BAB III

METODE PENELITIAN / PERANCANGAN SISTEM

3.1. Metode Penelitian

Metodologi penelitian yang digunakan untuk mendukung penyelesaian perancangan dan pembuatan program ini meliputi :

1. Studi literatur

   Mempelajari referensi baik buku maupun web site yang berhubungan dengan konversi notasi infix, prefix, dan postfix.

2. Analisa permasalahan

   Melakukan analisa mengenai bagaimana mengkonversi notasi infix, prefix, postfix serta bagaimana mengimplementasikan dalam struktur data melalui pemanfaatan stack.

3. Perancangan dan pembuatan program

   Setelah analisa permasalahan untuk mencapai tujuan dari pembuatan program ini, maka penulis merancang dan membuat program untuk mengkonversi notasi infix, prefix, postfix yang dijelaskan melalui algotritma dan flowchart.

4. Uji coba dan implementasi program

   Menguji coba program yang sudah dibuat dengan mengambil beberapa contoh notasi, dari hasil uji coba akan diketahui konversi dari ketiga notasi tersebut.

5. Dokumentasi dan Penulisan Laporan Tugas Akhir

   Menyusun dan membuat buku Tugas Akhir, yang berisi tentang segala sesuatu yang berhubungan dengan sistem ini.

3.2. Perencanaan Program

Secara umum program konversi ini digambarkan dengan diagram blok seperti gambar di bawah ini :

Gambar 3.1. Blok diagram program secara global

User menginputkan satu ungkapan numeris, kemudian ungkapan tersebut diproses atau dikonversi menjadi beberapa notasi hingga menghasilkan suatu output.

3.3. Perancangan Proses Konversi Notasi Prefix ke Infix

Konversi notasi prefix ke infix dijelaskan dalam algoritma dan flowchart di bawah ini :

a) Algoritma konversi notasi prefix ke Infix

   1. Set derajat valensi

      [^ → valensi derajat 3

      *, / → valensi derajat 2

      +, - → valensi derajat 1]

2. Valensi ← 3

   [menset valensi ke harga tertinggi yang berarti bahwa yang dikerjakan terlebih dahulu adalah operator yang mempunyai valensi tertinggi]

3. ***While*** (valensi >0) ***do***

   - Cari notasi yang sesuai dengan aturan notasi prefix [operator + operand + operand] dan operator tersebut mempunyai derajat valensi yang sama dengan nilai variabel valensi.
   - **If** (ketemu=true) **then**

     a. Konversikan ke notasi infix [operand + operator + operand] dengan terlebih dahulu menambahkan parantesis berupa kurung buka dan kurung tutup.

        ['(' + operand + operator + operand + ')']

        { Hasil konversi diperlakukan sebagai operand baru dan diletakkan pada satu alamat stack. }

     b. Geser operand atau operator yang terletak pada alamat, diatas operand hasil konversi

     c. Valensi ←3

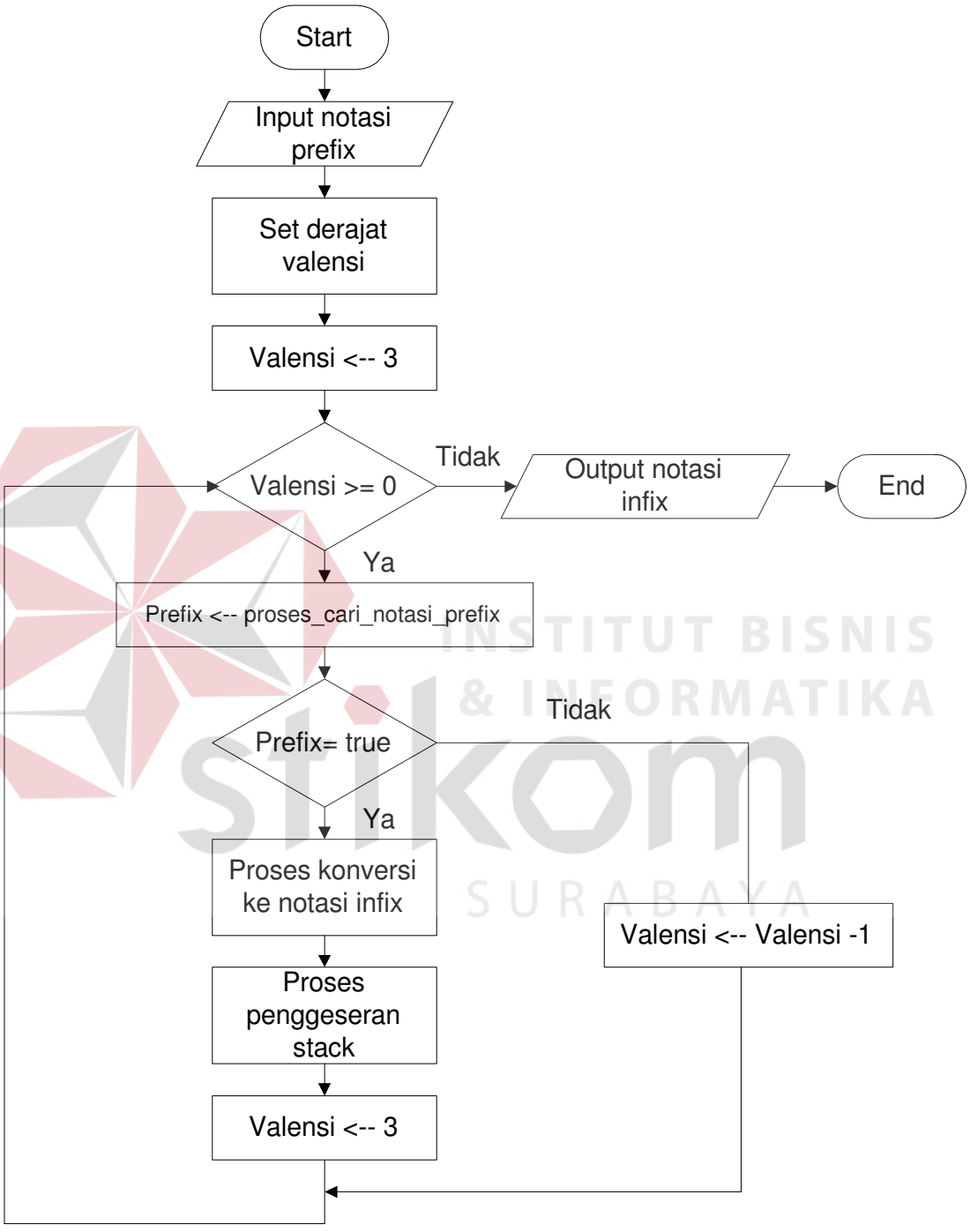
   Else

   Valensi← valensi -1

4. [Selesai]

