Nama dan NIM Anggota Kelompok

- 13511071 Setyo Legowo
- 13511083 Fawwaz Muhammad

Kelas, Mata Kuliah, dan Tugas

Kelas 1 – IF2052 Teori Bahasa dan Otomata

Tugas 2: Precompiler

Deskripsi Persoalan

Buatlah sebuah aplikasi untuk mengenali apakah sintaks sebuah program sederhana itu benar atau tidak. Aplikasi akan membuka file yang berisi sintaks program dan kemudian memeriksa apakah semua sintaks pada program tersebut valid atau tidak. Jika tidak valid maka aplikasi akan mengeluarkan informasi baris yang menyebabkan ketidakvalidan sintaks program.

Batasan Masalah

Program sederhana yang dimaksud hanya memiliki algoritma, tanpa pernyataan kamus atau judul program. Tipe variabel hanya integer dan operasi terhadap variabel hanya berupa operasi matematika untuk integer.

Sintaks algoritma yang dimiliki program terdiri atas:

1. Begin – end sebagai batas awal dan akhir dari program
2. If – then {begin end}
3. If – then – {begin end} – else {begin end}
4. Repeat – until
5. While – do {begin end}
6. Kondisi (dengan operator: <, >, <=, >=, =, <>)
   Contoh: a >= b
7. Assignment (hanya operasi matematika untuk integer yaitu +, -, *)
   Contoh: a = b + c
8. Input (variabel)
   Contoh: input(z)
9. Output (variabel) atau Output (operasi matematika)
   Contoh: output(z)
10. Komentar yang ditandai dengan kurung buka (“{“) dan kurung tutup (“}”)

Jawab Persoalan

Telah dibuat sebuah aplikasi yang dapat memeriksa bahasa yang telah didefinisikan pada batasan masalah. Bahasa pemrograman yang digunakan untuk mengimplementasikan persoalan diatas yaitu bahasa C. Dengan menggunakan algoritma CYK maka dapat memudahkan implementasi CFL. Namun masih memungkinkan untuk tidak dapat memvalidasi baris dari batasan masalah yang tidak valid.

Tata Bahasa yand Digunakan / Diimplementasikan

Context Free Grammars yang dikonversi ke bentuk Chomsky Normal Form

CFG

```
A -> bDe
D -> VE | I | VO | VS
E -> i(H)tI | i(H)tbDe | i(H)tIcI | i(H)tbDecI | i(H)tIcbDe | i(H)tbDecbDe
H -> JKJ
I -> L | M | N | VL | VM | VN
J -> [Operasi +, -, * dengan hirarkinya]
K -> < | > | <= | >= | = | <>
L -> W=J
M -> j(W)
N -> k(W)
O -> rDu(H)
S -> w(H)dI | w(H)dbDe
V -> { ? }
W -> [Pemeriksaan Variabel]

[TERMINAL]
b -> 'begin'
e -> 'end'
i -> 'if'
t -> 'then'
r -> 'repeat'
u -> 'until'
c -> 'else'
w -> 'while'
d -> 'do'

j -> 'input'
k -> 'output'
```

