

LAPORAN TUGAS BESAR AKHIR SEMESTER

16TIN2074 - STRUKTUR DATA DAN ALGORITMA

APLIKASI KALKULATOR



Disusun oleh :

Dzakira Fabillah - 191524040

Rizka Auliarahmi - 191524057

**Program Studi D4 Teknik Informatika
Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Bandung
2020**

DAFTAR ISI

DAFTAR ISI	1
BAB I DESKRIPSI APLIKASI	2
1.1 Landasan teori	2
1.2 Kemampuan Program	4
1.3 Batasan Aplikasi	5
BAB II DESAIN APLIKASI	7
2.1 Bahasa Pemrograman	7
2.2 Struktur Data	7
2.3 Desain Proses	8
2.4 Desain Interface	10
2.5 Input Program	13
2.6 Proses Program	13
2.7 Output Program	14
BAB III PENUTUP	17
5.1 Alur Kerja dan Pembagian Tugas	17
5.2 Lesson Learned	17
DAFTAR PUSTAKA	19

BAB I DESKRIPSI APLIKASI

1.1 Landasan teori

1.1.1 Kalkulator

Menurut KBBI, mesin hitung atau kalkulator adalah alat untuk menghitung dari perhitungan sederhana seperti penjumlahan, pengurangan, perkalian dan pembagian sampai kepada kalkulator sains yang dapat menghitung rumus matematika tertentu.

Pada tahun 1623, kalkulator yang berupa mesin penambah pertama di dunia dibuat oleh Shickard. Kalkulator ini hanya dapat melakukan operasi penambahan. Seorang ahli astronom Philip Matthaues Hann, pada tahun 1773, merancang sebuah kalkulator yang dapat membantu menghitung parameter jam dan planetarium yang dibuat dengan 12 drum dalam susunan melingkar yang diaktifkan menggunakan putaran yang terletak di sumbu drum .

Kalkulator ilmiah pertama kali diperkenalkan pada tanggal 1 Februari 1972 dan dijual di Amerika Serikat oleh Hawller-Packard. Kalkulator ini bernama HP-35 yang sudah dapat menyelesaikan operasi algoritma dan trigonometri dan dapat memberikan angka ilmiah hingga sepuluh digit dibelakang koma dan dua digit eksponen.

Seiring dengan perkembangan zaman, kalkulator menjadi fitur atau fungsi tambahan pada handphone, komputer, dan laptop. Sehingga kalkulator menjadi alat hitung yang praktis karena sudah menjadi bagian dari gadget yang tidak bisa lepas dari kehidupan manusia.

Keuntungan dari kalkulator adalah membantu pekerjaan manusia. Kalkulator, terutama kalkulator scientific membantu perhitungan yang sulit misalnya jika melibatkan operator 'log', 'ln', dll. Selain itu, kalkulator juga akan menghasilkan hasil yang akurat dan cepat. Tidak seperti manusia, kalkulator tidak akan keliru dalam melakukan perhitungan karena sudah diberikan algoritma yang sesuai.

1.1.2 Ekspresi Aritmatika

Ekspresi aritmatika terdiri dari operand dan operator. Operator dalam ekspresi aritmatika dapat dibagi menjadi 2 jenis, yaitu :

- Binary operator (operator pasangan) yaitu operator yang memiliki 2 buah operand (diapit oleh 2 buah operand). Operator – operator yang termasuk dalam binary operator adalah operator penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/), modulo (mod), divisor (div), dan pemangkatan (^).

- Unary operator (operator tunggal) yaitu adalah operator yang hanya memiliki 1 buah operand (diikuti oleh sebuah operand). operator yang termasuk dalam unary operator adalah operator minus (\sim), operator faktorial (!), operator trigonometri (seperti operator sinus, cosinus, tangen), operator exponential (exp) dan fungsi logaritma (log).