   Return

b. Flowchart konversi notasi prefix ke Infix



Gambar 3.2 Flowchart Konversi Notasi Prefix ke Infix

3.4. Perancangan Proses Konversi Notasi Prefix ke Postfix

Untuk mengkonversi notasi prefix ke postfix dijelaskan melalui algoritma dan flowchart sebagai berikut :

a) Algoritma konversi notasi prefix ke postfix

1. Set derajat valensi

   [^ → valensi derajat 3

   *, / → valensi derajat 2

   +, - → valensi derajat 1]

2. Valensi ← 3

   [menset valensi ke harga tertinggi yang berarti bahwa yang dikerjakan terlebih dahulu adalah operator yang mempunyai valensi tertinggi]

3. ***While*** (valensi >0) ***do***

   - Cari notasi yang sesuai dengan aturan notasi prefix [operator + operand + operand] dan operator tersebut mempunyai derajat valensi yang sama dengan nilai variabel valensi.

   - **If** (ketemu=true) **then**

     a. Konversikan ke notasi postfix [operand + operand + operator]

        { Hasil konversi diperlakukan sebagai operand baru dan diletakkkan pada satu alamat stack. }

     b. Geser operand atau operator yang terletak pada alamat, diatas operand hasil konversi

     c. Valensi ←3

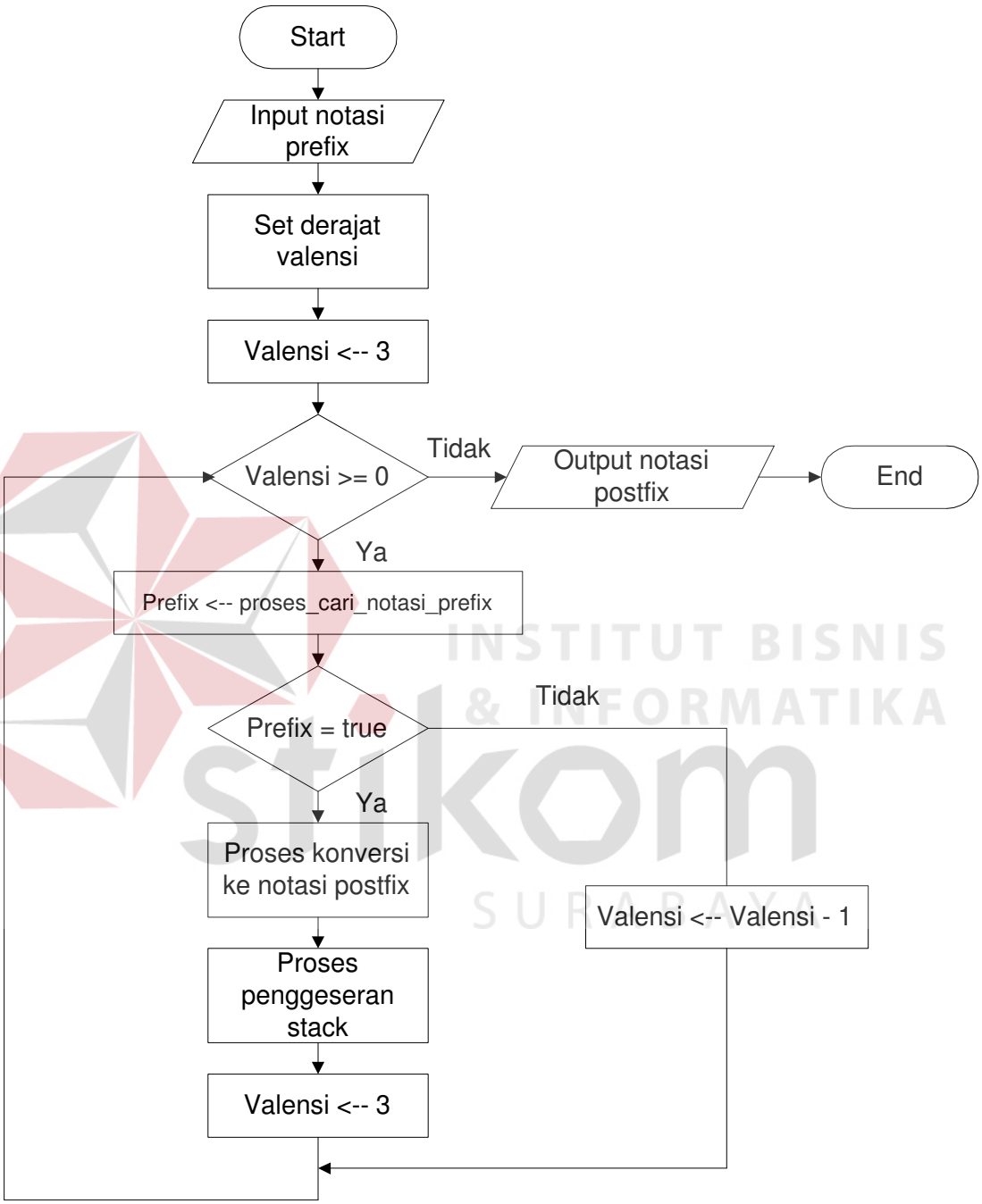
Else

Valensi← valensi -1

4. [Selesai]

Return

b) Flowchart konversi notasi prefix ke postfix



Gambar 3.3. Flowchart konversi notasi prefix ke postfix

3.5. Perancangan Proses Konversi Notasi Postfix ke Prefix

Proses konversi notasi postfix ke prefix merupakan kebalikan dari konversi prefix ke postfix, proses tersebut dijabarkan dalam algoritma dan flowchart sebagai berikut :

a) Algoritma konversi notasi postfix ke prefix

1. Set derajat valensi

   [^ → valensi derajat 3

   *, / → valensi derajat 2

   +, - → valensi derajat 1]

2. Valensi ← 3

   [menset valensi ke harga tertinggi yang berarti bahwa yang dikerjakan terlebih dahulu adalah operator yang mempunyai valensi tertinggi]

3. ***While*** (valensi >0) ***do***

   - Cari notasi yang sesuai dengan aturan notasi postfix [operand + operand + operator] dan operator tersebut mempunyai derajat valensi yang sama dengan nilai variabel valensi.
   - **If** (ketemu=true) **then**

     a. Konversikan ke notasi prefix [operator + operand + operand]

