

LAPORAN TUGAS BESAR II
TEORI BAHASA FORMAL DAN OTOMATA (TBFO)
IF2220

***APLIKASI CFG DAN PDA PADA PENGENALAN
EKSPRESI MATEMATIKA***



DIBUAT OLEH:

T. Andra Oksidian Tafly / 13517020

Andrian Cedric / 13517065

Jan Meyer Saragih / 13517131

INSTITUT TEKNOLOGI BANDUNG
TEKNIK INFORMATIKA
2018

BAB 1

DESKRIPSI PERSOALAN

Kalkulator adalah sebuah perangkat yang dapat digunakan untuk menghitung angka-angka dengan membubuhkan operan-operan yang disediakan. Kalkulator standar dapat digunakan untuk melakukan penjumlahan, pengurangan, pembagian, maupun perkalian pada angka-angka yang dimasukkan. Selain itu, ada juga kalkulator saintifik yang dapat digunakan untuk melakukan operasi-operasi yang lebih rumit, seperti operasi perpangkatan, trigonometri, kalkulus, matriks, maupun bilangan kompleks.



Pada tugas kali ini, kami diminta untuk membuat sebuah kalkulator sederhana yang dapat dioperasikan melalui *command prompt* maupun terminal dengan menerapkan prinsip tata bahasa bebas kompleks (CFG) dan/atau *pushdown automata* (PDA). Pengguna akan memasukkan *input* dengan format yang sama dengan kalkulator saintifik (pengguna memasukkan angka dengan diikuti oleh operan-operan yang diinginkan, seperti “+”, “-“, dan sebagainya) lalu program akan

menerjemahkannya menjadi bilangan beserta operannya, lalu dihitung untuk dikeluarkan hasil perhitungannya.

Kalkulator sederhana ini diharapkan mampu melakukan operasi penjumlahan, pengurangan, perkalian, pembagian, dan perpangkatan serta melakukan operasi dengan bilangan berpecahan desimal dan tanda kurung. Jikalau sintaks yang dimasukkan tidak sesuai, misal operan ganda yang berdempetan ataupun penggunaan operan yang tidak tepat, kalkulator akan mengeluarkan pesan “SYNTAX ERROR”. Di sisi lain, jikalau operasi hitung menghasilkan hasil yang tidak terdefinisi, misal 1 dibagi 0, kalkulator akan mengeluarkan pesan “MATH ERROR”.

BAB 2

CONTEXT FREE GRAMMAR YANG DIPAKAI

Berikut adalah tata bahasa bebas konteks (CFG) yang dipakai dalam program kalkulator kami:

$$S \rightarrow T \mid S + T \mid S - T$$

$$T \rightarrow P \mid T * P \mid T / P$$

$$P \rightarrow C \mid C^P$$

$$C \rightarrow D \mid (S) \mid -D \mid -(S)$$

$$D \rightarrow E \mid E.E$$

$$E \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

Dengan baris atas adalah baris untuk penjumlahan dan pengurangan, lalu diikuti dengan perkalian maupun pembagian, diikuti dengan perpangkatan, diikuti dengan tanda kurung beserta tanda-tandanya, lalu diakhiri dengan angka (digit). Penjelasan dari *context free grammar* akan dijelaskan lebih lanjut pada bab selanjutnya.

BAB 3

PENJELASAN STATE DAN AKSI

Dalam pembuatan kalkulator sederhana ini, kami memodelkan pengoperasian menggunakan tata bahasa bebas konteks (CFG). Program kami akan menerima input dari pengguna dalam bentuk *string*, kemudian *string* akan diolah sesuai dengan urutan pengoperasian yang telah disepakati (diutamakan perpangkatan dan tanda kurung, diikuti dengan perkalian dan pembagian, serta diakhiri dengan penjumlahan dan pengurangan).

