Jurnal Computech & Bisnis, Vol. 5, No. 1, Juni 2011, 51-57
ISSN 1978-9629

RANCANG BANGUN PERANGKAT LUNAK SIMULASI PEMBELAJARAN POHON EKSPRESI (*EXPRESSION TREE*) DARI EKSPRESI ARITMATIKA *PREFIX*, *INFIX* DAN *POSTFIX*

Sumadi, Haryoko
STMIK Mardira Indonesia, Bandung

Abstract

Along with the development of human civilization and the rapid advances in technology, computers have unwittingly contributed to the world of education, especially its use as a teaching tool. Learning with computer software arises from a number of disciplines, especially computer science and psychology. From computer science and mathematics appeared programs that make all the calculations and functions easier and more useful.
Simulation Software Learning Tree Expression is used to help track the expression tree devoted to completing arithmetic expressions. There are 3 forms of arithmetic expressions that Prefix, Infix and Postfix. The software is also intended to assist in improving the effectiveness or power of knowledge and interest in training for users.
In the Design of Simulation Software Learning Tree Expression from Arithmetic Expression Prefix, Infix and Postfix, the author uses the object - oriented method. Software built using Visual Basic 6.0.

Keywords: *Arithmetic, Prefix, Infix, Postfix*

Abstrak

Seiring dengan perkembangan peradaban manusia dan kemajuan pesat di bidang teknologi, tanpa disadari komputer telah ikut berperan dalam dunia pendidikan terutama penggunaannya sebagai alat bantu pengajaran. Perangkat lunak pembelajaran dengan komputer muncul dari sejumlah disiplin ilmu, terutama ilmu komputer dan psikologi. Dari ilmu komputer dan matematika muncul program – program yang membuat semua perhitungan dan fungsi lebih mudah dan bermanfaat.
Perangkat Lunak Simulasi Pembelajaran Pohon Ekspresi (*Expression Tree*) digunakan untuk membantu penelusuran pohon ekspresi yang ditujukan dalam menyelesaikan ekspresi aritmatika. Ada 3 bentuk ekspresi aritmatika yaitu *Prefix, Infix* dan *Postfix*. Perangkat lunak ini juga dimaksudkan untuk membantu dalam meningkatkan pengetahuan serta efektifitas atau daya minat pelatihan bagi pengguna.
Dalam Rancang Bangun Perangkat Lunak Simulasi Pembelajaran Pohon Ekspresi (*Expression Tree*) Dari Ekspresi Aritmatika *Prefix*, *Infix* dan *Postfix* ini, penulis menggunakan metode berorientasi objek. Perangkat Lunak ini dibangun dengan menggunakan bahasa pemrograman Visual Basic 6.0.

Kata Kunci : Aritmatika, *Prefix, Infix, Postfix*

Latar Belakang Masalah

Teknologi informasi (TI) merupakan cara untuk mengelola dan meningkatkan nilai tambah dari suatu informasi. Salah satu contoh dari teknologi informasi adalah teknologi perangkat lunak (*software)*, yang merupakan suatu struktur informasi (*program*) yang digunakan untuk membantu mengolah informasi menjadi informasi baru yang bernilai lebih tinggi.

Manfaat yang dapat diambil dari ilmu informatika yaitu dapat memberikan suatu hasil yang lebih optimal, baik dari segi kualitas maupun kuantitas dalam hal kinerja sistem. Perkembangan ilmu tersebut semakin hari semakin meningkat, hal tersebut memungkinkan sistem yang dibuat secara komputerisasi bisa membantu manusia di berbagai kegiatan, bahkan dalam pengambilan keputusan sekalipun, yang tentunya sudah terukur secara tepat.

Pada awal pengembangannya, orientasi utama dari bahasa pemrograman adalah untuk membantu para *programmer* dalam menyelesaikan ekspresi – ekspresi aritmatika. *Compiler* yang dibangun harus memiliki kemampuan untuk menyelesaikan semua bentuk ekspresi aritmatika. Tahapan – tahapan penyelesaian suatu ekspresi aritmatika dapat direpresentasikan dalam bentuk *graph* yang dinamakan pohon ekspresi (*expression tree*).