        { Hasil konversi diperlakukan sebagai operand baru dan diletakkkan pada satu alamat stack. }

     b. Geser operand atau operator yang terletak pada alamat, diatas operand hasil konversi

c. Valensi ←3

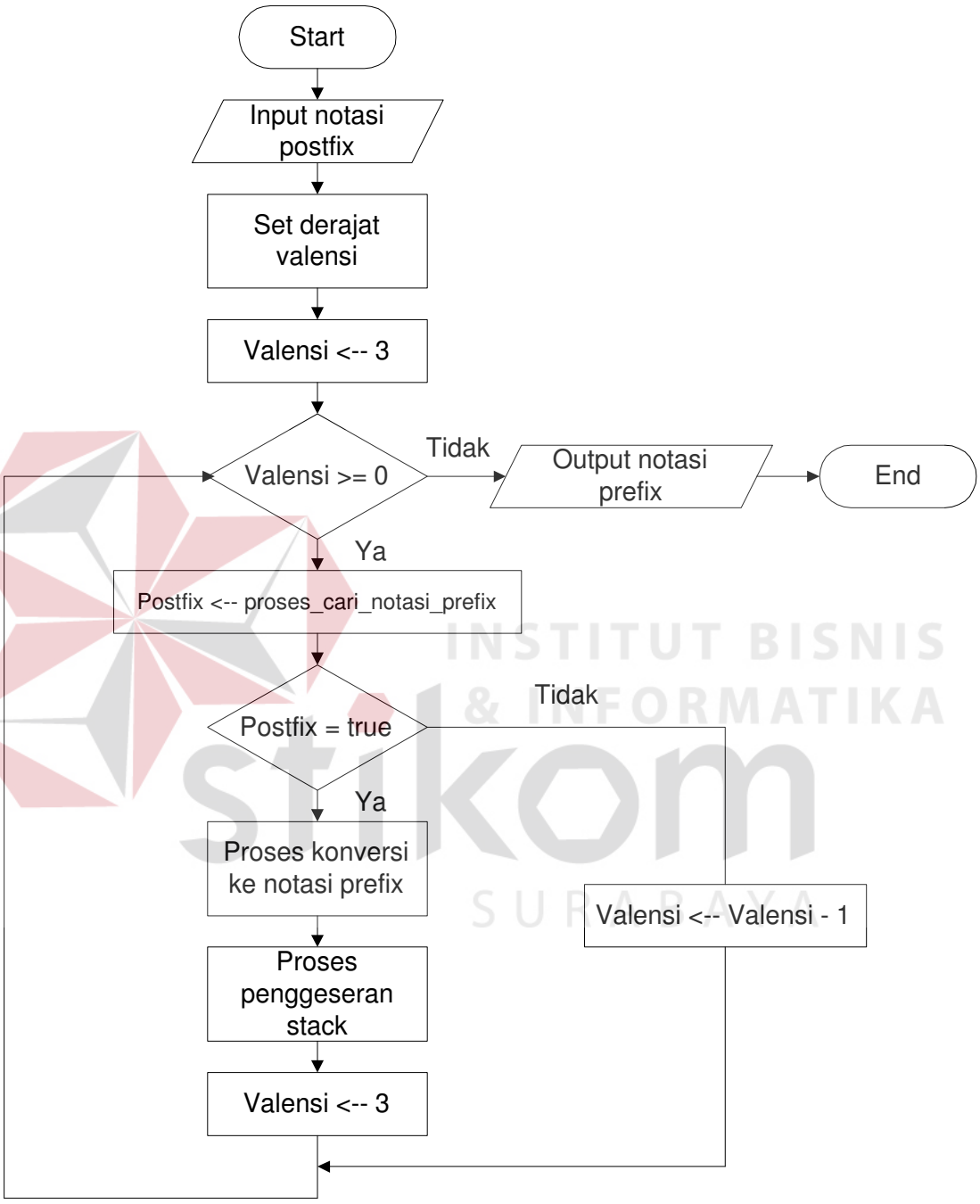
Else

Valensi← valensi -1

4. [Selesai]

Return

b) Flowchart konversi notasi postfix ke prefix



Gambar 3.4. Flowchart Konversi Notasi Postfix ke Prefix

3.6. Perancangan Proses Konversi Notasi Postfix ke Infix

Pada dasarnya konversi notasi postfix ke infix tidak jauh beda atau sama dengan proses konversi notasi prefix ke infix, perbedaanya hanya pada letak operatornya saja, berikut algoritma dan flowchart konversi notasi postfix ke infix :

a) Algoritma Konversi Postfix ke Infix

1. Set derajat valensi

   [^ → valensi derajat 3

   *, / → valensi derajat 2

   +, - → valensi derajat 1]

2. Valensi ← 3

   [menset valensi ke harga tertinggi yang berarti bahwa yang dikerjakan terlebih dahulu adalah operator yang mempunyai valensi tertinggi]

3. ***While*** (valensi >0) ***do***

   - Cari notasi yang sesuai dengan aturan notasi postfix [operand + operand + operator] dan operator tersebut mempunyai derajat valensi yang sama dengan nilai variabel valensi.
   - **If** (ketemu=true) **then**

     a. Konversikan ke notasi infix [operand + operator + operand] dengan terlebih dahulu menambahkan parantesis berupa kurung buka dan kurung tutup.

     ['(' + operand + operator + operand + ')']

{ Hasil konversi diperlakukan sebagai operand baru dan diletakkan pada satu alamat stack. }

b. Geser operand atau operator yang terletak pada alamat, diatas operand hasil konversi