Program akan menelusuri setiap karakter pada *string* yang dimasukkan oleh pengguna. Saat program menemukan operasi penjumlahan ataupun pengurangan, maka program akan menelusuri kembali *string* yang tersisa untuk mencari operasi pengalian maupun pembagian. Jika tidak ada, program akan langsung menghitung operasi pengurangan atau penjumlahan tersebut. Akan tetapi, saat program menemukan adanya operasi pengalian maupun pengurangan, program akan mencari pada *string* yang tersisa apakah ada operasi perpangkatan maupun tanda kurung. Jikalau tidak ada, maka program akan menghitung operasi pembagian atau pengalian dan hasilnya akan dipakai sebagai sebuah angka yang akan dijumlahkan atau dikurangkan oleh operasi penjumlahan maupun pengurangan. Akan tetapi, jika program menemukan perpangkatan maupun tanda kurung, program akan memprioritaskan penghitungan pada operasi perpangkatan maupun tanda kurung. Lalu hasil dari operasi akan dipakai untuk dihitung dengan operasi-operasi yang berkasta lebih rendah.

Urutan penggunaan *context-free grammar* dapat dilihat pada bab sebelumnya, yaitu urutan dari pengoperasian *context-free grammar*. Pejumlahan dan pengurangan diletakkan paling atas, kemudian dilanjutkan dengan perkalian dan pembagian, diikuti dengan perpangkatan, tanda kurung dan minus, tanda desimal, dan diakhiri oleh angka (digit).

Selain melakukan perhitungan seperti biasa, program ini juga mengecek apakah *string* yang kita masukkan valid atau tidak. Ada dua jenis kesalahan yang dapat terjadi dalam program kalkulator kami, yaitu galat matematika (MATH_ERROR) dan galat sintaks (SYNTAX_ERROR). Galat matematika terjadi ketika masukka yang kita berikan tidak dapat dihitung karena melanggar teori-teori dan konsep matematika yang ada, misal $1/0$. Kemudian, pada glat intaks, program menunjukkan pesan kesalahan jikalau masukkan yang diberikan tidka sesuai dengan kaidah operasi matematika, misal tanda kurung yang tidak diakhiri dengan tutup maupun penempatan operan yang berdempetan, seperti $2++3$.

BAB 4

KODE SUMBER

Dalam pembuatan kalkulator sederhana ini, kami menggunakan bahasa C untuk mengimplementasikan kalkulator-kalkulator yang ada.

4.1. Source Code untuk Operasi Kalkulator

```
#include
<stdio.h>

#include <math.h>
#include "boolean.h"

// Asumsi hasil tidak mencapai Fail dan MathError
#define Fail -9999999

char *now;
char *start;
boolean syntaxerror;
boolean matherror;

// List perintah
boolean IsDigit(char c);
float charTofloat(char c);
boolean FailedInput();
boolean CanBeInteger(float x);

float checksign();
float digit();
float plusminus();
float timesdiv();
float brackets();
float power();
float executePower(float x);

// Implementasi perintah-perintah

boolean IsDigit(char c)
// Mengembalikan nilai true jika char tersebut adalah digit
{
    return (c >= '0' && c <= '9');
}

float charTofloat(char c)
// Mengganti karkater ke float
```

```

{
    return (c - '0');
}

boolean FailedInput()
// Memberikan nilai true jika x adalah Fail
{
    return (syntaxerror || matherror);
}

boolean CanBeInteger(float x)
// Memberikan nilai true jika x dapat diconvert menjadi integer
{
    return (ceil(x) == x);
}

float checksign()
// Mengecek apakah digit tersebut memiliki nilai negatif, tanda kurung
// atau hanya bilangan biasa saja
{
    float checker;

    if (*now == '-')
    {
        if (now == start)
        {
            now++;
            checker = digit();
        }
        else
        {
            now--;
            if (*now == '(')
            {
                now++;now++;
                checker = digit();
            }
            else
            {
                checker = Fail;
                syntaxerror = true;
            }
        }
    }
    if (!FailedInput())
    {
        return ((-1) * checker);
    }
}