Pohon Ekspresi (*expression tree*) adalah sebuah pohon biner (*binary tree*) dimana daun berisi operand yang terdapat dalam ekspresi aritmatika dan akar berisi operator yang terdapat dalam ekspresi aritmatika tersebut. Proses pembacaan dari pohon ekspresi dimulai dari daun paling kiri hingga akar utama. Operand dan operator yang berada pada level bawah akan dibaca terlebih dahulu. Penulisan ekspresi aritmatika terdiri dari 3 bentuk, yaitu bentuk *prefix, suffix (postfix)* dan *infix*. Dalam bentuk *prefix*, operator ditulis di depan dari operandnya, dan dalam bentuk *suffix (postfix)*, operator ditulis di belakang dari operandnya. Sedangkan bentuk *infix* merupakan bentuk penulisan normal dari ekspresi aritmatika.

Identifikasi Masalah

Berdasarkan latar belakang masalah, maka dapat diidentifikasikan masalah-masalah salah satunya adalah bagaimanakah menganalisis dan merancang suatu perangkat lunak pembelajaran pohon ekspresi (*expression tree*).

Batasan Masalah

Dari sejumlah identifikasi masalah yang dikemukakan diatas maka penulis membatasi masalah hanya pada rancang bangun Perangkat Lunak Simulasi Pembelajaran Pohon Ekspresi (*Expression Tree*) dari Ekspresi Aritmatika *Prefix*, *Infix* dan *Postfix* saja.

LANDASAN TEORI

Ekspresi Aritmatika

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

- *Binary operator* (operator pasangan)
- *Unary operator* (operator tunggal)

Binary operator adalah operator yang memiliki 2 buah operand (diapit oleh 2 buah operand), sedangkan *unary operator* adalah operator yang hanya memiliki 1 buah operand (diikuti oleh sebuah operand). Operator – operator yang termasuk dalam *binary operator* adalah operator penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/), modulo (mod), divisor (div), pemangkatan (^), operator logika *AND*, operator logika *OR*, dan operator perbandingan (seperti operator lebih besar, lebih kecil, sama dengan, lebih besar sama dengan, lebih kecil sama dengan, dan tidak sama dengan).

Sedangkan operator yang termasuk dalam *unary operator* adalah operator minus (~), operator faktorial (!), operator trigonometri (seperti operator *sinus, cosinus, tangen, cotangen, secan,* dan *cosecan*), operator logika *NOT*, 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 (^) 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* (=).

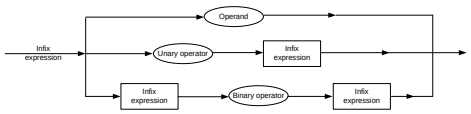
Ekspresi aritmatika akan diselesaikan berdasarkan urutan prioritas dari operator di atas dengan ketentuan operator yang memiliki prioritas yang lebih tinggi akan diselesaikan terlebih dahulu. Tahapan – tahapan penyelesaian suatu ekspresi aritmatika dapat direpresentasikan dalam bentuk *graph* yang dinamakan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

Notasi/Penulisan Ekspresi Aritmatika

Polish Notation diperkenalkan oleh seorang ahli matematika Polandia bernama Jan Lukasiewicz. *Polish Notation* merupakan notasi penulisan ekspresi aritmatika. *Polish Notation* terdiri dari 3 bentuk.

Infix

Bentuk *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*. Diagram bentuk *infix* dapat dilihat pada gambar di bawah ini.

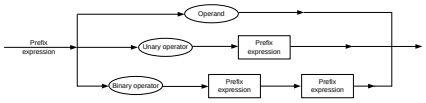


Gambar *Diagram bentuk infix*

Bentuk *infix* dari ekspresi aritmatika tersebut tidak mengubah bentuk penulisan ekspresi dan hanya melakukan pembuangan spasi-spasi yang berlebihan saja. Sebagai contoh, misalkan diketahui suatu ekspresi aritmatika x * y + 2 * (z – 3), maka bentuk *infix*-nya berupa x*y+2*(z–3).

Prefix

Bentuk *prefix* merupakan cara / bentuk penulisan ekspresi aritmatika dimana operator ditulis di depan dari operandnya. Suatu *prefix* dapat berupa operand tunggal, atau gabungan dari *unary operator* dengan *prefix*, ataupun berupa gabungan dari *binary operator* dengan dua buah *prefix*. Diagram bentuk *prefix* dapat dilihat pada gambar di bawah ini.