CNF

```
[SINGLE]
       [VARIABEL REGISTERED]
       A B C D E F G H I J K L M . O P . . S . . V W . . Z

       [NON-TERMINAL]
       S -> A,B | A,C <-- begin sesuatu atau begin end
       B -> D,C
       D -> D,D | F,J | E,K | W,O | G,L | H,M | I,P <-- atau fungsi sesuatu

       [TERMINALS]
       A -> 'begin'
       C -> 'end'
       E -> 'input'
       F -> 'output'
       G -> 'if'
       H -> 'repeat'
       I -> 'while'

[EXTENDED 2]
       [VARIABEL REGISTERED]
       J K L M O V W
       F0 F1 F2
```

```
I0
J0 J1
K0 K1
L0
N0 N1 N2
O1 O4 O5 O6

[NON-TERMINAL]
       [OUTPUT]
       J  -> O4,J1 | O4,J  <-- buka kurung sesuatu atau buka kurung buka kurung
       J1 -> F2,O5 | W ,O5 | N0,O5
              <-- (bilangan atau variabel atau operasi matematika) tutup kurung

       [INPUT]
       K  -> O4,K1 <-- buka kurung (sebelum variabel)
       K1 -> W ,O5 | F0, O5 <-- variabel tutup kurung

       [ASSIGNMENT]
       O  -> O6,N0 | O6,F2 | O6,W
              <-- sama dengan (operasi matematika atau bilangan atau variabel)

       [IF THEN ELSE]
       L  -> O4,I0
       I0 -> L0,I1
       I1 -> O5,I2
       I2 -> O8,I3 | O8,I4 | O8,S
       I3 -> W ,O  | F ,J  | E ,K
       I4 -> I3,I5
       I5 -> O9,I3 | O9,S

       [REPEAT UNTIL]
       M  -> D,M0
       M0 -> OA,M1
       M1 -> O4,M2
       M2 -> L0,O5

       [WHILE DO]
       P  -> O4,P0
       P0 -> L0,P1
       P1 -> O5,P2
       P2 -> OB,P3 | OB,S
       P3 -> W ,O  | F ,J  | E ,K

       [MATHEMATIC OPERATION]
       N0 -> F2,N3 | W ,N3 | O4,N2
       N2 -> N0,O5
       N3 -> O1,N0 | O1, W | O1,F2

       [LOGIC OPERATION]
       L0 -> W ,L1 | F2,L1 | N0,L1
       L1 -> O7,W  | O7,F2 | O7,N0

       [COMMENT]
       V  -> O2,V0
       V0 -> F3,O3

       [VARIABEL DETECTION]
       W -> F0,F1

[TERMINALS]
O1 -> '+' | '-' | '*'
O2 -> '{'
O3 -> '}'
```

```
O4 -> '('
O5 -> ')'
O6 -> '='
O7 -> '=' | '<' | '>' | '<=' | '>=' | '<>'
O8 -> 'then'
O9 -> 'else'
OA -> 'until'
OB -> 'do'

[FUNCTIONS]
F0 -> [IsAlpha]
F1 -> [IsAlphaNumeric]
F2 -> [IsNumber]
F3 -> [IsValidChar]
```

Source Code

File: main.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>

#include "boolean.h"
#include "cnf.h"
#include "mesinkar.h"
#include "mesinkata.h"

// Definisi prosedur fungsi
boolean IsFileExist(char * namaFile);
void ArrayCharCopy(char * st1, char * st2);

// Definisi Variabel Global
unsigned int line_num;
extern char CC;
extern Kata CKata;
extern boolean EndKata;
extern char Nama_File[100];

int main(int argc, char **argv)
{
    if(argc > 1)
    {
        if(IsFileExist(argv[1]))
        {
            ArrayCharCopy(argv[1],Nama_File);
            line_num = 1;

            ReadFile();
        } else {
            printf("\nPERHATIAN: Tidak ditemukan nama file \"%s\"\n",argv[1]);
        }
    } else {
        printf("\nPERHATIAN: Nama file tidak terdefinisi.\n");
    }

    return 0;
}

void ArrayCharCopy(char * st1, char * st2)
{
    int i;
    for(i=0; st1[i] > ' '; i++)
    {
        st2[i] = st1[i];
    }
}
```

```
boolean IsFileExist(char * namaFile)
{
    struct stat st;
    FILE *fp;

    if(stat(namaFile,&st) != 0)
        return 0;
    else
        return 1;
}
```

File: cnf.c

```
#include "boolean.h"
#include "mesinkata.h"
#include "cnf.h"

extern unsigned int line_num;
extern boolean error_found;
extern Kata CKata;
extern Kata listTerminal[MaxTerminal];
extern boolean EndKata;
static int N = 0;
static int TerminalLine[MaxTerminal];

void ReadFile() {
    int i;
    int line = 0;

    STARTKATA();
    //system("pause");
    i = 0;
    while(!EndKata) {
        //CetakKata(CKata);
        CopyKata(CKata,&listTerminal[i]);
        //CetakKata(listTerminal[i]);
        TerminalLine[i] = line_num;
        ADVKATA();
        //printf("\n");
        //system("pause");
        i++;
        N += 1;
    }
    CLOSE();

    /*
    *
    *@author : Fawwaz Muhammad
    *
    */

    // Read terminals
    /*
    * Format file TERMINAL
    * harus dimulai dengan sebuah angka yang menyatakan jumlah TERMINAL yang ada dalam file
tersebut
    * lalu setiap variabel harus berupa angka integer
    * bentuk konversi dari penulisan umum ke format file ini adalah sbb:
    * format penulisan umum :
    *   A->a B->c
    * format penulisan file
    *   A : a
    *   B : c
    * sehingga contoh lengkapnya isi filenya adalah sbb
    * CONTOH:
    *   2
```