```

```

    }
    else
    {
        return (Fail);
    }
}
else if (*now == '(')
{
    now++;
    return (brackets());
}
else
{
    return (digit());
}
}

float digit ()
// Menghasilkan digit angka pada bilangan
{
    float temp = 0;
    if (IsDigit(*now))
    {
        while (IsDigit(*now))
        {
            temp = (temp*10) + charTofloat(*now);
            now++;
        }
        if (*now == '.')
        {
            now++;
            float multiplier = 0.1;
            if (!IsDigit(*now))
            {
                syntaxerror = true;
                return (Fail);
            }
            else
            {
                while (IsDigit(*now))
                {
                    temp = temp + charTofloat(*now)*multiplier;
                    multiplier /= 10;
                    now++;
                }
            }
        }
    }
}

```

```

    }
    return (temp);
}
else
{
    syntaxerror = true;
    return (Fail);
}
}

float plusminus ()
// Menghasilkan hasil penjumlahan dan pengurangan
{
    float x = timesdiv();
    if (FailedInput())
    {
        return (Fail);
    }
    else
    {
        while (*now == '+' || *now == '-')
        {
            char opr = *now;
            now++;
            float y = timesdiv();
            if (FailedInput())
            {
                return (Fail);
            }
            if (opr == '+')
            {
                x += y;
            }
            else
            {
                x -= y;
            }
        }
        return x;
    }
}

float timesdiv ()
// Menghasilkan hasil perkalian dan pembagian
{
    float x = power();

```



```

    if (FailedInput())
    {
        return (Fail);
    }
    while (*now == '*' || *now == '/')
    {
        char opr = *now;
        now++;
        float y = power();
        if (FailedInput())
        {
            return (Fail);
        }
        if (opr == '*')
        {
            x *= y;
        }
        else
        {
            if (y == 0)
            {
                matherror = true;
                return (Fail);
            }
            else
            {
                x /= y;
            }
        }
    }
    return x;
}

float brackets ()
// Menghasilkan hasil dari tanda kurung jika berhasil
{
    float res = plusminus();
    if (*now != ')')
    {
        syntaxerror = true;
        return (Fail);
    }
    else
    {
        now++;
        return res;
    }
}

```

```

    }
}

float power ()
// Mengembalikan hasil perpangkatan dari sebuah bilangan
{
    float x = checksign();
    if (*now == '^' && !FailedInput())
    {
        now++;
        x = executePower(x);
    }
    if (FailedInput())
    {
        return (Fail);
    }
    else
    {
        return (x);
    }
}

float executePower (float x)
// Membantu perpangkatan dengan pemanggilan rekursif
{
    float y = checksign();
    if (FailedInput())          // Basis
    {
        return (y);
    }
    else if (*now != '^')      // Basis
    {
        float a=pow(x,y);
        if (isnan(a)){
            matherror = true;
            return (Fail);
        }
        else{
            return (a);
        }
    }
    else                        // Rekurens
    {
        now++;
        float z = executePower(y);
        if (!FailedInput())

```

```

    {
        float b=pow(x,z);
        if (isnan(b)){
            matherror = true;
            return (Fail);
        }
        else{
            return (b);
        }
    }
    else
    {
        return (Fail);
    }
}

void utama ()
{
    int choice;
    char input[50];

    printf("Pilihan Menu\n1. Calculator\n2. Quit\n");
    scanf("%d", &choice);

    while(choice != 2)
    {
        if (choice == 1)
        {
            printf("Masukkan operasi yang ingin dilakukan!\n");
            scanf ("%s", input);
            now = input;
            start = input;
            syntaxerror = false;
            matherror = false;
            float count = plusminus();

            if (syntaxerror || *now != '\0')
            {
                printf("Syntax error\n");
            }
            else if (matherror)
            {
                printf("Math error\n");
            }
            else

```

```

        {
            printf("Result = %.2f\n", count);
        }
    }
else
{
    printf("Input tidak dikenal\n");
}
printf("Pilihan Menu\n1. Calculator\n2. Quit\n");
scanf("%d", &choice);
}
}