Gambar *Diagram bentuk prefix*

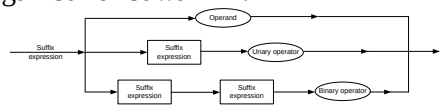
Sebagai contoh, misalkan diketahui suatu ekspresi aritmatika x * y + 2 * (z – 3), maka bentuk *prefix*-nya berupa : + * x y * 2 – z 3.

Proses pengubahan ekspresi aritmatika x * y + 2 * (z – 3) ke dalam bentuk *prefix* dilakukan berdasarkan urutan prioritas dari operator – operator yang terdapat di dalam ekspresi aritmatika tersebut. Proses dimulai dengan mengubah sub ekspresi (z – 3) menjadi - z 3 sehingga ekspresi aritmatika tersebut menjadi x * y + 2 * - z 3. Proses dilanjutkan dengan mengubah sub ekspresi x * y menjadi *

x y sehingga ekspresi aritmatika menjadi * x y + 2 * - z 3 dan dilanjutkan dengan mengubah bentuk 2 * - z 3 menjadi * 2 – z 3 sehingga ekspresi aritmatika menjadi * x y + * 2 – z 3. Proses diakhiri dengan mengubah ekspresi aritmatika menjadi bentuk prefix + * x y * 2 – z 3.

Suffix (Postfix)

Bentuk *suffix (postfix)* merupakan cara / bentuk penulisan ekspresi aritmatika dimana operator ditulis di belakang dari operandnya. Suatu *suffix* dapat berupa operand tunggal, atau gabungan dari *suffix* dengan *unary operator*, ataupun berupa gabungan dari dua buah *suffix* dengan *binary operator*. Diagram bentuk *suffix* dapat dilihat pada gambar di bawah ini.

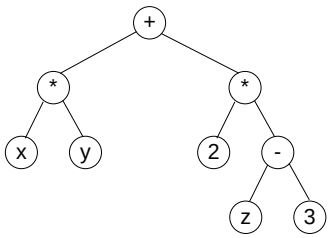


Gambar *Diagram bentuk suffix*

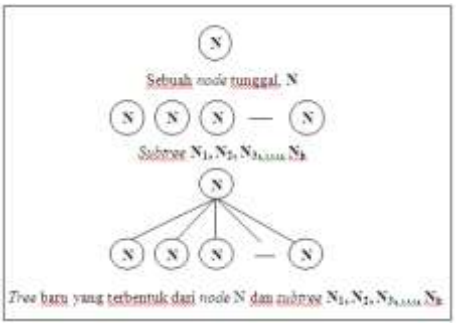
Sebagai contoh, misalkan diketahui suatu ekspresi aritmatika x * y + 2 * (z – 3), maka bentuk *suffix*-nya berupa : x y * 2 z 3 - * +.

Proses pengubahan ekspresi aritmatika x * y + 2 * (z – 3) ke dalam bentuk *suffix (postfix)* juga dilakukan berdasarkan urutan prioritas dari operator – operator yang terdapat di dalam ekspresi aritmatika tersebut. Proses dimulai dengan mengubah sub ekspresi (z – 3) menjadi z 3 - sehingga ekspresi aritmatika tersebut menjadi x * y + 2 * z 3 -. Proses dilanjutkan dengan mengubah sub ekspresi x * y menjadi x y * sehingga ekspresi aritmatika menjadi x y * + 2 * z 3 - dan dilanjutkan dengan mengubah bentuk 2 * z 3 - menjadi 2 z 3 - * sehingga ekspresi aritmatika menjadi x y * + 2 z 3 - *. Proses diakhiri dengan mengubah ekspresi aritmatika menjadi bentuk *suffix* x y * 2 z 3 - * +.

