Laporan Program Kalkulator

*Dokumen Spesifikasi Program untuk Tugas Akhir Semester MK Struktur Data adn Algoritma (Praktik)*



Disusun Oleh :

|  |  |
| --- | --- |
| Alifah Fisalsabilawati | 201511035 |
| Faishal Muhammad | 201511039 |

Kelas 1 – B

**Jurusan Teknik Komputer dan Informatika Program Studi D3 Teknik Informatika**

**Politeknik Negeri Bandung 2021**

**TABEL REVISI**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Tanggal** | **Keterangan** | **PIC** |
| 1 | 15 Juni 2021 | 1. Menambahkan deskripsi dan fungsi kalkulator pada sub bab deskripsi program 2. Memahami cara mengubah infix ke postfix menggunakan stack 3. Memahami cara pengisian operasi pada stack |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**DAFTAR ISI**

**BAB I**

**SPESIFIKASI PROGRAM**

* 1. **Deskripsi Program**

1. **Definisi**

Kalkulator adalah alat untuk menghitung dari perhitungan sederhana seperti [penjumlahan](https://id.wikipedia.org/wiki/Penjumlahan), [pengurangan](https://id.wikipedia.org/wiki/Pengurangan), [perkalian](https://id.wikipedia.org/wiki/Perkalian) dan [pembagian](https://id.wikipedia.org/wiki/Pembagian) sampai kepada kalkulator  [sains](https://id.wikipedia.org/wiki/Sains)

yang dapat menghitung [rumus](https://id.wikipedia.org/wiki/Rumus) [matematika](https://id.wikipedia.org/wiki/Matematika) tertentu. Pada perkembangannya sekarang ini, kalkulator sering dimasukkan sebagai [fungsi](https://id.wikipedia.org/wiki/Fungsi) tambahan daripada [komputer](https://id.wikipedia.org/wiki/Komputer), [handphone](https://id.wikipedia.org/wiki/Handphone), bahkan sampai [jam tangan](https://id.wikipedia.org/wiki/Jam_tangan). Rata-rata kalkulator memiliki 12 hingga 15 digit, tetapi mesin hitung ilmiah sanggup menghitung hingga angka 1 [googol](https://id.wikipedia.org/wiki/Googol) menggunakan rumus tertentu.

Dalam program yang akan dibuat merupakan kalkulator basic yang biasa digunakan pada umumnya. Didalamnya kami menggunakan operasi infix, prefix dan postfix dengan struktur list stack dan struktur data tree.

1. **Konsep Program**

Pertama yang dilakukan yaitu mengubah operasi infix menjadi postfix. Dalam merubahnya menggunakan 2 cara, yang pertama menggunakan cara manual dan menggunakan stack. Berikut contoh notasi infix

* Cara stack

*Stack* bisa dibaratkan sebagai tumpukan . Stack memiliki cara kerja, “yang pertama masuk ke kotak, maka akan terakhir kali diambil kembali” atau “*first in* *last out*”*,* atau sebaliknya, “yang terakhir masuk ke kotak, akan diambil yang pertama kali,”

atau “*last in first out*.”

1. Proses yang akan dilakukan dari kiri ke kanan
2. Kondisi stack masih NULL
3. Jika yang diproses pertama adalah operand, maka tulis di hasil. Di contoh operand adalah “5”

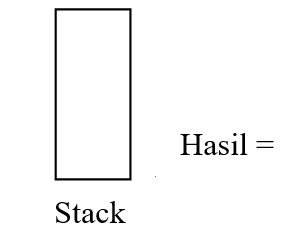
5 \* (2 + 3 ) / 1 – 6

NULL

Hasil = 5

Stack

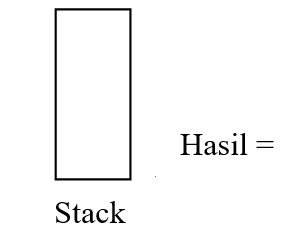
1. Lanjutkan ke operator “\*”, karena kondisi stack masih NULL, maka masukkan operator tersebut ke dalam stack