```
    *   A : a
    *   B : c
    * artinya file diatas hanya memiliki 2 jenis terminal, yaitu A->a dan B->c
    *
    **/
    FILE *terminalfile;
    int jumlahterminal;
    terminalfile= fopen("terminal.txt","r");
    fscanf(terminalfile,"%d\n", &jumlahterminal);

    int Terminal[jumlahterminal];
    // char *Terminal[jumlahterminal];
    char arti[jumlahterminal][15];


    int z;
    for(z=0; z<jumlahterminal; z++){
        fscanf(terminalfile, "%d : %s\n", &Terminal[z], &arti[z]);
        //printf("%d : %s\n", Terminal[z], &arti[z]);
        //system("pause");
    }
    fclose(terminalfile);

    //printf("%d : %s\n", Terminal[1], &arti[1]);
    FILE *nonterminalfile;

    /* Format file productions ( rule / grammar)
    * harus dimulai dengan sebuah angka yang menyatakan jumlah Rule yang ada dalam file
tersebut
    * lalu setiap variabel harus berupa angka integer
    * bentuk konversi dari penulisan umum ke format file ini adalah sbb:
    * format penulisan umum :
    *   A->BC | CC
    * format penulisan file
    *   A : B , C
    *   A : C , C
    * sehingga contoh lengkapnya isi filenya adalah sbb
    * CONTOH:
    *   2
    *   A : B , C
    *   A : C , C
    * artinya file diatas hanya memiliki 2 rule, yaitu A->BC dan A->CC
    *
    */
    nonterminalfile = fopen("nonterminal.txt", "r");
    int JumlahNonTerminal;
    fscanf(nonterminalfile,"%d\n", &JumlahNonTerminal);

    //Readrules
    int From[JumlahNonTerminal];
    int To1[JumlahNonTerminal];
    int To2[JumlahNonTerminal];

    for(z=0; z<JumlahNonTerminal; z++){
        fscanf(nonterminalfile, "%d : %d , %d\n", &From[z], &To1[z], &To2[z]);
        //printf("%d : %d , %d \n", From[z], To1[z], To2[z]);
    }


    fclose(nonterminalfile);
```

```
    //  CCC Y   Y K  K
    // C     Y Y  K K
    // C      Y   KK
    // C      Y   K K
    //  CCC   Y   K  K
    /*
    *
    * get from wikipedia translated from pesudocode by Fawwaz
    *
    */

    // setting up
    Kata ArrayOfArti[jumlahterminal];
    for (i = 0; i < jumlahterminal; ++i)
    {
        Kata temp=KataMaker(arti[i]);
        CopyKata(temp,&ArrayOfArti[i]);
    }

    // configuration
    int Inputcounter=N;
    int NonTerminalcounter=JumlahNonTerminal;
    int Terminalcounter=jumlahterminal;
    int StartVariable=0;
    int p,q,r,s,t; // dummy vairable for looping
    boolean MatriksCYK[Inputcounter][Inputcounter][NonTerminalcounter];

    for (p = 0; p < Inputcounter; ++p)
    {
        for (q = 0; q < Inputcounter; ++q)
        {
            for (r = 0; r < NonTerminalcounter; ++r)
            {
                MatriksCYK[p][q][r]=false;
            }

        }
    }
    boolean wasSelected;
    for (p = 0; p < Inputcounter; ++p){
        wasSelected = false;
        for (q = 0; q < Terminalcounter; ++q){
            if (IsKataSama(listTerminal[p],ArrayOfArti[q]))
            {
                MatriksCYK[0][p][Terminal[q]]=true;
                printf("koordinat di %d - %d - %d adalah ", 0,p,Terminal[q]);
                CetakKata(listTerminal[p]);
                printf(" dibandingkan dengan ");
                CetakKata(ArrayOfArti[q]);
                printf("\n");
                system("pause");
                wasSelected = true;
            } else if((IsKataSama(KataMaker("IsAlphaNumeric"),ArrayOfArti[q])) &&
(!wasSelected)) {
                if(IsVariabel(p)) {
                    MatriksCYK[0][p][33]=true;
                    printf("koordinat di %d - %d - %d adalah ", 0,p,33);
                    CetakKata(listTerminal[p]);
                    printf(" (terbaca sebagai variabel) dibandingkan dengan ");
                    CetakKata(ArrayOfArti[q]);
                    printf("\n");
                    system("pause");
                } else if(IsBilangan(p)) {
                    MatriksCYK[0][p][54]=true;
                    printf("koordinat di %d - %d - %d adalah ", 0,p,54);
                    CetakKata(listTerminal[p]);
                    printf(" (terbaca sebagai bilangan) dibandingkan dengan ");
                    CetakKata(ArrayOfArti[q]);
                    printf("\n");
```