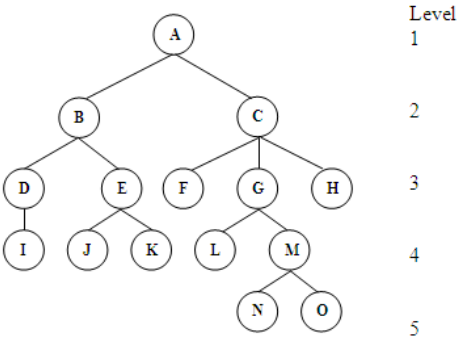
Pohon Ekspresi (*Expression Tree*)



1. Sebuah *node* tunggal adalah sebuah *tree*.
2. Jika terdapat sebuah *node* N dan beberapa *subtree* N_1, N_2, N_3, …, N_k maka dari *node* N dan *subtree* yang ada dapat dibentuk sebuah *tree* yang mempunyai *root* pada *node* N.



Gambar *Contoh pembentukan Tree*



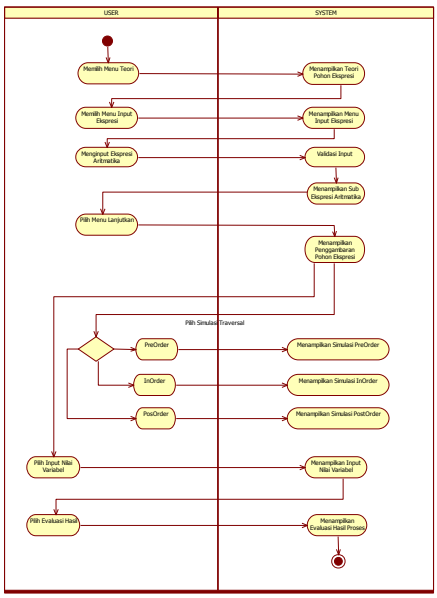
Gambar 2.12 *Contoh tree dengan 15 node*

ANALISIS SISTEM

Untuk mempermudah pembuatan aplikasi, maka diperlukan mekanisme pemecahan masalah menjadi bagian-bagian yang lebih kecil sehingga mempermudah dalam pengembangan dan pengecekan kesalahan.

Pada sub bab ini akan dimodelkan dari aliran – aliran kegiatan yang terjadi yang digambarkan dalam *activity diagram* dan secara garis besar adalah untuk memodelkan aliran kerja

(workflow) dan operasi dari perangkat lunak tersebut.

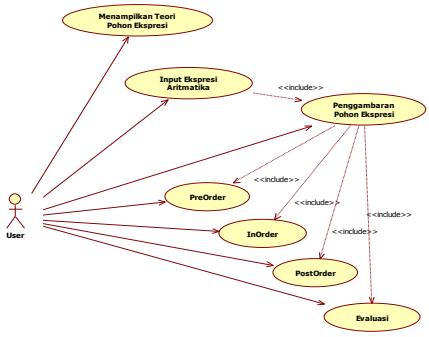


Gambar *Activity Diagram*

PERANCANGAN SISTEM

Use Case Diagram

Berikut ini akan dijelaskan proses-proses yang terjadi pada Perangkat Lunak Simulasi Pembelajaran Pohon Ekspresi (*Expression Tree*) yang dipetakan menggunakan *actor* dan *use case* dalam *diagram use case*.

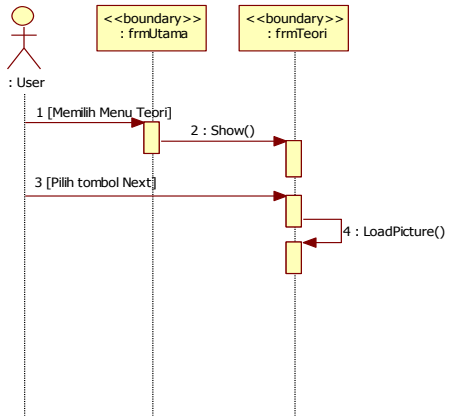


Gambar *System Use Case* Perangkat Lunak Simulasi Pembelajaran Pohon Ekspresi (*Expression Tree*)

Diagram Sekuensial

Diagram Sekuensial merupakan perincian dari *use case* diagram. Diagram sekuensial menggambarkan interaksi antar objek didalam dan diluar sistem. Pada sub bab ini akan menjelaskan sistem per case dalam diagram sekuensial.

