adalah mesin yang dapat melaksanakan seperangkat perintah dasar (instruction set), maka agar komputer dapat melakukan sesuatu hal, kita harus memberinya perintah yang dapat komputer laksanakan, yaitu dalam bentuk kumpulan perintah-perintah dasar tersebut. DOS (Disk Operating System) Programming merupakan nama yang diberikan untuk pemrograman yang dilakukan didalam lingkungan DOS, DOS Programming mengalami masa kejayaan ketika system operasi belum mempunyai GUI yang menarik dan tidak memiliki memori yang besar, Turbo C merupakan salah satu jenis bahasa pemrograman yang banyak digunakan pada saat itu. Lexical analyzer merupakan cara ataupun teknik yang biasa digunakan untuk mengenali bentuk penulisan, bentuk penulisan disini merupakan perintah-perintah yang terdapat dalam suatu bahasa pemrograman, proses pengenalan perintah tersebut tentunya berdasarkan compiler yang ada pada bahasa program yang digunakan, perintah-perintah tersebut akan dibedakan berdasarkan fungsi dari perintah yang ada, misalnya string, variable, function dan lainnya, hal ini digunakan untuk memudahkan programmer membedakan perintah yang ada. Pada metode Lexical analyzer kata disebut dengan token, dimana Lexical analyzer melakukan scanning terhadap karakter yang sesuai dengan token yang sudah ditetapkan. Token yang diidentifikasi dalam penelitian ini adalah kata kerja, dimana kata kerja terdiri dari murni dan kata kerja yang memperoleh imbuhan.

Kata Kunci: Teks; Editor; Lexical; Analyzer

Abstract−A computer is a machine that can carry out a set of basic commands (instruction set), so that a computer can do something, we must give it a command that the computer can carry out, namely in the form of a collection of these basic commands. DOS (Disk Operating System) Programming is the name given to programming carried out in the DOS environment, DOS Programming experienced its heyday when the operating system did not have an attractive GUI and did not have a large memory, Turbo C was one of the most widely used programming languages. at the time. Lexical analyzer is a method or technique commonly used to recognize the form of writing, the form of writing here is the commands contained in a programming language, the process of recognizing these commands is of course based on the compiler in the program language used, these commands will be distinguished by functions of existing commands, such as strings, variables, functions and others, this is used to make it easier for programmers to distinguish existing commands. In the Lexical analyzer method the word is called a token, where the Lexical analyzer scans the characters that match the tokens that have been set. The tokens identified in this study are verbs, where the verb consists of pure and verbs that have affixes.

Keywords: Text; Editor; Lexical; Analyzer

1. PENDAHULUAN

Saat ini, teknologi komunikasi dan informasi berkembang dengan pesat dan memberikan pengaruh besar bagi kehidupan manusia. Contoh dari perkembangan ini adalah bahasa pemrograman, yang pada saat ini telah memungkinkan banyak orang menggunakan bahasa pemrograman dari bahasa pemrograman yang lama sampai bahasa pemrograman yang terbaru serta dari bahasa pemrograman tingkat rendah sampai bahasa pemrograman tingkat tinggi.

Turbo C merupakan bahasa C yang dapat digolongkan ke dalam bahasa level menengah karena bahasa C dikombinasikan dengan elemen-elemen bahasa tinggi dan elemen bahasa rendah. Di dalam bahasa C tersedia hampir di seluruh computer, bahasa C juga memiliki kode yang sifatnya adalah portable dan fleksibel untuk semua jenis komputer. Bahasa C berorientasi pada objek tetapi dapat dipresentasikan oleh mesin dengan cepat. Bahasa memiliki dua buah tipe data yang berfungsi untuk menampung data yang berkoma. Tipe data tersebut adalah float dan double.

Bahasa pemrograman digunakan manusia agar dapat mengendalikan komputer, namun secara dasar komputer hanya mampu memahami bahasa mesin. Sehingga bahasa pemrograman tingkat tinggi dibuat untuk menyederhanakan pemrograman komputer. Beberapa factor penting seseorang dalam memilih bahasa pemrograman adalah syntax, editor, dokumentasi, performa, library, fleksibilitas, komunitas dan popularitas. Banyaknya source code yang harus diperiksa mengakibatkan sulitnya melakukan pemeriksaan serta sulitnya mengukur kredibilitas masing-masing. Kode program terperiksa yang memiliki tingkat similarity (kemiripan) yang cukup tinggi antar kode.