c. Valensi ←3

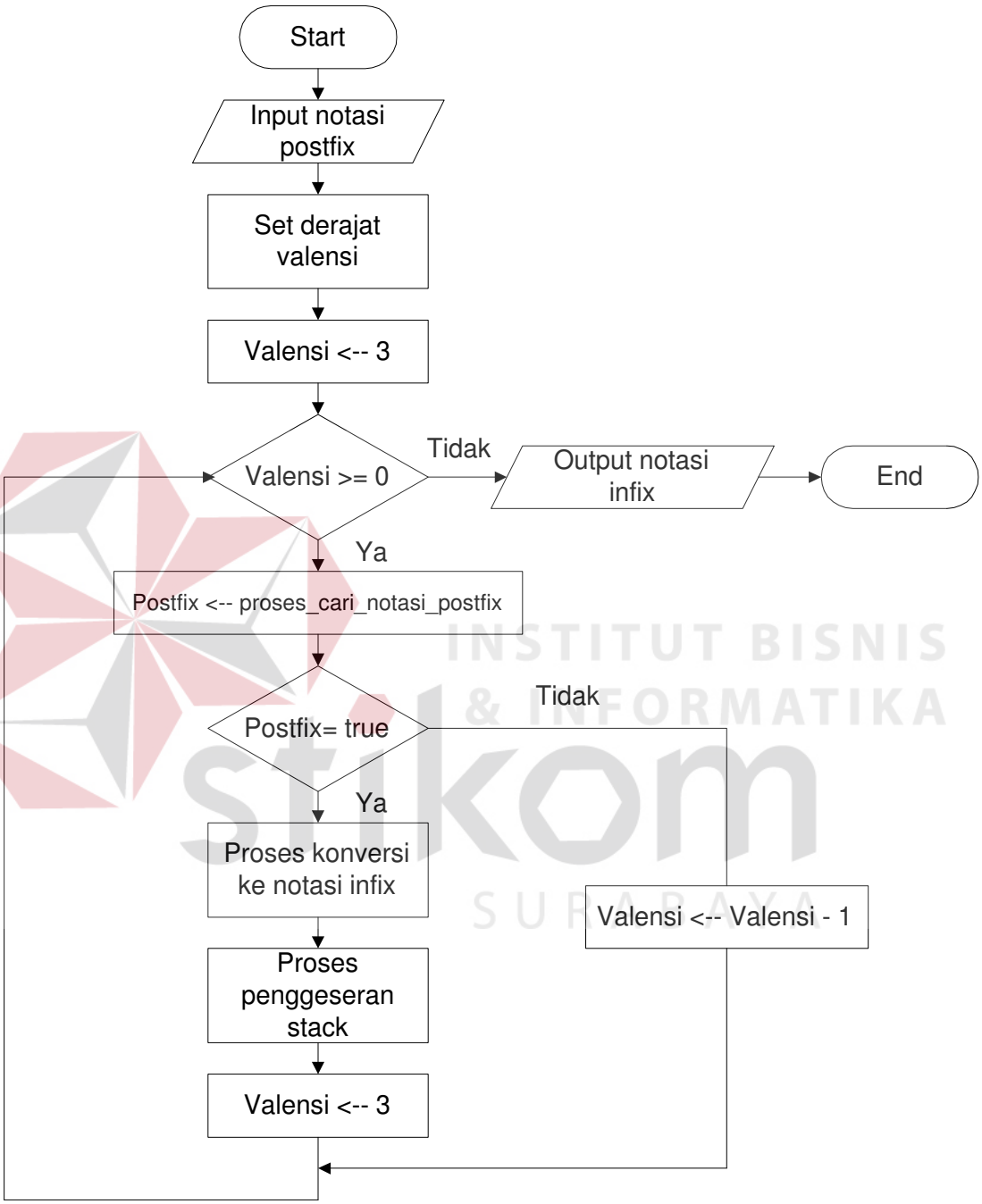
Else

Valensi← valensi -1

4. [Selesai]

Return

b) Flowchart Konversi Postfix ke Infix



Gambar 3.5 Konversi notasi postfix ke infix

3.7. Perancangan Proses Konversi Notasi Infix ke Prefix

Konversi notasi prefix ke infix dijelaskan dalam algoritma dan flowchart dibawah ini :

a. Algoritma konversi notasi infix ke prefix

1. Set derajat valensi

   [^ → valensi derajat 3

   *, / → valensi derajat 2

   +, - → valensi derajat 1]

2. Valensi ← 3

   [menset valensi ke harga tertinggi yang berarti bahwa yang dikerjakan terlebih dahulu adalah operator yang mempunyai valensi tertinggi]

3. Proses pengecekan jumlah kurung

4. ***While*** (valensi >0) ***do***

   - Cari notasi yang sesuai dengan aturan notasi '('+ infix [operand + operand + operator] +')'

   - **If** (ketemu=true) **then**

     a. Konversikan ke notasi prefix [operand + operator + operand] dengan terlebih dahulu menghapus parantesis berupa kurung buka dan kurung tutup.

        { Hasil konversi diperlakukan sebagai operand baru dan diletakkan pada satu alamat stack }

b. Geser operand atau operator yang terletak pada alamat, diatas operand hasil konversi

- Cari notasi yang sesuai dengan aturan notasi infix (tanpa kurung)
- **If** (ketemu=true) **then**

a. Konversikan ke notasi prefix [operand + operator + operand] dengan terlebih dahulu menghapus parantesis berupa kurung buka dan kurung tutup.

{ Hasil konversi diperlakukan sebagai operand baru dan diletakkan pada satu alamat stack. }

b. Geser operand atau operator yang terletak pada alamat, diatas operand hasil konversi