```

4.2 Source Code untuk Main Program

```

#include
"kalkulator.c"

#include <stdio.h>
#include <stdlib.h>

//KAMUS
char CC;

int main(){
    printf(" _ _ _ _ _ \n");
    _ printf("/ | / | _ / | / | _ / | \n");
    / | _ _ _ _ _ \n");
    printf("$ $ | /$ $ / _ _ _ $ $ |$ $ | _ _ _ $ $ | _ _ _ \n");
    _$ $ | _ _ _ _ _ \n");
    printf("$ $ | /$ $ / _ _ \ \ $ $ |$ $ | / | / | / |$ $ | / \ \ / $ $ | / _ \ \ / \ \ \n");
    printf("$ $ $ $ < $ $ $ $ $ |$ $ |$ $ | _ /$ $ / $ $ | $ $ |$ $ | $ $ $ $ $ |$ $ $ $ $ / _ /$ $ $ $ $ | /$ $ $ $ $ | \n");
    printf("$ $ $ $ $ \ \ / $ $ |$ $ |$ $ $ $ < $ $ | $ $ |$ $ | / $ $ | $ $ | _ $ $ | $ $ |$ $ | $ $ / \n");
    printf("$ $ |$ $ \ \ /$ $ $ $ $ |$ $ |$ $ $ $ $ \ \ $ $ \ \ _ $ $ |$ $ | /$ $ $ $ $ | $ $ | / |$ $ \ \ _ $ $ |$ $ | _ \n");
    printf("$ $ | $ $ |$ $ $ $ |$ $ |$ $ |$ $ |$ $ |$ $ $ $ / $ $ |$ $ $ $ | $ $ $ $ / $ $ $ $ / $ $ | _ \n");
    printf("$ $ / $ $ / $ $ $ $ $ / $ $ / $ $ / $ $ / $ $ $ $ $ / $ $ / $ $ $ $ $ / $ $ $ $ / $ $ / \n");

    printf("-----Gunakanlah seperti kalkulator biasa-----\n");
    printf(">> Contoh masukkan :\n");
    printf("Penjumlahan >> 2+3 \n");

```

```

printf("Pengurangan >> 2-3 \n");
printf("Perkalian >> 2*3 \n");
printf("Pembagian >> 2/3 \n");
printf("Perpangkatan >> 2^3 \n");
printf("Perkurungan >> 2*(3+2) \n");

utama();
printf("\nMau hitung lagi? (Y/N)\n");
scanf("%s", &CC);
while ((CC != 'N') && (CC != 'n')){
    utama();
    printf("\nMau hitung lagi? (Y/N)\n");
    scanf("%s", &CC);
}

printf("\n- C O N T R I B U T O R S -\n");
printf("1. T. Andra Oksidian Tafly / 13517020\n");
printf("2. Andrian Cedric / 13517065\n");
printf("3. Jan Meyer Saragih / 13517131\n");
printf(" KALKULATOR. c. 2018\n");

return 0;
}

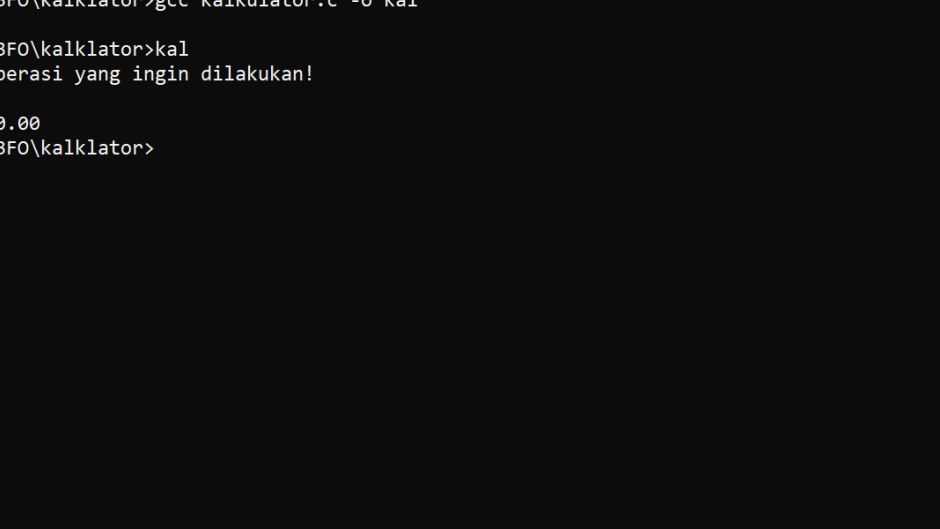
```