```
                    system("pause");
                }
            }
        }
    }


    for (p = 2; p <= Inputcounter; p++) // i
    {
        printf("Ketinggian sekarang %d\n", p);
        for (q = 1; q <= Inputcounter-p + 1; q++) // j
        {
            printf("koordinat kanan ke %d\n", q);
            for (r = 1; r <= p - 1; r++) //
            {
                printf("A: (%d,%d) ? B: (%d,%d)\n", r - 1,q - 1,p-r-1,q+r - 1);
                for (s = 1; s <= NonTerminalcounter; s++) // RULE ke berapa
                {
                    if ((MatriksCYK[r-1][q-1][To1[s-1]]==true) && (MatriksCYK[p-r- 1][q+r -
1][To2[s - 1]]==true))
                    {
                        MatriksCYK[p-1][q-1][From[s-1]]=true;
                        printf(" TRUE at (%d,%d) ",p-1,q-1);
                        system("pause");
                        line = p - 1;
                    }
                }
            }
        }
    }
    if (MatriksCYK[Inputcounter-1][0][0]==true)
    {
        printf("MASUKAN VALID UNTUK DI COMPILE (LULUS UJI) \n");
    }else{
        printf("MASUKAN TIDAK LULUS COMPILE ADA KESALAHAN DI LINE : %d\n",TerminalLine[line] +
1);
    }
}

/* ---------------------------------------------------------------------------- */
/* --------------------------------- TERMINAL --------------------------------- */
/* ---------------------------------------------------------------------------- */

boolean IsBilangan(int no) {
    int i = 0;
    while(listTerminal[no].Length > i) {
        if(IsNumber(listTerminal[no].TabKata[i]))
            i++;
        else
            break;
    }
    return ((listTerminal[no].Length == i) && (i != 0));
}
boolean IsVariabel(int no) {
    if(listTerminal[no].Length == 0)
        return false;
    else {
        int i = 0;
        if(IsAlphaChar(listTerminal[no].TabKata[i])) {
            i++;
            while(listTerminal[no].Length > i) {
                if(IsAlphaNumeric(listTerminal[no].TabKata[i]))
                    i++;
                else
                    break;
            }
            return (listTerminal[no].Length == i);
        } else return false;
    }
}
```

Contoh Masukan dan Keluaran

Contoh 1:

```
begin
end
```

```
Membaca file: dummy.txt

MASUKAN VALID UNTUK DI COMPILE (LULUS UJI)
```

Contoh 2:

```
begin

```

```
Membaca file: dummy.txt

MASUKAN TIDAK LULUS COMPILE ADA KESALAHAN DI LINE: 2
```

Contoh 3:

```
begin
      input(7)
end
```

```
membaca file: dummy.txt

MASUKAN VALID UNTUK DI COMPILE (LULUS UJI)
```

Contoh 4:

```
begin
   input(x)
   output(x)
end
```

```
Membaca file: dummy.txt

MASUKAN VALID UNTUK DI COMPILE (LULUS UJI)
```

Contoh 5:

```
begin
   input(x)
   if (x < 7) then
      output(x)
end
```

```
Membaca file: dummy.txt

MASUKAN VALID UNTUK DI COMPILE (LULUS UJI)
```

Contoh 6:

```
begin
```

```
   input(x)
   if (x < 7) Then
   begin
      output(x)
      output(0)
   end
end
```

```
Membaca file: dummy.txt

MASUKAN VALID UNTUK DI COMPILE (LULUS UJI)
```

Contoh 7:

```
begin
   input(x)
   if (x < 7) Then
   begin
      output(x)
      output(0)
end
```

```
Membaca file: dummy.txt

MASUKAN TIDAL LULUS COMPILE ADA KESALAHAN DI LINE : 8
```

Contoh 7:

```
begin
   input(x)
   while (x > 7) do
      output(x - 7)
end
```

```
Membaca file: dummy.txt

MASUKAN TIDAK LULUS COMPILE ADA KESALAHAN DI LINE : 8
```

Contoh 8:

```
begin
   input(x)
   while(x > 7) do
      output(x)
end
```

```
Membaca file: dummy.txt

MASUKAN VALID UNTUK DI COMPILE (LULUS UJI)
```