c. Valensi ←3

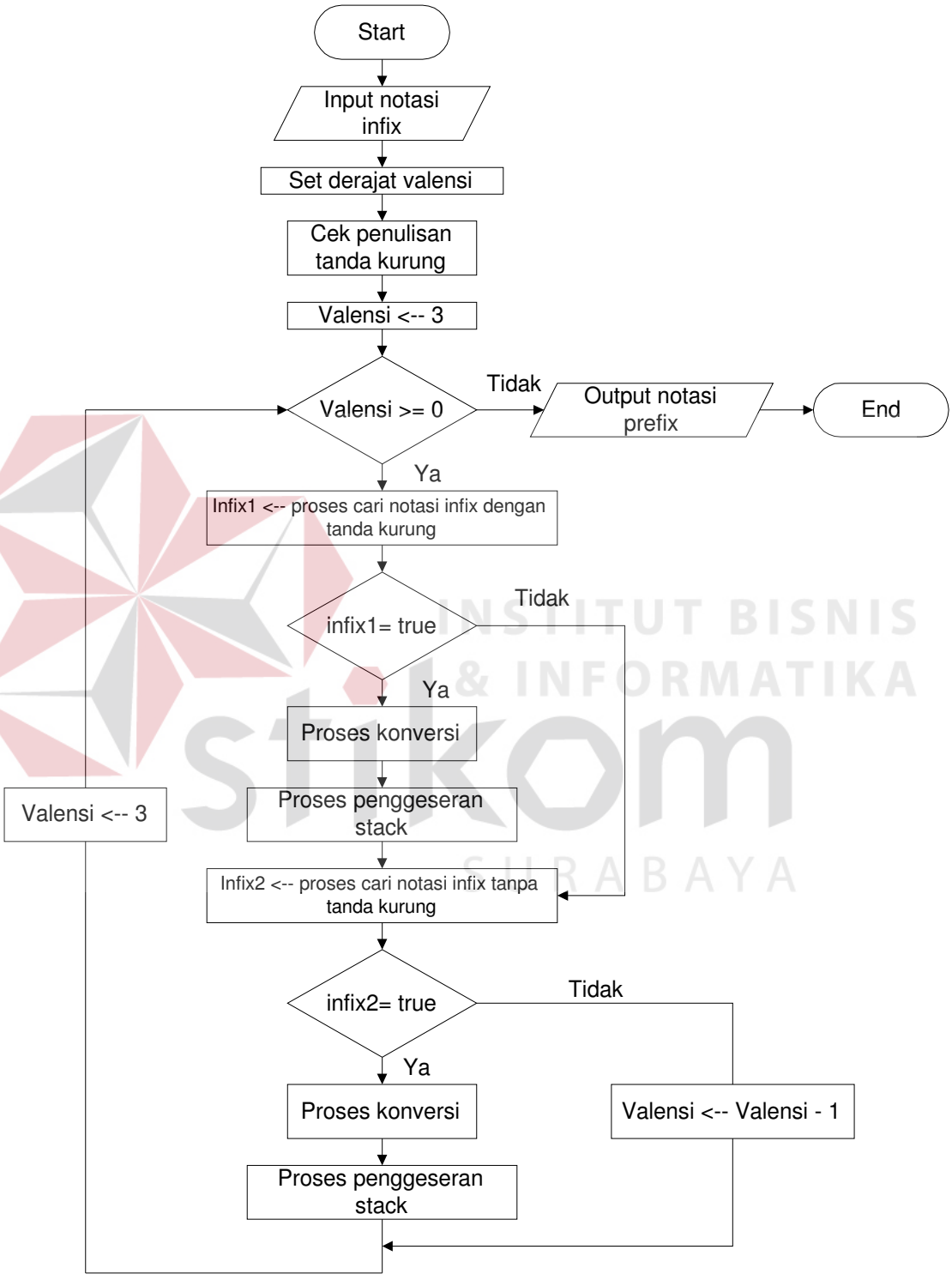
Else

Valensi← valensi -1

5. [Selesai]

Return

b. Flowchart konversi notasi infix ke prefix



Gambar 3.6 Konversi Notasi Infix ke Prefix

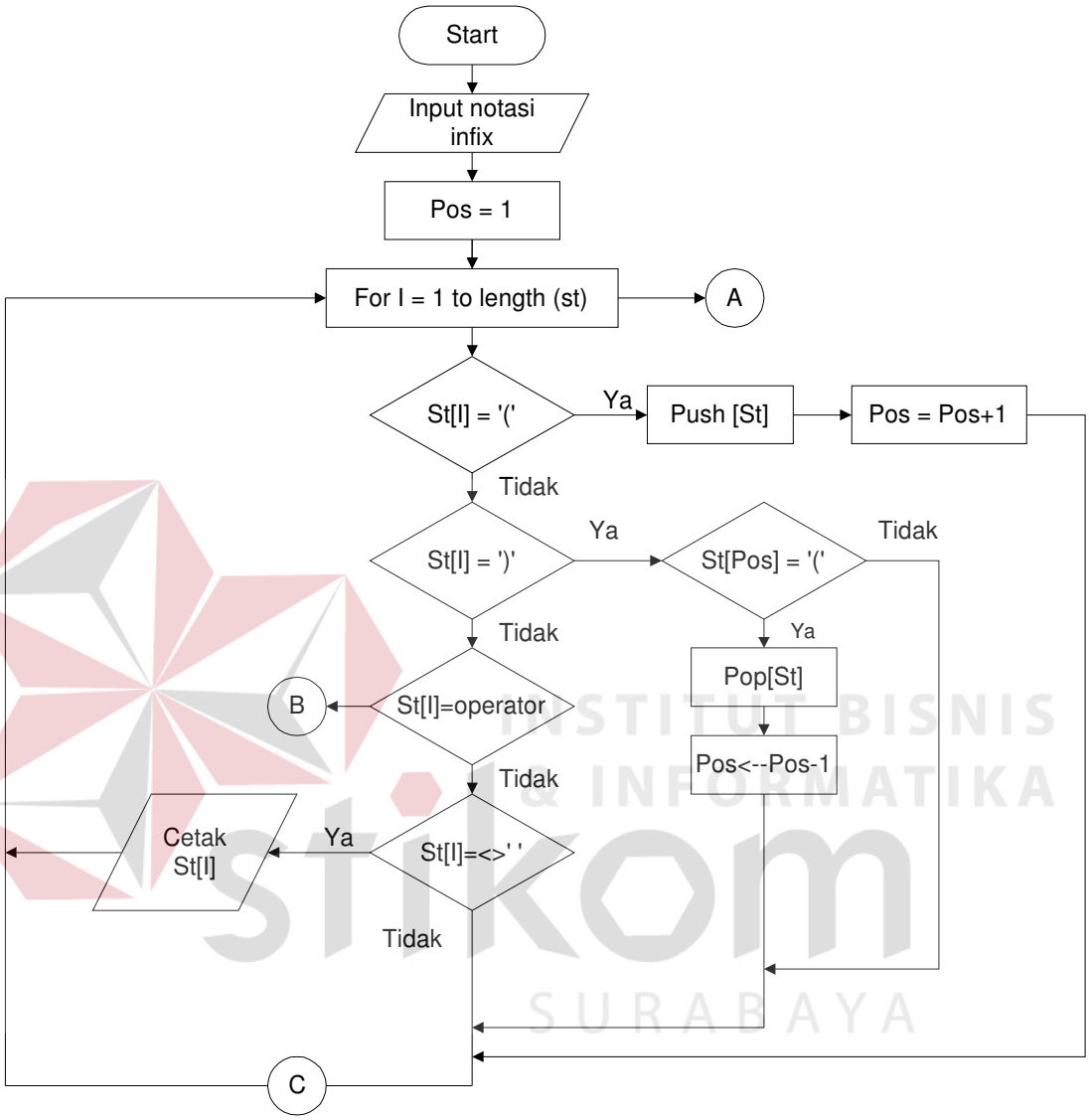
3.8. Perancangan Proses Konversi Notasi Infix ke Postfix