Prioritas / kedudukan dari masing – masing operator (baik unary operator maupun binary operator) dari tinggi ke rendah adalah sebagai berikut:

1. Operator pemangkatan (\wedge) dan semua unary operator.
2. Operator perkalian ($*$), pembagian ($/$), modulo (mod) dan divisor (div).
3. Operator penjumlahan (+) dan pengurangan (-).
4. Operator perbandingan, yaitu operator lebih besar, lebih kecil, sama dengan, lebih besar sama dengan, lebih kecil sama dengan, dan tidak sama dengan.
5. Operator logika NOT.
6. Operator logika AND dan OR.
7. Assignment Operator (=).

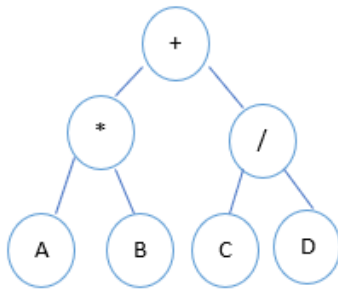
Ekspresi aritmatika akan diselesaikan berdasarkan urutan prioritas dari operator di atas dengan ketentuan operator yang memiliki prioritas yang lebih tinggi akan diselesaikan terlebih dahulu. Tahapan – tahapan penyelesaian suatu ekspresi aritmatika dapat direpresentasikan dalam bentuk graph yang dinamakan pohon ekspresi (expression tree).

1.1.3 Notasi Ekspresi Aritmatika

Notasi penulisan ekspresi matematika diantaranya :

1. Infix. Infix merupakan bentuk penulisan normal dari ekspresi aritmatika. Suatu infix dapat berupa operand tunggal, atau gabungan dari unary operator dengan infix, ataupun berupa gabungan dari binary operator dengan dua buah infix. Dalam binary tree, notasi infiks adalah hasil dari traversal inorder.
2. Prefix. Prefix merupakan bentuk penulisan ekspresi aritmatika dimana operator ditulis di depan dari operandnya. Dalam binary tree, notasi prefiks adalah hasil dari traversal PreOrder.
3. Suffix / Postfix. Bentuk suffix/postfix merupakan bentuk penulisan ekspresi aritmatika dimana operator ditulis di belakang dari operandnya. Dalam binary tree, notasi postfix adalah hasil dari traversal PostOrder.

Sebagai perbandingan, berikut contoh penulisan masing masing notasi :



Notasi Infix : $A * B + C / D$

Notasi Prefix : $A B * C D / +$

Notasi Postfix : $+ * A B / C D$

1.1.4 Binary Expression Tree

Binary Expression tree adalah binary tree yang digunakan untuk merepresentasikan ekspresi aritmatika. Setiap internal node berisi operator, dan leaf/daun berisi operand. Operator dapat berbentuk Binary Operator maupun Unary Operator.

1.2 Kemampuan Program

Program kami dapat melakukan operasi matematika terhadap operator yang valid. Operator yang valid di antaranya adalah $(,)$, $^$, $\sqrt{\text{sqrt}}$, \times , \div , $+$, $-$, \sin , \cos , \tan , \log , \ln , \exp dan $!$ (faktorial). Operand yang dapat digunakan berupa bilangan bulat, bilangan desimal, nol, bernilai positif atau bernilai negatif. Hasil dari ekspresi matematika dalam program kami dapat di konversi ke Binary, Hexadecimal, dan juga Octal. Namun hasil konversi tidak dapat dijadikan ekspresi matematika melainkan hanya ditampilkan sebagai output. Sehingga tidak dapat melakukan operasi matematika selain operand desimal.

Pada program kalkulator yang kami buat, ekspresi matematika dapat berasal dari file dan input berdasarkan button button yang terdapat di Jendela yang dibuat menggunakan JFrame. Keduanya akan menjadi String dengan notasi Infix yang kemudian akan dieksekusi oleh program.

Pada program kami terdapat juga fitur untuk melakukan operasi terhadap dua matriks. Operasi yang berlaku adalah tambah, kurang, kali, dan bagi. Pada penghitungan matriks, input dimasukkan di textArea pada jendela matriks. Angka yang dimasukkan di textArea akan dieksekusi sesuai operator oleh program.

1.3 Batasan Aplikasi

1.3.1 String Input

Kalkulator kami dapat menerima input dari file yang ekstensinya .txt dan input berdasarkan button button yang terdapat di Jendela yang dibuat menggunakan JFrame. Keduanya akan menjadi String dengan notasi Infix yang kemudian akan dieksekusi oleh program.

a. Jendela menggunakan JFrame.