Diagram Sekuensial Menampilkan Teori



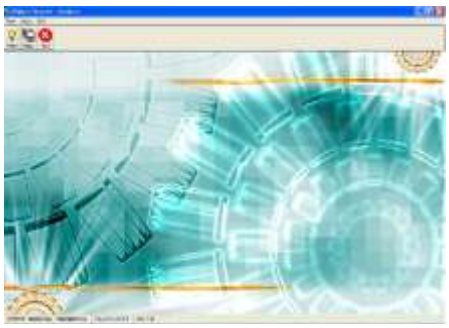
Gambar Diagram Sekuensial Menampilkan Teori

IMPLEMENTASI DAN UJI KUALITAS

Form Splash

Form ini berfungsi sebagai splash setelah program dijalankan dan sebelum tampil menu utama.

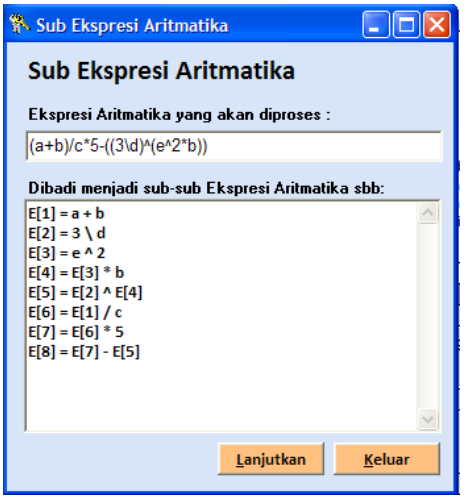
Gambar Tampilan Form Splash



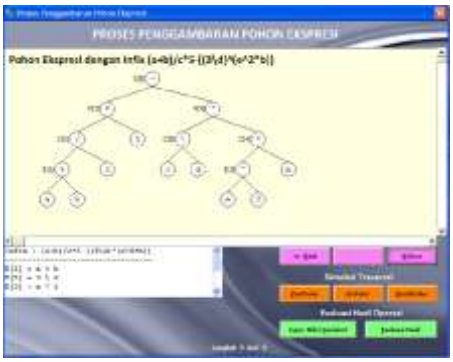
Gambar Tampilan Menu Utama

Gambar Tampilan Form Proses

Gambar Tampilan Form Input Token



Gambar Tampilan Form Sub Ekspresi Aritmatika



Uji Kualitas Perangkat Lunak

Tabel Tabel Pengujian Perangkat Lunak

ID_ Tes	Deskripsi Tes	Inisialisasi	Skenario	Hasil Yang Diharapkan	Hasil Pengujian
T_01	Tes Tombol Teori	Form menu utama sudah dibuka	Klik tombol Teori	Jika ditekan maka akan muncul form teori dan informasi	Sesuai
T_02	Tes Tombol Input	Form menu utama sudah dibuka	Klik tombol Input	Jika ditekan maka akan muncul form input ekspresi aritmatika	Sesuai
T_03	Tes Tombol Exit	Form menu utama sudah dibuka	Klik tombol Exit	Jika ditekan maka akan keluar dari aplikasi ini	Sesuai
T_04	Tes Tombol Buka	Form input Ekspresi Aritmatika sudah dibuka	Input file (.eks) dari folder penyimpanan	Input file (.eks) yang diinginkan akan masuk pada *TextBox Ekspresi*	Sesuai
T_05	Tes Tombol Input Token	Form input Ekspresi Aritmatika sudah dibuka	Input ekspresi aritmatika dengan menginput token pada form pilih	Setelah menginput token pada form pilih, jika menekan tombol oke maka input ekspresi aritmatika akan masuk pada *TextBox Ekspresi*	Sesuai
T_06	Tes Tombol Simpan (Input Ekspresi Aritmatika)	Form input Ekspresi Aritmatika sudah dibuka	Klik tombol Simpan	Jika ditekan maka akan keluar form tempat penyimpanan yang kita inginkan	Sesuai
T_07	Tes Tombol Proses	Form input Ekspresi Aritmatika sudah dibuka	Klik tombol Proses	Jika menekan tombol Proses maka akan muncul form Sub Ekspresi Aritmatika dari ekspresi yang telah diinputkan tadi	Sesuai
T_08	Tes Tombol Lanjutkan	Form Sub Ekspresi Aritmatika sudah dibuka	Klik tombol Lanjutkan	Jika ditekan tombol Lanjutkan maka akan tampil form Proses penggambaran pohon ekspresi	Sesuai
T_09	Tes Tombol Keluar (Sub Ekspresi Aritmatika)	Form Sub Ekspresi Aritmatika sudah dibuka	Klik tombol Keluar	Jika ditekan tombol Keluar maka akan kembali pada form Input Ekspresi Aritmatika	Sesuai
T_10	Tes Tombol Next	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol Next	Setiap menekan tombol Next yang aktif maka proses penggambaran pohon ekspresi akan maju satu langkah	Sesuai
T_11	Tes Tombol Back	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol Back	Setiap menekan tombol Back yang aktif maka proses penggambaran pohon ekspresi akan mundur satu langkah	Sesuai
T_12	Tes Tombol Keluar (Proses penggambar	Form Proses penggambaran pohon ekspresi	Klik tombol Keluar	Jika ditekan tombol Keluar maka akan kembali	Sesuai