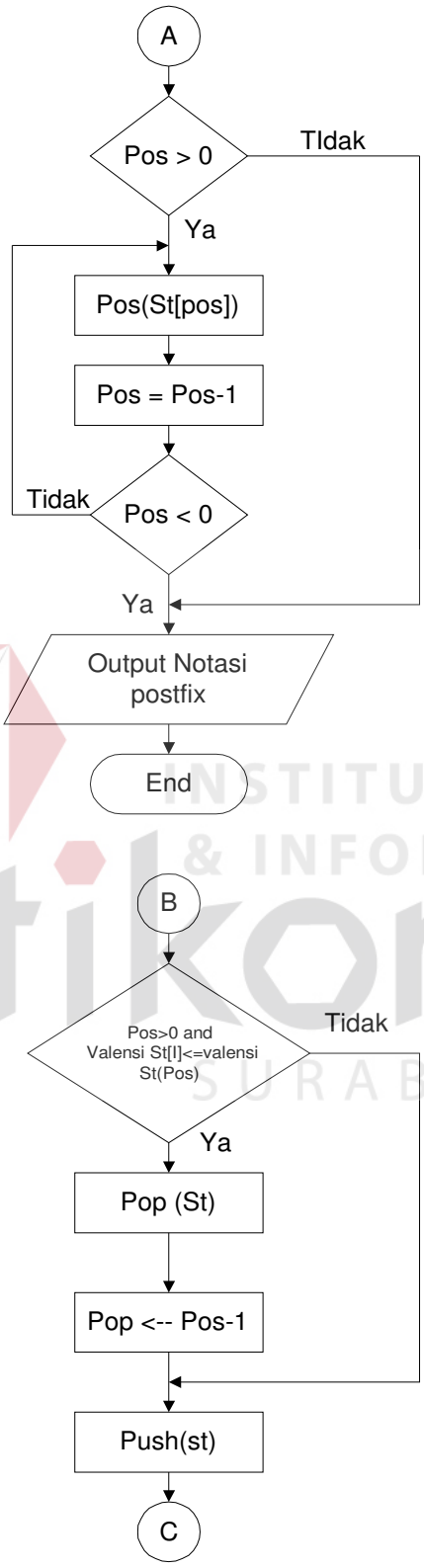
Ilustrasi pengubahan notasi infix menjadi notasi postfix secara lengkap tersaji dalam algoritma dan flowchart dibawah ini :

a. Algoritma konversi notasi infix ke postfix

1. Baca ungkapan notasi infix per karakter
2. For I = 1 to panjang karakter do
    a. Jika stack[I]=operand maka langsung dicetak
    b. Jika stack[I]='(', push ke dalam tumpukan
    c. Jika stack[I] = operator, maka
        - Jika stack paling atas adalah '(', push operator
        - Jika operator memiliki prioritas lebih tinggi dari pada derajat ujung paling atas stack, maka push operator.
        - Jika tidak, pop operator dari stack lalu cetak, kemudian ulangi step 4
5. Jika stack[I] = ')', pop operator kemudian cetak sampai ketemu '(', tetapi '(' tidak usah ditulis.
6. Jika stack[I] <> 0, kembali ke step 1
7. Jika stack[I]=0, cetak seluruh notasi yang ada di stack operand

b. Flowchart konversi notasi infix ke postfix



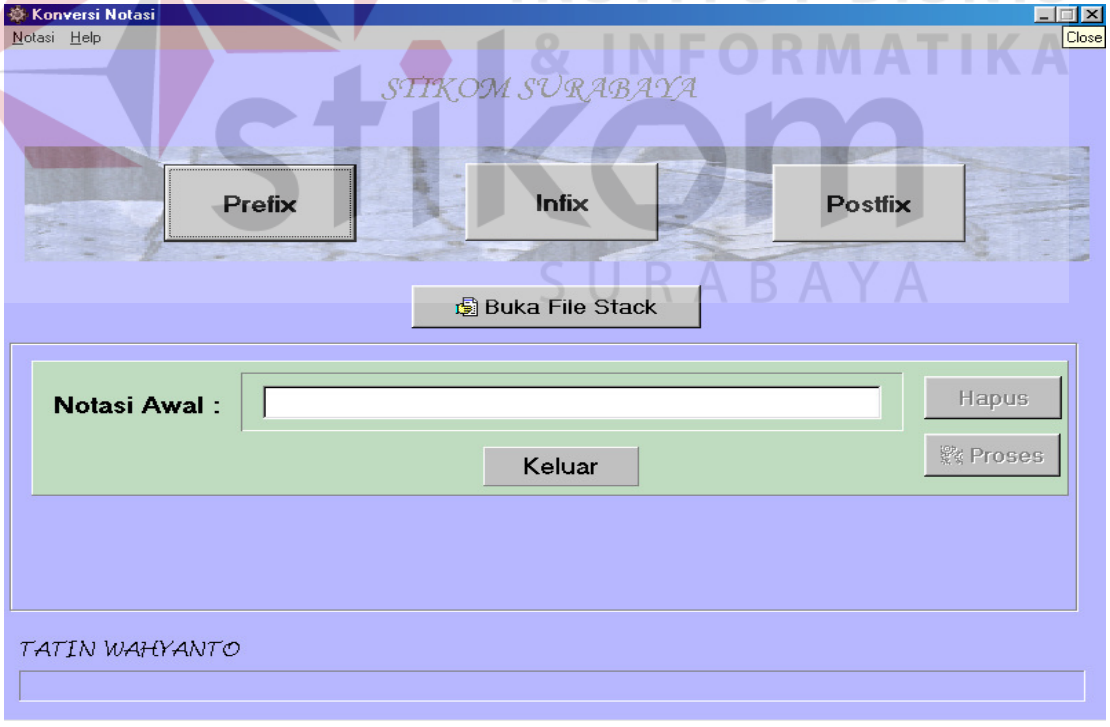


Gambar 3.7 Konversi Notasi Infix ke Postfix

3.9. Desain Input Output

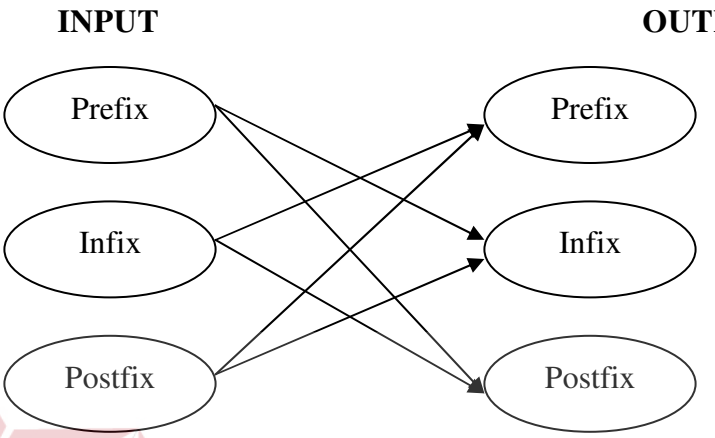
Pada perancangan desain input output ini dibuat dengan menggunakan alat-alat input komputer seperti penggunaan mouse ataupun keyboard untuk mempermudah proses konversi dan hanya terdiri dari satu menu utama saja.