Jendela ini berisi button button operand dan operator, Text Area untuk menampilkan output, dan Text Field untuk input. Keunggulan dari input dari Jendela adalah meminimalisir input yang tidak valid.

b. File

File dapat berisi lebih dari satu baris ekspresi matematika, kemudian setiap barisnya akan diproses kemudian dibuat file bernama result.txt yang berisi ekspresi matematika beserta hasilnya. Tujuannya adalah jika ada beberapa ekspresi yang sudah ada bentuk file dapat langsung diproses tanpa harus menginput nilai ekspresi satu per satu.

1.3.2 Operator dan Operand

Operand yang valid yaitu bilangan desimal, bilangan negatif, dan bilangan positif, bilangan bulat, dan nol. Operator yang valid di antaranya adalah $(,), ^, \sqrt{\text{ (sqrt) }}, \times, \div, +, -, \sin, \cos, \tan, \log, \ln, \exp$ dan $!$ (faktorial). Operator yang diproses terlebih dahulu mengikuti aturan sebagai berikut :

a. derajat $'(^{\circ})'$, tanda kurung

b. derajat $'(^{\circ})'$ (pangkat) = derajat $'(\sqrt{})'$ (akar) = $'\sin' = 'cos' = 'tan'$, pangkat, akar bilangan dan trigonometri.

c. derajat $'\times' = \text{derajat } '\div'$, perkalian dan pembagian

d. derajat $'+' = \text{derajat } '-'$, pertambahan dan pengurangan

e. derajat $'(' > \text{derajat } '^{\circ} > \text{derajat } '\times' > \text{derajat } '+'$

1.3.3 Batasan Lainnya

- Ekstensi file yang digunakan untuk input harus berupa .txt, hal ini dilakukan dengan tujuan untuk membatasi input dari file yang tidak sesuai.

- Hasil konversi berupa biner, hexadesimal, ataupun octal tidak dapat dijadikan operand. Hanya menjadi output yang ditampilkan. Output di input Field tetaplah bilangan desimal. Konversi juga tidak dapat dilakukan jika input berasal dari file.
- Konversi hanya bisa dilakukan apabila hasilnya positif karena java tidak bisa mengkonversi bilangan negatif ke binary, hexa, dan octa. Sehingga akan ada pemberitahuan bahwa konversi tidak dapat dilakukan namun program masih dapat berjalan lancar.

BAB II DESAIN APLIKASI

2.1 Bahasa Pemrograman

Bahasa Pemrograman yang kami gunakan adalah bahasa pemrograman java.

2.2 Struktur Data

2.2.1 Class Node

Atribut dari Object Node terdiri dari :

```
private Object value;  
private Node parent;  
private Node left;  
private Node right;
```

2.2.2 Class Binary Tree

Binary Tree terdiri dari elemen elemen berupa Node.

Attribute dari BinaryTree

```
private int size;  
private Node root;
```

2.2.3 Class Process

Atribut dari process

```
public static final char SQRT = 'V';  
public static final char SQUARE = 'S';  
public static final char SIN = 'N';  
public static final char COS = 'C';  
public static final char TAN = 'T';  
public static final char POWER = 'P';  
public static final char PLUSMIN = 'L';  
  
private static final char[] nonNumeric = {'+', '-', '/',  
'*', '%', '!', '(', ')', SQRT, SQUARE, SIN, COS, TAN, POWER,  
PLUSMIN, 'm' };  
  
private BinaryTree tree;  
private String result;
```


2.3 Desain Proses

2.3.1 Operasi-operasi Struktur Data

2.3.1.1 Operasi-operasi Node

Operasi-operasi yang terjadi pada node adalah

Method	Proses
Node getParent()	Mengembalikan Node parent.
Node getLeftChild()	Mengembalikan Node left child.
Node getRightChild()	Mengembalikan Node right child.
Node getValue ()	Mengembalikan isi dari node.
setRight(Node v)	Mengatur node right child
setLeft(Node v)	Mengatur node left child
setParentt(Node v)	Mengatur node parent