2. METODOLOGI PENELITIAN

2.1 Bahasa C

Boleh dikatakan bahwa akar dari bahasa C adalah bahasa BCPL (Basic Combined Programming Language) yang dikembangkan oleh Martin Richards pada tahun 1967. Kemudian berdasar pada bahasa BCPL ini Ken Thompson yang bekerja di Bell Telephone Laboratories (Bell Labs) mengembangkan bahasa B pada tahun 1970. Saat itu bahasa B telah berhasil diimplementasikan di komputer DEC PDP-7 dengan operating system (OS) UNIX. Pada tahun 1972, peneliti lain di Bell Labs bernama Dennis Ritchie menyempurnakannya menjadi bahasa C. Pada tahun 1978, Dennis Ritchie bersama dengan Brian Kernighan mempublikasikan buku yang kemudian menjadi legenda dalam sejarah perkembangan bahasa C, yang berjudul The C Programming Language. Buku ini diterbitkan oleh Prentice Hall, dan pada saat ini telah

Bulletin of Artificial Intelligence
Volume 1, No 1, April 2022 Page: 13−16
ISSN 2962-3944 (media online)
https://journal.grahamitra.id/index.php/buai

diterjemahkan dalam berbagai bahasa di dunia. Boleh dikatakan bahwa buku ini adalah buku yang paling banyak direfer orang dan dijadikan buku panduan tentang pemrograman bahasa C sampai saat ini. Teknik dan gaya penulisan bahasa C yang merefer kepada buku ini kemudian terkenal dengan sebutan K&R C atau Classic C atau Common C [8].

2.2 Teks Editor

Teks Editor merupakan suatu perangkat lunak (software) aplikasi yang mana memungkinkan penggunanya untuk membuat, mengubah ataupun mengedit file yang berupa plainteks. Editor teks tidak menambahkan pemformatan ke teks, melainkan berfokus kepada fungsi pengeditan untuk teks biasa.

2.3 Lexical Analyzer

Lexical Analysis adalah tahap awal dari sebuah compiler. Tugas utama yang terpenuhi dalam tahap ini akan mengidentifikasi satu rangkaian dari suatu bahasa yang terjadi pada masukan sumber program. Lexical Analyzer berperan sebagai perantara di antara masukan sumber program untuk di-compile dan langkah berikutnya pada compiler. Program masukan dapat dipertimbangkan sebagai urutan karakter. Lexical analyzer mengkonversi runtun karakter ini ke dalam runtun kata-kata yang benar, yang lebih dikenal dengan nama token. Token-token dihasilkan dengan cara memisahkan program sumber tersebut dilewatkan ke parser. Analisis Leksikal harus mengirim token ke parser. Untuk mengirim token, scanner harus mengisolasi barisan karakter pada teks sumber yang merupakan 1 token valid. Parser memeriksa rangkaian token dan mengidentifikasi konstruksi bahasa yang terdapat pada masukan program. Bagaimanapun, parser dan lexical analyzer bekerja sama, dalam arti ketika parser membutuhkan token yang lebih jauh untuk memproses, parser meminta pada lexical analyzer. Sedangkan pada

3. HASIL DAN PEMBAHASAN

Text Editor merupakan perangkat lunak editor berbasis teks, dengan text editor pemakai dapat melakukan penyuntingan ataupun pengolahan teks. Banyak sekali jenis perangkat lunak text editor yang sering digunakan untuk mengolah teks diantaranya adalah Notepad, WordPad, Pico, dan sebagainya. Perangkat lunak di atas merupakan perangkat lunak text editor yang mampu melakukan penyuntingan atau pengolahan kode sumber, dengan menggunakan perangkat lunak text editor ini pemakai dapat melihat dengan jelas setiap kemunculan token seperti keywords (reserved word), identifier, operator.