Pernyataan matematis ditulis pada Notasi Awal (seperti tampak pada gambar), dengan menentukan notasi yang akan diproses terlebih dahulu, yaitu notasi prefix, infix, atau postfix. Karakter yang digunakan untuk operand pada notasi prefix, infix, dan postfix adalah huruf A sampai dengan Z, a sampai dengan z, sedangkan untuk operator adalah ^, *, /, +, -. Khusus untuk input notasi infix ada tambahan karakter berupa parentesis '(' dan ')'.



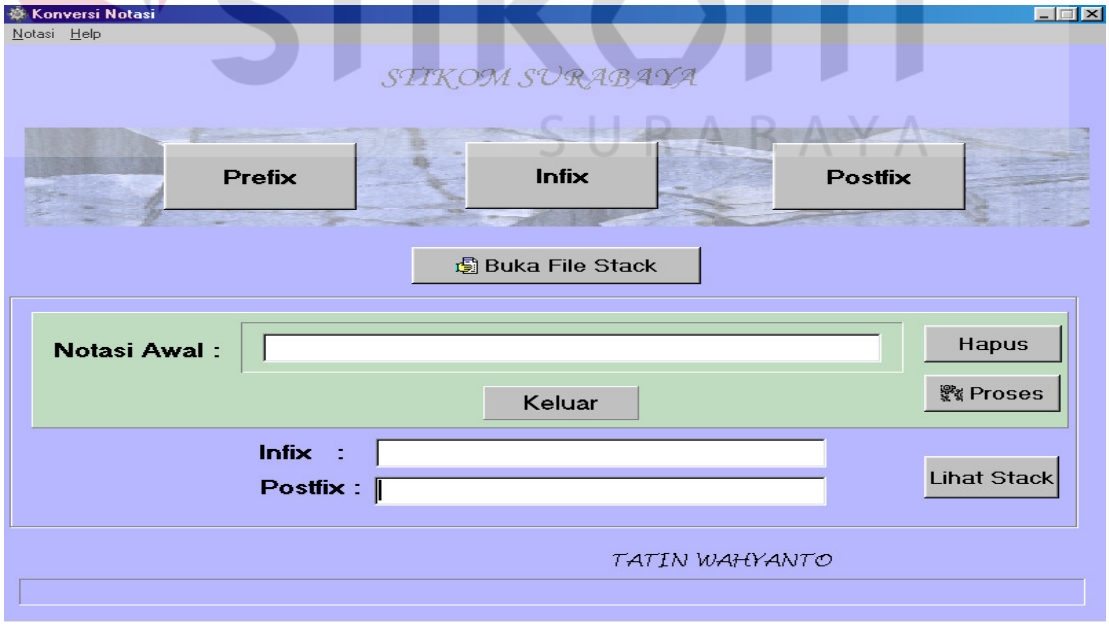
Gambar 3.8 Desain Input

Setelah input pernyataan matematis, maka dilakukan proses konversi dengan menekan tombol proses. Proses perubahan konversi ditunjukkan pada gambar dibawah ini :



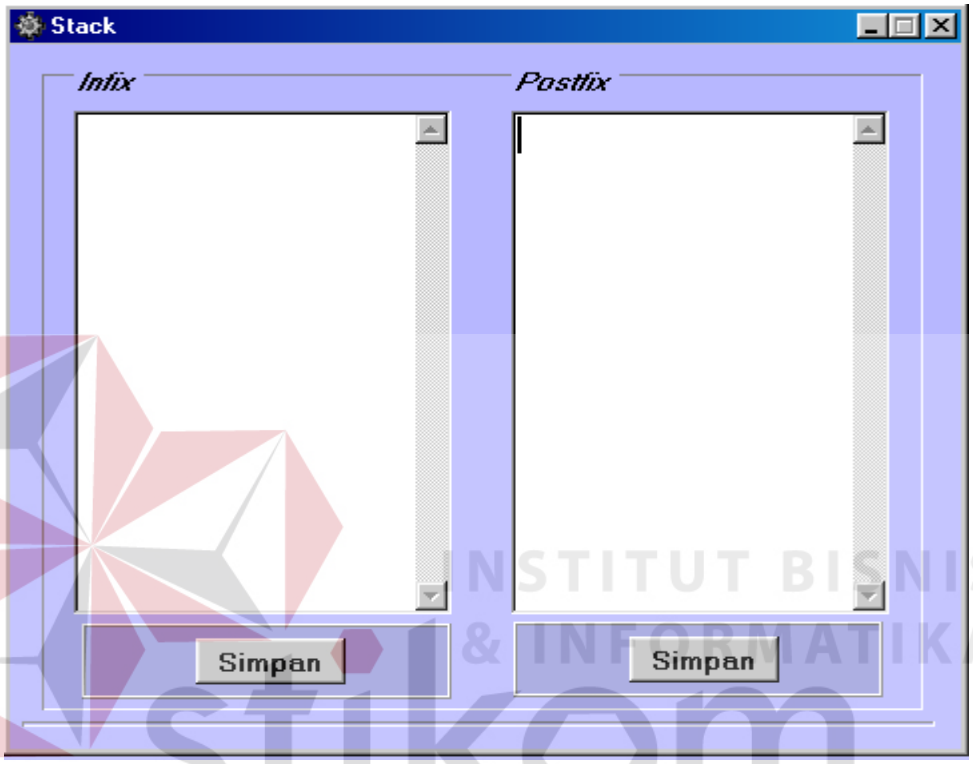
Gambar 3.9 Proses Konversi

Jika notasi yang diinputkan berupa notasi prefix, maka hasil output akan berupa notasi infix dan postfix, seperti terlihat pada gambar dibawah ini :



Gambar 3.10 Desain output konversi notasi prefix

Untuk melihat implementasi stack, tekan tombol lihat stack, maka akan muncul form baru seperti terlihat pada gambar dibawah ini :