2.3.1.2 Operasi-operasi Binnary Tree

Method	Proses
Node root()	Mengembalikan root
Int size()	Mengembalikan ukuran tree
isNotLeaf(Node elm)	Mengembalikan nilai true jika node bukan daun
Boolean isLeaf()	Mengembalikan nilai true jika node adalah daun
Boolean hasLeft()	Mengembalikan nilai true jika node memiliki anak kiri
Booelan hasRight()	Mengembalikan nilai true jika node memiliki anak kanan

Boolean isRoot()	Mengembalikan nilai true jika node adalah root
updateTree(@NotNull Node newRoot, @Nullable Node rightChild)	Memperbarui tree yang lama dengan yang baru, root sebelumnya menjadi anak kiri dari root baru, anak kanan boleh diinsert boleh juga tidak

2.3.1.3 Operasi-operasi Process

Method	Proses
String getResult();	Mengembalikan string yang merupakan hasil dari operasi matematika.
double extractNumber(String str);	Mereplace <i>opening bracket</i> dan <i>closing bracket</i> '(' ')' dengan spasi kemudian melakukan trim() pada String untuk menghapus spasi di awal dan akhir dan kemudian string di cast dan di return menjadi tipe double.
makeTree(String expression);	Membentuk Binary Expression Tree dari String Input berupa notasi infix sesuai dengan hakikat derajat operator yang sudah ditentukan. Operator akan selalu menjadi parent yang akan mengkalkulasikan hasil dari rightChild dan LeftChild nya.
getOperator()	Mereturn posisi dari operator dalam sebuah ekspresi matematika.
Booelan isOther	Function ini akan mereturn true apabila operator adalah selain dari tambah '+', kurang '-', kali '*', bagi '/', dan pangkat '^'.
Node getNewChild(String expression)	Mereturn node yang akan menjadi anak dari node lain. Function ini sebenarnya mirip dengan function makeTree().
double evaluateTree(Node v)	Mengkalkulasi kan Operasi Matematika berdasarkan Expression Tree yang sudah dibentuk.

replace(String str)	Me replace beberapa operator seperti sin, cos, tan, dll dengan final char agar kode lebih mudah di pahami.
String getValidSubstring(String str)	Menghapus opening dan closing bracket yang berada di awal dan di akhir ekspresi.
int faktorial(int num)	Function yang akan mereturn nilai faktorial dari argumen yang diberikan.

2.3.1.4 Operasi-operasi File

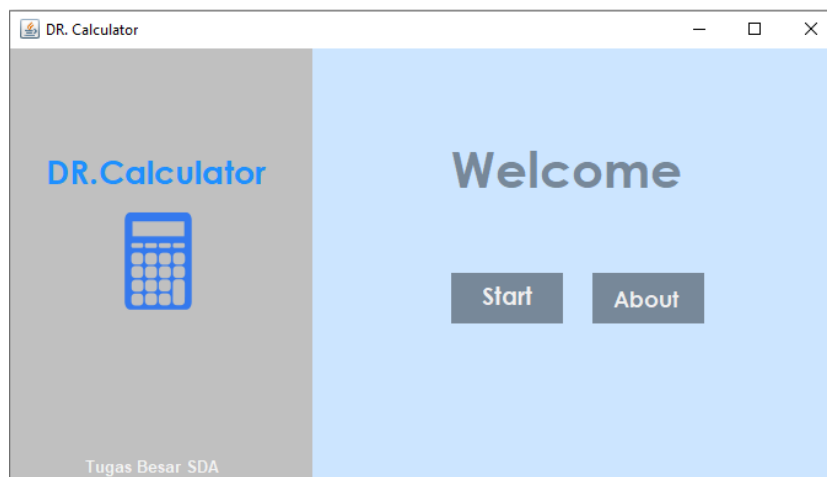
Dalam class ini hanya terdapat satu method yaitu run();

Method	Process
run()	Dalam method ini terjadi proses meBaca file, memproses String ekspresi matematika, dan menuliskan ekspresi aritmatikanya dan hasilnya ke file output.