	an pohon ekspresi)	sudah dibuka		pada form Input Ekspresi Aritmatika	
T_13	Tes Tombol PreOrder	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol PreOrder	Jika ditekan tombol PreOrder maka akan muncul input-box kecepatan simulasi PreOrder	Sesuai
T_14	Tes Tombol InOrder	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol InOrder	Jika ditekan tombol InOrder maka akan muncul input-box kecepatan simulasi InOrder	Sesuai
T_15	Tes Tombol PostOrder	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol PostOrder	Jika ditekan tombol PostOrder maka akan muncul input-box kecepatan simulasi PostOrder	Sesuai
T_16	Tes Tombol OK (input-box)	Form input-box kecepatan simulasi sudah dibuka	Klik tombol OK	Jika ditekan tombol OK maka simulasi traversal akan berjalan pada pohon ekspresi di form proses, dan hasil traversal akan ditampilkan pada form Hasil Trafersal	Sesuai
T_16	Tes Tombol Cancel (input-box)	Form input-box kecepatan simulasi sudah dibuka	Klik tombol Cancel	Jika ditekan tombol Cancel maka akan kembali pada form proses	Sesuai
T_17	Tes Tombol Input Nilai Variabel	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol Input Nilai Variabel	Jika ditekan tombol Input Nilai Variabel maka akan tampil form Input Nilai Variabel	Sesuai
T_18	Tes Tombol Simpan (Input Nilai Variabel)	Form Input Nilai Variabel sudah dibuka	Klik tombol Simpan	Jika ditekan tombol simpan maka input variable yang dimasukkan akan disimpan	Sesuai
T_19	Tes Tombol Keluar (Input Nilai Variabel)	Form Input Nilai Variabel sudah dibuka	Klik tombol Keluar	Jika ditekan tombol Keluar maka akan kembali pada form proses	Sesuai
T_20	Tes Tombol Evaluasi Hasil	Form Proses penggambaran pohon ekspresi sudah dibuka	Klik tombol Evaluasi Hasil	Jika diklik tombol Evaluasi Hasil maka akan muncul form evaluasi hasil dari perhitungan dengan input variable yang telah dimasukkan	Sesuai

DAFTAR PUSTAKA

Kurniadi, Adi. 2000. *Pemrograman Visual Basic 6*, Jakarta: PT. Elex Media Komputindo.

Kusrini, M.Kom. 2007. *Tuntunan Praktis Membangun Sistem Informasi Akuntansi Dengan Visual Basic & SQL Server*, Yogyakarta; Andi Yogyakarta.

Pamungkas, Ir. 2000. *Tip dan Trik Microsoft Visual Basic 6.0*, Jakarta:PT. Elex Media Komputindo.

Pressman, Roger S. Ph.D. 2002. *Rekayasa Perangkat Lunak*. Yogyakarta; Andi Yogyakarta.

http://cbs-bogor.net/ebooklain/Programing/modul%20vb_d3.pdf, 2011

http://elib.unikom.ac.id/files/disk1/451/jbptunikompp-gdl-audiairria-22549-2-unikom_a-i.pdf, 2011

http://dspace.widyatama.ac.id/xmlui/bitstream/handle/10364/583/bab3.pdf?sequen ce=5, 2011

http://www.pustakaskripsi.com/perancangan-perangkat-lunak-pembelajaran pohon-ekspresi-expression-tree-2094.html