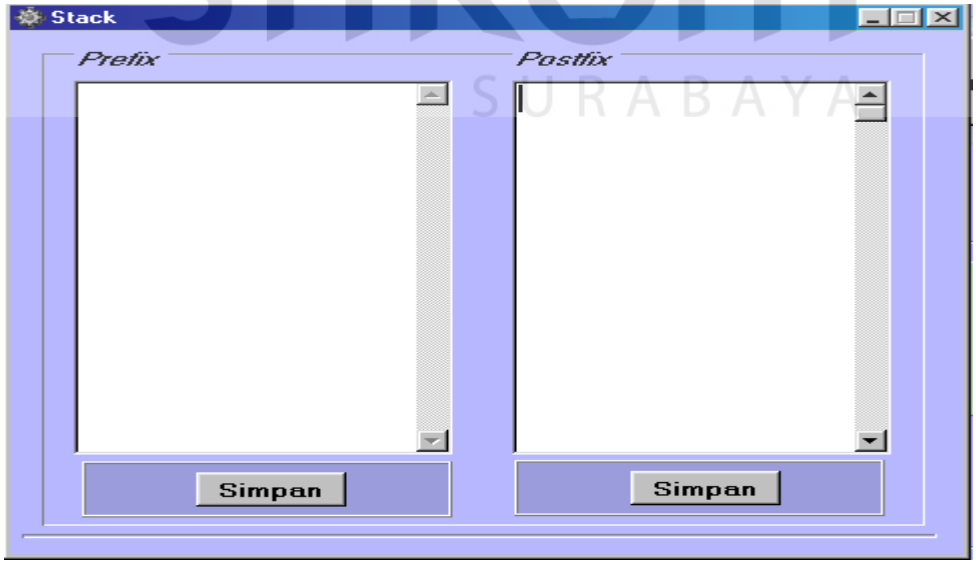


Gambar 3.11 Desain output stack konversi notasi prefix

Tombol Simpan digunakan untuk menyimpan file hasil konversi. Jika notasi yang diinputkan berupa notasi infix, maka hasil output akan berupa notasi prefix dan postfix, seperti terlihat pada gambar dibawah ini :

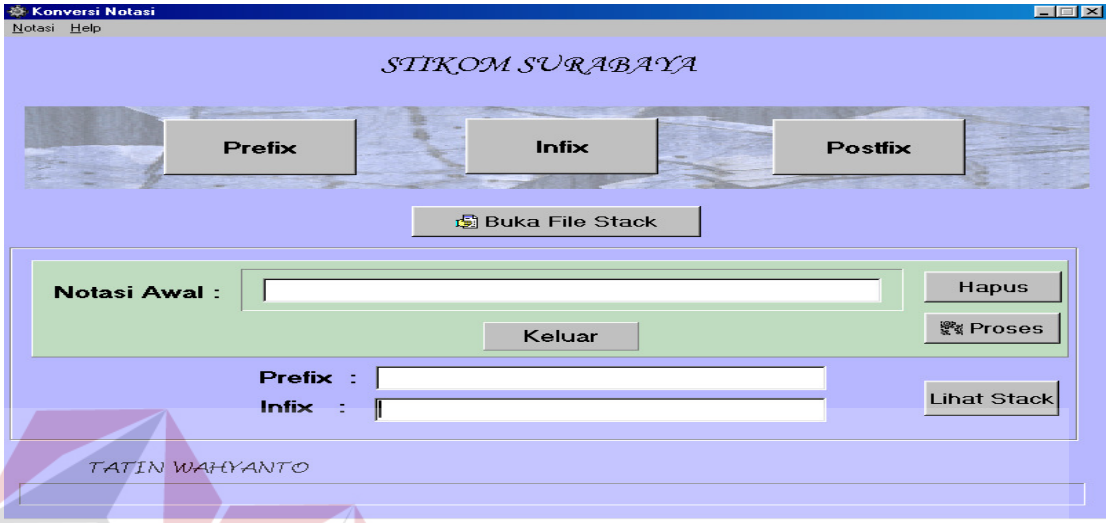
Gambar 3.12 Desain output konversi notasi infix

Untuk melihat implementasi stack, tekan tombol lihat stack, maka akan gambar seperti terlihat pada gambar dibawah ini :



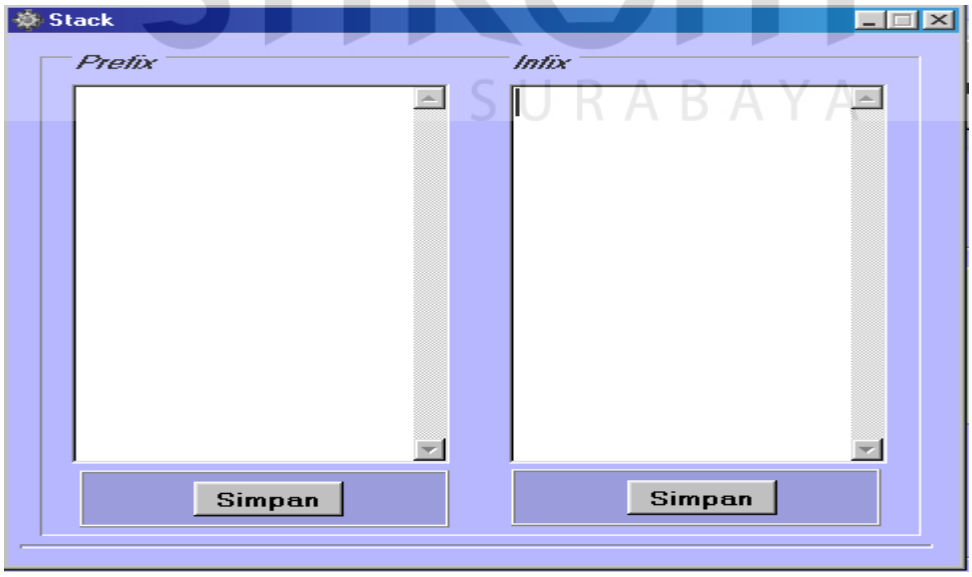
Gambar 3.13 Desain output stack konversi notasi infix

Jika notasi yang diinputkan berupa notasi infix, maka hasil output akan berupa notasi prefix dan postfix, seperti terlihat pada gambar dibawah ini :



Gambar 3.14 Desain output konversi notasi postfix

Untuk melihat implementasi stack, tekan Tombol Lihat Stack, maka akan muncul form baru seperti terlihat pada gambar dibawah ini :



Gambar 3.15 Desain output stack konversi notasi postfix