2.4 Desain Interface

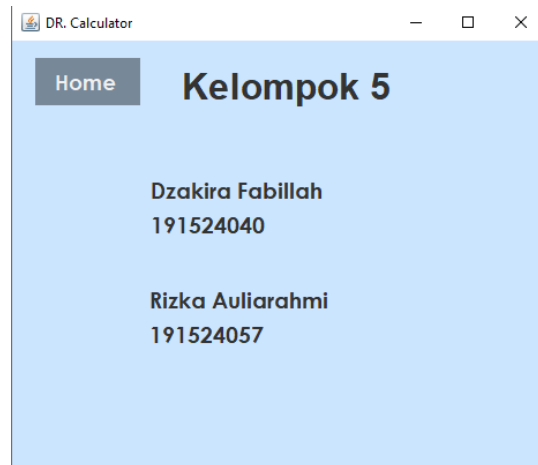
2.4.1 Welcome

Jendela pertama yang akan dimunculkan ketika run program.



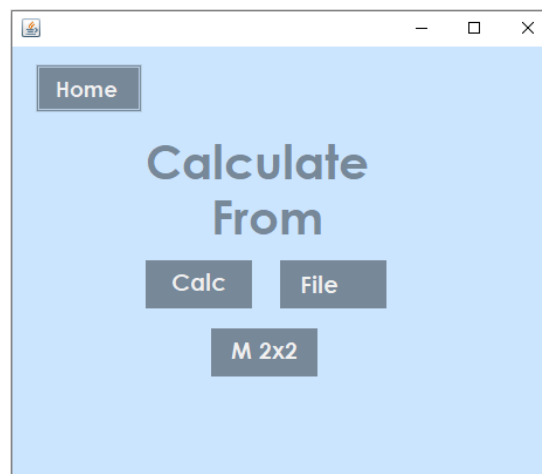
2.4.2 About

Berisi Data nama dan NIM dari pembuat program.



2.4.3 Start

Berisi 3 Button yaitu 'Calc' untuk membuka jendela kalkulator, 'File' untuk melakukan kalkulasi pas ekspresi matematika yang tersimpan dalam file, dan M 2x2 yang merupakan fitur tambahan kalkulator kami untuk melakukan operasi tambah, kurang dan kali 2 buah matriks berordo 2 X 2.



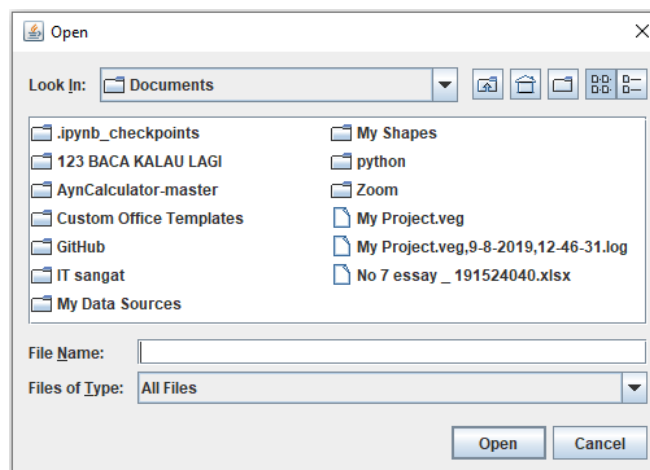
2.4.4 View (tampilan Calculator)

Berisi button button operand dan operator, Text Area untuk menampilkan output, dan Text Field untuk input.



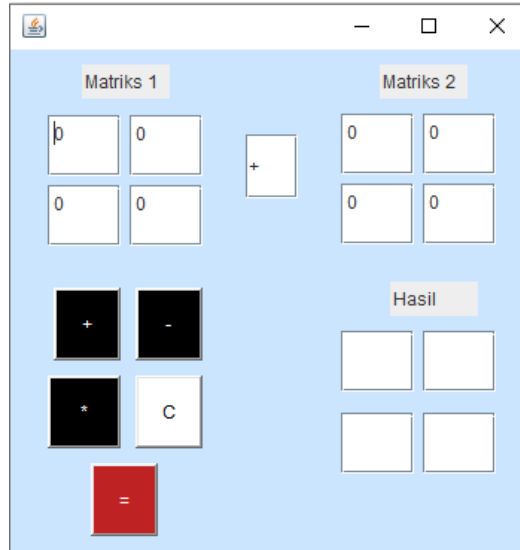
2.4.5 File

Tampilan ketika open file menggunakan JFileChooser.