 5 \* ( 2 + 3 ) / 1 – 6

5

\*

## Lanjutkan ke operator “(”, lalu masukkan ke dalam stack saja

5 \* ( 2 + 3 ) / 1 – 6

(

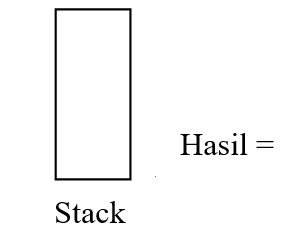
\*

## 

5

## Lanjutankan ke operand 2, karena operand maka 2 dijadikan hasil saja

5 \* ( 2 + 3 ) / 1 – 6



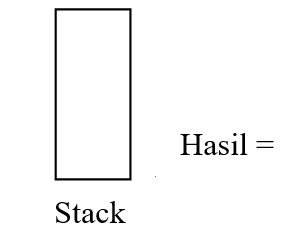
## 

(

\*

5 2

## Lanjutkan ke operator “+”, operator ini dimasukkan ke stack saja

5 \* ( 2 + 3 ) / 1 – 6

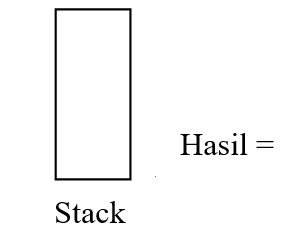
+

(

\*

5 2

## Lanjutankan ke operand 3, karena operand maka 3 dijadikan hasil saja

5 \* ( 2 + 3 ) / 1 – 6

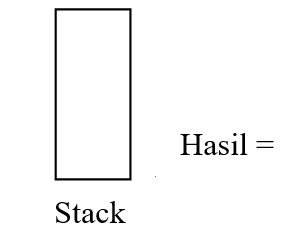
+

(

\*

5 2 3

## Lanjut ke operator “)”, operator ini akan mengeluarkan seluruh isi stack dimulai dari top stack hingga bertemu operator “(” yang jadi pasangannya. Karena diantara operator “(” dan “)” hanya ada “+” maka hanya operator “+” saja yang dijadikan hasil.Tanda kurung tidak usah dimasukkan



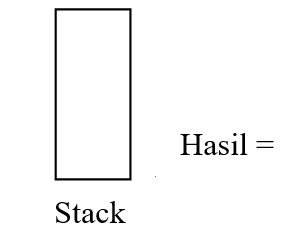
5 \* (2 + 3 ) / 1 – 6

\*

5 2 3 +

1. Lanjutkan ke operator “/“, operator ini akan dimasukkan ke dalam *stack*. Karena di *top stack* sudah ada isinya, maka bandingkan keduanya. Bila yang akan masuk memiliki derajat yang lebih besar, maka tumpuk saja. Sebaliknya, bila yang akan masuk memiliki derajat yang sama atau lebih kecil, maka keluarkan *top stack* hingga operator yang berada di *top stack* berderajat lebih kecil dari operator yang akan masuk.

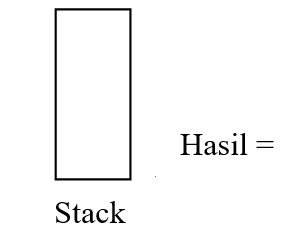
Karena “/” berderajat sama dengan “\*” maka keluarkan *top stack* (“\*”). Karena *stack* sudah hampa, maka operator “/” dimasukkan ke dalam *stack* sebagai *top stack*nya.

 5 \* (2 + 3 ) / 1 – 6

/

5 2 3 + \*

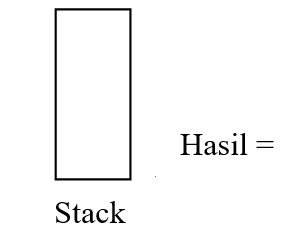
1. Lanjutkan ke *operand* “1”, karena sebagai *operand*, maka “1” dijadikan hasil saja.