Siklus ini bersifat sistematik, dengan pendekatan sekuensial untuk membangun perangkat lunak, dimulai pada tingkat sistem berkembang sampai dengan pengetesan, dan perbaikan kesalahan. Model Classic Life Cycle adalah paradigma rekayasa perangkat lunak yang paling luas dipakai dan paling tua. Paradigma model ini memiliki tempat yang terbatas namun penting di dalam kerja rekayasa perangkat lunak.

Penerapan Lexical Analyzer dalam aplikasi ini terdapat pada proses pencacahan perintah-perintah yang diketikkan, setiap perintah yang diketikkan pasti mempunyai jenis string yang berbeda, seperti string statement, string command, string comment, string operator dan lainnya.

3.1 Penerapan Metode Lexical Analyzer

Analisis Leksikal (Scanner) melakukan pemeriksaan terhadap kode sumber dengan perubahan status yang terjadi. Dalam penerapannya, suatu kode sumber akan diuraikan menjadi simbol-simbol tertentu yang disebut token. Scanner berperan sebagai antarmuka antara source code dengan proses analisis sintaksis (parser). Scanner akan memeriksa setiap karakter dari kode sumber. Analisis leksikal berkaitan dengan penulisan bahasa pemrograman (dalam penelitian yaitu bahasa C). Membaca source program (input characters) dan mengasilkan output berupa rangkaian token. Berikut ini merupakan tabel dari token, lexeme, pattern.

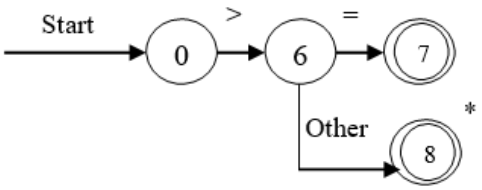
Tabel 1. Token, Lexeme, Pattern

Token	Lexeme	Pattern
Const_sym	Const	Const
If_Sym	If	If
Relation	<,<=,=,<>,>,>=	< atau <= atau = <> atau > atau >=
Id	pi, count, d2	Letter diikuti oleh letters atau digits
Num	3, 14, 0, 6, 0E 10	Sembarang konstanta angka
Literal	“Hallo”	Sembarang karakter “diantara” dan “kecuali”

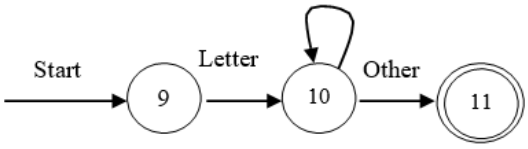
Pada pengenalan token yang digunakan lexical analyzer dapat di lihat seperti di bawah ini.
Stmt if expr then stmt
If expr then stmt else stmt e
expr term relop term term

Bulletin of Artificial Intelligence
Volume 1, No 1, April 2022 Page: 13−16
ISSN 2962-3944 (media online)
https://journal.grahamitra.id/index.php/buai

term id
num
Dimana terminal if, then, else, relop, id dan num membentuk himpunan dari rangkaian yang diberikan oleh RD berikut :
if
then then
else else
relop < | <= | = | <> | >| > = id letter (leter | digits) *
num digit (.digit) ? (E(+ | -)? digit)? Transisi diagram untuk pola >= dan >



Gambar 1. Transisi Diagram Transition diagram identifier dan keyword



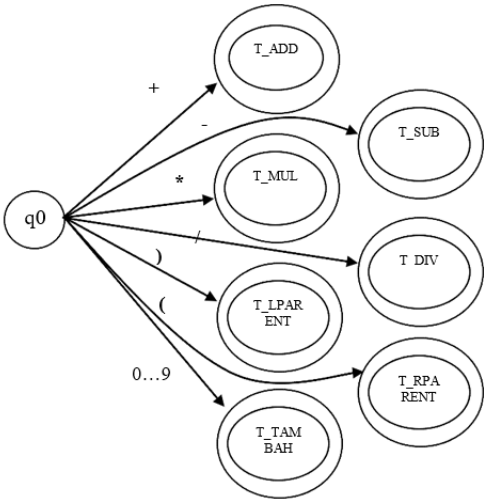
Gambar 2. Transisi diagram identifier dan keyword

Untuk memperoleh token dan nilai atribut. Prosedur install_id berguna untuk akses ke input buffer dimana lexeme diketemukan dan mengembalikan pointer pada entry.