2.4.6 viewMatrisk1

Tampilan untuk melakukan operasi pada matriks 2 x 2



2.5 Input Program

Input yang dapat diterima oleh aplikasi kami adalah file yang ekstensinya .txt, input berdasarkan button button yang terdapat di Jendela yang dibuat menggunakan JFrame, dan input yang ditambahkan di text area pada jendela matriks. Keduanya akan menjadi String dengan notasi Infix yang kemudian akan dieksekusi oleh program.

2.6 Proses Program

2.6.1 Kalkulator

Proses program kalkulator dalam mengeksekusi ekspresi matematika yaitu :

1. Program menerima input string ekspresi matematika dengan notasi infix. Input berasal dari tombol pada jendela kalkulator.
2. Pada method `equalsClicked()`, Objek dari class `Process.java` di instansiasikan dengan mempassing String ekspresi matematika dengan notasi infix .
3. Tipe data pada operand ekspresi matematika akan dikonversi menjadi double oleh method `extractNumber()` dan akan di generate menjadi expression tree.
4. Sebelum menjadi expression tree, ekspresi matematika akan di proses di `getOperator(String str)` untuk mengecek posisi operator, mengecek ada tidaknya operator, dan memproses operator sesuai dengan derajat operator.

5. Expression tree akan dibuat di `makeTree(String expression)`. Jika operator hanya membutuhkan satu operand (operator unary) maka hanya left child saja yang terisi sedangkan right child null. Sedangkan binary operator akan memiliki dua child. Operator tidak akan menjadi daun. Daun pasti berisi operand.
6. Expression tree akan diproses sesuai operator pada `evaluateTree(Node v)`. Jika operator membutuhkan satu operand(unary) maka `evaluateTree(Node v)` akan memproses operator dengan left child. Jika operator membutuhkan dua operand (binary operator), maka `evaluateTree(Node v)` akan memproses operator dengan left child dan right child.
7. Hasil dari expression tree akan ditampilkan pada text Area di jendela kalkulator.

2.6.2 File

Proses penghitungan yang inputnya dari file sebenarnya prosesnya sama saja dalam proses penghitungannya seperti menggunakan kalkulator, perbedaanya terletak di input saja.

1. Membuka file menggunakan `JFileChooser`.
2. Membaca File menggunakan `BufferedReader`.
3. Membaca baris per baris ekspresi matematika.
4. Menginstansiasikan objek dari class `Process.java` untuk memproses ekspresi matematika.
5. Menuliskan pada file `result.txt` ekspresi matematika tersebut ditambahkan dengan hasilnya.
6. Proses yang sama berulang pada baris baris selanjutnya.

2.6.3 Matriks

1. Program membaca input angka yang dimasukan di setiap baris dan kolom matriks satu dan dua.
2. Program membaca operator yang berada di `TextPane operator`. Operator default adalah '+'.
3. Program memproses input berdasarkan operator yang dipilih.
4. Output ditampilkan pada kolom dan baris matriks hasil.

2.7 Output Program

2.7.1 Output dari Kalkulator

Output berupa string yang dimunculkan di TextField dan juga history dari operasi yang dilakukan disimpan di Text Area. Contohnya sebagai berikut :