 5 \* (2 + 3 ) / 1 – 6

/

5 2 3 + \* 1

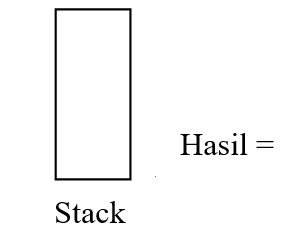
1. Lanjutkan ke operator “-“, operator ini akan dimasukkan ke dalam *stack*. Karena di *top stack* sudah ada isinya, maka bandingkan keduanya. Karena “-“ berderajat lebih kecil dari “/” maka operator “/” dikeluarkan dari tumpukan dan dijadikan hasil.

 5 \* (2 + 3 ) / 1 – 6

-

5 2 3 + \* 1 /

1. Lanjutkan ke *operand* “6”, karena sebagai *operand*, maka “6” dijadikan hasil saja.



5 \* (2 + 3 ) / 1 – 6

-

5 2 3 + \* 1 / 6

1. Karena proses telah selesai, maka keluarkan seluruh isi stack mengikuti kaidahnya, *last in first out*. Karena hanya ada “-“ maka hasil akhirnya menjadi:

## 5 2 3 + \* 1 / 6 -

## Cara binary tree

## 5 \* (2 + 3 ) / 1 – 6

## Langkah pertama adalah melihat terlebih dahulu derajat operasi dari notasi infix yang akan diproses, setelahnya disederhanakan menjadi inisial huruf atau angka. Diperoleh ( 2 + 3) lalu anggap saja menjadi A, maka notasi infix menjadi

## 5 \* A / 1 – 6

## Setelah itu sederhanakan pula 5 \* A menjadi I, maka notasi infix menjadi

## I / 1 - 6

## Bedasarkan notasi infix di langkah kedua maka sederhanakan lagi I / 1 menjadi J, maka notasi infix menjadi J – 6.

## Setelah selesai proses diatas maka selanjutnya kita ubah notasi infix menjadi postfix menggunakan tree. Menurut langkah ketiga yaitu J – 6 ,maka struktur pohon binernya adalah