1. Scanner
   Scanner bertugas melakukan analisis leksikal, yaitu mengidentifikasi semua besaran yang membangun suatu bahasa pada suatu program sumber. Scanner adalah bagian dari kompilator yang merima input berupa stream karakter kemudian memilah program sumber menjadi satuan leksik yang disebut dengan token. Token ini akan menjadi input bagi parser.
2. Besar Leksik
   a. Identifier
      Bisa berupa keywords atau nama. Keywords atau kata kunci yang sudah didefinisikan oleh suatu bahasa seperti BEGIN, END, IF, ELSE di dalam
      Pascal. Nama dideklarasikan sendiri oleh pemakai, seperti nama sebuah variabel misalnya. Contoh bila pada suatu program Pascal terdapat deklarasi :
      Var
      Nomor : Integer; suhu : Real;
      Maka Nomor dan Suhu akan dikenali sebagai besaran leksik berupa nama variabel yang terdapat pada program tersebut. Sedangkan VAR, INTEGER, dan REAL merupakan Keyword pada program tersebut. Nama bisa berupa nama program, procedure, var, type, constant.
   b. Nilai Konstanta
      Adalah suatu konstanta yang terdapat pada program. Bisa berupa konstanta integer, real, boolean, character, string, dan sebagainya. Misalkan saja dalam Pascal pada suatu program terdapat statement.
      N : = R + 5 * 10
      Kata : = kata1 + ‘makan’ A : =0.333
      Selesai := True
      Kode sumber pada kompiler memerlukan BNF dan diagram sintaks agar pembuat program mudah dalam membuat program. Adapun beberapa rancangan BNF dari kompiler yang dibuat adalah :<simple_exp> ::= <term> { <add_operator> <term> }
      <add_operator> ::= T_ADD | T_SUB
      <term> ::= <signed_fact> { <mul_operator> <signed_fact> }
      <signed fact> ::= <add_operator> <fact> | <fact>
      <mul_operator> ::= T_MUL | T_DIV
      <fact> ::= T_LPARENT <exp> T_RPARENT | T_NUMERIC
      Dimana : T_ADD = ’+’ T_SUB = ’-‘ T_MUL = ’*’
      T_DIV = ’/‘ T_LPARENT = ’(‘ T_RPARENT = ’)’ T_ANGKA = ’0’..’9’

Dalam memeriksa sintaks dari kode sumber, parser membutuhkan bantuan scanner untuk memberikan token dari sintaks. Berdasarkan BNF dan diagram sintaks yang telah dibuat.

Bulletin of Artificial Intelligence
Volume 1, No 1, April 2022 Page: 13−16
ISSN 2962-3944 (media online)
https://journal.grahamitra.id/index.php/buai



Gambar 3. Diagram Sintaks

4. KESIMPULAN

Sebagai penutup pembahasan kesimpulan yang penulis peroleh yakni hasil pengujian telah berhasil dilakukan dengan menggunakan Lexical Analyzer dengan menggunakan bahasa C. Proses kompiler bekerja dengan memproses code yang diinputkan sehingga prosesnya dapat bekerja dengan efisien. Hasil dari aplikasi ini digunakan sebagai acuan untuk pembuatan teks editor

REFERENCES

[1] Budiman, 2017, "Pembangunan Perangkat Lunak Puniedit (Perangkat Lunak Editor Untuk Multi Language Programming)", Vol. 1, No. 1
[2] Muktiadi, 2014, "Analisis Leksikal Untuk Mengidentifikasi Kata Kerja", Jurnal JUITA, Vol. III, No. 1 ISSN : 2086-9398
[3] Gun gun Febriansyah, "Kompiler Untuk Pemrograman Dalam Bahasa Indonesia" Jurnal KOMPUTA, Vol. 2, No. 1, ISSN : 2089-9033.
[4] Roger S. Pressman, 2002, "Rekayasa Perangkat Lunak Pendekatan Praktisi", Penerbit Andi, Yogyakarta.
[5] Jan Wantoro, 2017, "Algoritma & Struktur Data Dalam Bahasa C/C++", Penerbit : Muhammadiyah University press
[6] Daya Kurniawan, 2016, "Pemrograman Dengan Bahasa C", Penerbit : Elex Media Komputindo, Jakarta