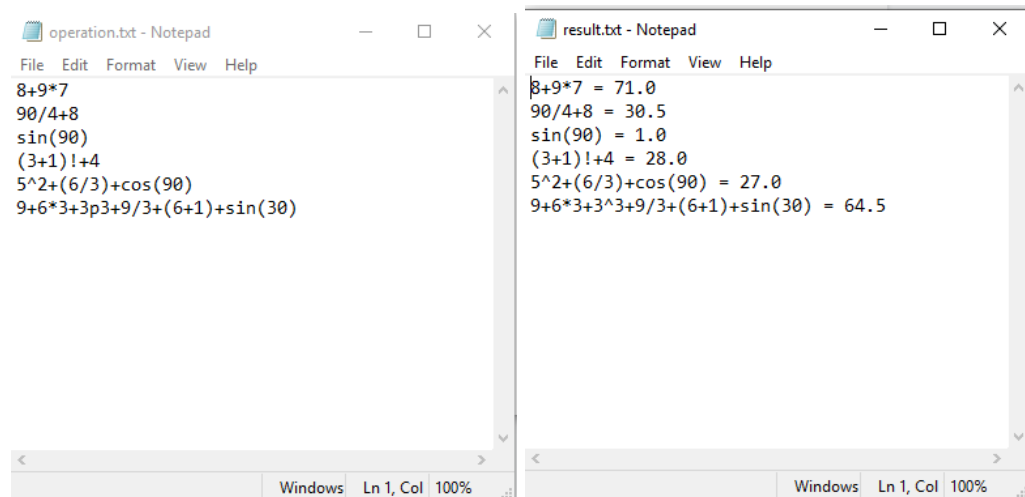


2.7.2 Output dari File

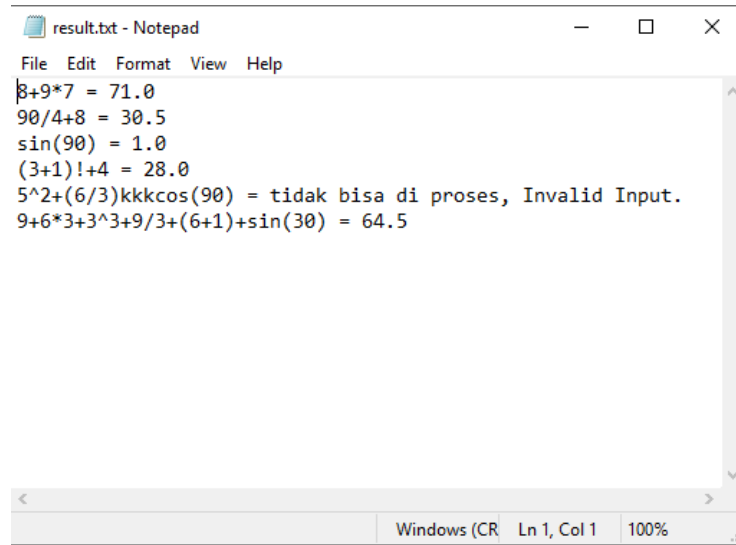
Output berupa file yaitu result.txt yang berisikan ekspresi aritmatika dan juga hasil nya. Contoh

File input :

File output :



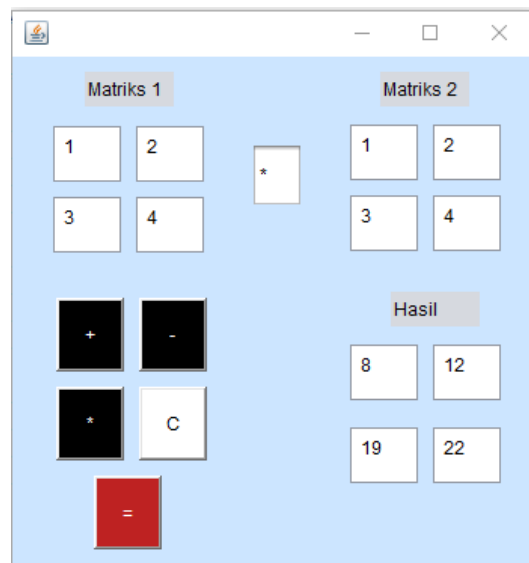
Jika terdapat operator atau operand yang tidak valid maka output nya sebagai berikut :



```
File Edit Format View Help
8+9*7 = 71.0
90/4+8 = 30.5
sin(90) = 1.0
(3+1)!+4 = 28.0
5^2+(6/3)kkkcos(90) = tidak bisa di proses, Invalid Input.
9+6*3+3^3+9/3+(6+1)+sin(30) = 64.5
Windows (CR Ln 1, Col 1 100%
```

2.7.3 Output dari Operasi Matriks

Output berada pada TextPane hasil. Contohnya sebagai berikut :



BAB III PENUTUP

5.1 Alur Kerja dan Pembagian Tugas

Pada permulaan pengerjaan, kami mendiskusikan mengenai bagaimana teknis algoritma perhitungan operasi matematika dengan Expression Tree, bagaimana cara generate expression tree dari input dan operator apa saja yang akan kami tambahkan dari standar minimal tugas besar. Kami bersama sama mencari source code kalkulator dari github sebagai referensi.