BAB 5

CONTOH MASUKAN DAN KELUARAN PROGRAM

[illegible]

Gambar 5.1. Tampilan utama kalkulator



The screenshot shows a Windows Command Prompt window with the following text:

```
C:\WINDOWS\system32\cmd.exe
D:\TUBES\TBFO\kalklator>gcc kalkulator.c -o kal
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
20+30
Result = 50.00
D:\TUBES\TBFO\kalklator>
```

The taskbar at the bottom of the screen displays various application icons, including Windows Explorer, Google Chrome, Microsoft Word, Adobe Reader, and several development tools like Visual Studio Code, PyCharm, and Photoshop. The system clock in the bottom right corner indicates the time is 13:30 on 16/11/2018.

Gambar 5.2 Contoh penjumlahan

```
CA\WINDOWS\system32\cmd.exe

D:\TUBES\TBFO\kalklator>gcc kalkulator.c -o kal

D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
20+30
Result = 50.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(20*3)/2
Result = 30.00
D:\TUBES\TBFO\kalklator>
```

Gambar 5.3 Contoh operasi perkalian dan pembagian dengan prioritas kurung

```
CA\WINDOWS\system32\cmd.exe

D:\TUBES\TBFO\kalklator>gcc kalkulator.c -o kal

D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
20+30
Result = 50.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(20*3)/2
Result = 30.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(-5)^(2/3)
Math error

D:\TUBES\TBFO\kalklator>
```

Gambar 5.4 Contoh pesan kesalahan program MATH_ERROR

```
CA\WINDOWS\system32\cmd.exe

D:\TUBES\TBFO\kalklator>gcc kalkulator.c -o kal

D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
20+30
Result = 50.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(20*3)/2
Result = 30.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(-5)^(2/3)
Math error

D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(2-3)/2
Result = -0.50
D:\TUBES\TBFO\kalklator>
```

Gambar 5.5 Contoh penggunaan kurung dengan tepat

```
CA\WINDOWS\system32\cmd.exe

D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
20+30
Result = 50.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(20*3)/2
Result = 30.00
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(-5)^(2/3)
Math error

D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
(2-3)/2
Result = -0.50
D:\TUBES\TBFO\kalklator>kal
Masukkan operasi yang ingin dilakukan!
2+3)
Syntax error

D:\TUBES\TBFO\kalklator>
```

Gambar 5.6 Contoh kesalahan sintaks pada kalkulator


```
C:\WINDOWS\system32\cmd.exe
Pengurangan >> 2-3
Perkalian >> 2*3
Pembagian >> 2/3
Perpangkatan >> 2^3
Perkurungan >> 2*(3+2)
Masukkan operasi yang ingin dilakukan!
2*(3+2)
Result = 10.00
Mau hitung lagi? (Y/N)
Y
Masukkan operasi yang ingin dilakukan!
2^(3*3)
Result = 512.00
Mau hitung lagi? (Y/N)
Y
Masukkan operasi yang ingin dilakukan!
2+(3
Failed input
Mau hitung lagi? (Y/N)
N

- C O N T R I B U T O R S -
1. T. Andra Oksidian Tafly / 13517020
2. Andrian Cedric / 13517065
3. Jan Meyer Saragih / 13517131
KALKULATOR. c. 2018

D:\TUBES\TBFO\kalkulator>
```

Gambar 5.7 Tampilan penutup dari kalkulator

DAFTAR PUSTAKA

Wikipedia. (2018). *Mesin Hitung* . Diakses November 2018, dari https://id.wikipedia.org/wiki/Mesin_hitung
<https://m.media-amazon.com/images/S/aplus-media/mg/a48e575c-d047-4ba5-89b0-1c8977511564.png>