## Karena J merupakan hasil penyederhana dari operasi I / 1 maka struktur pohon binernya adalah

## setelah itu gabungkan dengan struktur pohon yang sebelumnya menjadi seperti ini

## Karena I merupakan hasil penyederhana dari operasi 5 \* A maka struktur pohon binernya adalah

## setelah itu gabungkan dengan struktur pohon yang sebelumnya menjadi seperti ini

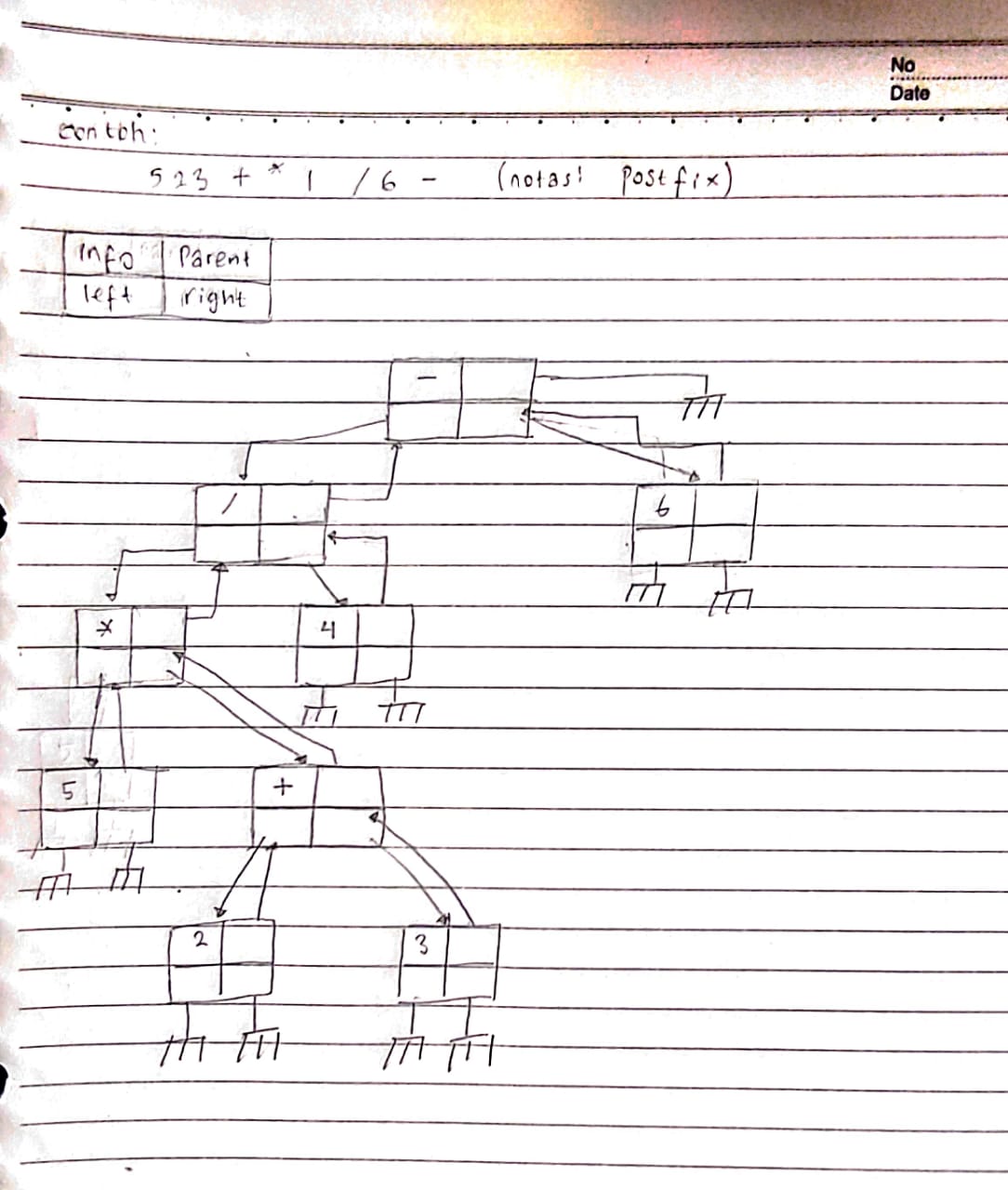
## Karena A juga merupakan hasil penyederhana dari operasi 5 \* A maka struktur final dari pohon binernya adalah

## Identifikasi Proses

* 1. Membaca inputan dari user
  2. Mengubah notasi infix menjadi postfix
  3. Melakukan perhitungan
  4. Menampilkan hasil

## Identifikasi Data

Representasi Struktur data :



**BAB II KESIMPULAN**

Program kalkulator yang dibuat akan menggunakan struktur data list stack dan struktur data tree, serta menggunakan proses prefix, infix dan postfix. Konsep dari program ini adalah mengubah infix menjadi postfix menggunakan **stack**, setelah itu menggunakan struktur data tree untuk notasi aritmatika. Dalam program kalkulator yang akan kami buat kira kira kami akan membuat fitur penjumlahan, penguraangan, perkalian dan pembagian.

# DAFTAR PUSTAKA

# DAFTAR KONTRIBUSI ANGGOTA KELOMPOK

|  |  |
| --- | --- |
| No. | : 1 |
| Nama | : Alifah Fisalsabilawati |
| Kontribusi | :  1. Mengerjakan bagian 1.1 Deskripsi program bagian deskripsi  2.  3. |

|  |  |
| --- | --- |
| No. | : 2 |
| Nama | : Faishal Muhammad |
| Kontribusi | :  1. Mengerjakan bagian 1.1 Deskripsi program bagian konsep program  2.  3. |