Seiring berjalannya waktu kami kembali mendiskusikan fitur tambahan yang ingin ditambahkan dan terus memperbaiki program bersama sama. Dalam program ini kami menggunakan referensi dari berbagai sumber diantaranya :

- Luka Kraij yang dapat di akses di <https://github.com/lukakralj/MyCalculator>
- Manash Kumar Mondal yang dapat diakses di <https://github.com/Manash-git/Scientific-Calculator-in-Java>
- Juhyun Lee untuk bagian upload file sebagai input yang dapat diakses di <https://github.com/alex87lee/Calculator-using-Binary-tree/blob/master/src/Project3.java>.

Pertama tama Dzakira membuat UI kalkulator kemudian Rizka menyambungkannya dengan class untuk melakukan proses operasi matematika yang didapat dari referensi dan mengaktifkan fitur sin, cos ,tan dan pangkat. Rizka membuat tampilan welcome dan about kemudian Dzakira menambahkan jendela start juga dan fitur input dari file.

Seiring dengan pengerjaan, kami terus eksplorasi mengenai kalkulator dan mendiskusikan tambahan operator dan juga fitur konversi yang kemudian diimplementasikan. Kami juga menambahkan fitur tambahan yaitu matriks bersama sama. Pengerjaan laporan juga dikerjakan bersama sama.

Media komunikasi yang kami gunakan adalah whatsapp dan Google Meet. Kami juga menggunakan github sebagai jalur sharing data. Pada awal awal pengerjaan, kami tidak langsung mengupload file ke github melainkan manual menggunakan whatsapp. Kami menggunakan google meet untuk berdiskusi dalam perancangan atau penyelesaian bug dan kami juga terkadang mengerjakan penambahan fitur bersama sama melalui google meets seperti penambahan fitur matriks.

5.2 Lesson Learned

Lesson Learned yang kami dapatkan adalah :

- Expression Tree

Kami mempelajari lebih dalam tentang expression tree, bagaimana cara meng-generate expression tree dan kemudian mengeksekusinya. Kami mempelajarinya melalui referensi source code dan juga materi mengenai expression tree.

- Pengetahuan baru terkait membuat Java GUI menggunakan java swing. Terdapat dua cara untuk membuat jendela di java. Dan juga didalamnya terdapat banyak fungsi/fitur fitur untuk membuat UI lebih menarik dan interaktif.
- Pengetahuan baru terkait function-function dari library java.Math seperti log(), exp(), ln(), dan masih banyak lagi.
- Pengetahuan baru mengenai dokumentasi seperti @nullable.
- Pengetahuan baru mengenai JFileChooser untuk membuka dokumen.
- Pengetahuan baru mengenai BufferedReader dan BufferedWriter untuk membaca dan menulis file.
- Manajemen Waktu
Membagi waktu antara tugas dan hal lainnya sangat lah penting untuk memaksimalkan hasil pekerjaan.
- Kerja Sama Tim
Kami mempelajari pentingnya komunikasi dalam anggota memiliki satu pemahaman dalam pembuatan program.

DAFTAR PUSTAKA

Sumardi, Haryoko. 2011. RANCANG BANGUN PERANGKAT LUNAK SIMULASI PEMBELAJARAN POHON EKSPRESI (EXPRESSION TREE) DARI EKSPRESI ARITMATIKA PREFIX, INFIX DAN POSTFIX. Jurnal Computech & Bisnis. Vol.5 No.1

Sains.kompas.com. (2018, 01 Februari). Penemuan yang Mengubah Dunia : Kalkulator. Diakses pada 19 Juli 2020, dari <https://sains.kompas.com/read/2018/02/01/203300023/penemuan-yang-mengubah-dunia-kalkulator?page=a>