Introdução ao Flex e Bison

Lucas Begnini Costa // André Moreira // George Lucas // Luis Sergio // Walderi

Universidade do Estado do Amazonas

May 24, 2013

Introdução ao Flex e Bison

Lucas Begnini Costa // André Moreira // George Lucas // Luis Sergio // Walderi

Universidade do Estado do Amazonas

May 24, 2013

Sumário

Introdução

Este trabalho tem como intuito apresentar a construção de uma calculadora que será utilizada através de linha de comando e interpretada através de:

- Flex - Analisador léxico

Mas para apresentar a calculadora será necessário uma breve apresentação a respeito de Flex e Bison

Introdução

Este trabalho tem como intuito apresentar a construção de uma calculadora que será utilizada através de linha de comando e interpretada através de:

- Flex - Analisador léxico
- Bison - Analisador semântico

Mas para apresentar a calculadora será necessário uma breve apresentação a respeito de Flex e Bison

FLEX

Como falado anteriormente o Flex é uma ferramenta para criação de analisadores Léxicos que procura padrões de escrita através de expressões regulares definidas dentro dos analisadores léxicos. Como resultado, depois da compilação do arquivo flex feito, é gerado um arquivo em C que valida todas as expressões regulares definidas no arquivo que passará pelo Flex

Instalando Flex no Linux

Antes de fazer qualquer coisa para criação do arquivo flex é necessário inicialmente a instalação do compilador flex no sistema operacional, no caso utilizado foi o Linux Ubuntu 13.04. Para instalação é necessário abrir o terminal e por linha de comando digitar:
$ sudo apt-get install flex

Criando um arquivo para o Flex

Para a criação de um arquivo para o flex não é necessário nenhuma extensão especifica, podendo ser o que o desenvolvedor quiser, mas como um ato de boa pratica de programação, padronizou-se como .lex ou .l para melhor distinguir de outros arquivos.

Criando um arquivo para o Flex

Dentro do programa há duas partes, separado por "%%" dividindo o programa em duas partes: A primeira para delimitação das expressões regulares que serão consideradas, e a segunda as ações que serão tomadas caso encontre as expressões regulares definidas anteriormente.Como podemos ver no exemplo seguinte: Exemplo de Flex

Criando um arquivo para o Flex

Para compilação de um arquivo flex é necessário 2 etapas:

- Compilação em flex primeiro do seguinte modo, pela linha de comando do linux:

Na primeira compilação ele pega arquivo de flex e gera um arquivo .C que vai validar as expressões regulares O parametro "-lfl" passado na compilação do gcc serve pra identificar que foi usado o flex pra gerar o .C

Criando um arquivo para o Flex

Para compilação de um arquivo flex é necessário 2 etapas:

- Compilação em flex primeiro do seguinte modo, pela linha de comando do linux:
- flex -o nome-do-programa.lex.c nome-do-programa.lex

Na primeira compilação ele pega arquivo de flex e gera um arquivo .C que vai validar as expressões regulares O parametro "-lfl" passado na compilação do gcc serve pra identificar que foi usado o flex pra gerar o .C

Criando um arquivo para o Flex

Para compilação de um arquivo flex é necessário 2 etapas:

- Compilação em flex primeiro do seguinte modo, pela linha de comando do linux:
- flex -o nome-do-programa.lex.c nome-do-programa.lex
- Em seguida é necessária a compilação em GCC com o seguinte comando

Na primeira compilação ele pega arquivo de flex e gera um arquivo .C que vai validar as expressões regulares O parametro "-lfl" passado na compilação do gcc serve pra identificar que foi usado o flex pra gerar o .C

Criando um arquivo para o Flex

Para compilação de um arquivo flex é necessário 2 etapas:

- Compilação em flex primeiro do seguinte modo, pela linha de comando do linux:
- flex -o nome-do-programa.lex.c nome-do-programa.lex
- Em seguida é necessária a compilação em GCC com o seguinte comando
- gcc -o nome-do-programa nome-do-programa.lex.c -lfl

Na primeira compilação ele pega arquivo de flex e gera um arquivo .C que vai validar as expressões regulares O parametro "-lfl" passado na compilação do gcc serve pra identificar que foi usado o flex pra gerar o .C

Bison

Bison é um gerador de interpretadores para fins gerais que converte uma gramática livre de contexto anotado em uma linguagem regular aplicando um parser nas tabelas do analisador lexico.

Instalando o Bison no Linux

Antes de fazer qualquer coisa para criação do arquivo Bison é necessário inicialmente a instalação do compilador Bison no sistema operacional, no caso utilizado foi o Linux Ubuntu 13.04. Para instalação é necessário abrir o terminal e por linha de comando digitar:
$ sudo apt-get install bison

Criando um arquivo para o Bison

- Compilação em Bison primeiro do seguinte modo, pela linha de comando do linux:

Criando um arquivo para o Bison

- Compilação em Bison primeiro do seguinte modo, pela linha de comando do linux:
- bison -d nome-do-programa.y

Criando um arquivo para o Bison

- Compilação em Bison primeiro do seguinte modo, pela linha de comando do linux:
- bison -d nome-do-programa.y
- Em seguida é necessária a compilação do arquivo lexico no caso o flex com o seguinte comando

Criando um arquivo para o Bison

- Compilação em Bison primeiro do seguinte modo, pela linha de comando do linux:
- bison -d nome-do-programa.y
- Em seguida é necessária a compilação do arquivo lexico no caso o flex com o seguinte comando
- flex -o nome-do-programa.lex.c nome-do-programa.lex

Criando um arquivo para o Bison

- Compilação em Bison primeiro do seguinte modo, pela linha de comando do linux:
- bison -d nome-do-programa.y
- Em seguida é necessária a compilação do arquivo lexico no caso o flex com o seguinte comando
- flex -o nome-do-programa.lex.c nome-do-programa.lex
- e em seguida a compilação do GCC com o seguinte comando

Criando um arquivo para o Bison

- Compilação em Bison primeiro do seguinte modo, pela linha de comando do linux:
- bison -d nome-do-programa.y
- Em seguida é necessária a compilação do arquivo lexico no caso o flex com o seguinte comando
- flex -o nome-do-programa.lex.c nome-do-programa.lex
- e em seguida a compilação do GCC com o seguinte comando
- gcc -o nome-do-programa nome-do-programa.lex.c nome-do-programa.tab.c -lfl -lm

Criando um arquivo para o Bison

Como é organizado o Bison e Bison vs YACC

Criando um arquivo para o Bison

Todos os arquivos de bison são terminados em .y

- Depois de definida a BNF o arquivo tem de ser complilado para gerar uma tabela de execucao

Criando um arquivo para o Bison

Todos os arquivos de bison são terminados em .y

- Depois de definida a BNF o arquivo tem de ser complilado para gerar uma tabela de execucao
- o arquivo gerado tem a extensao .tab.c para entao ser compilado no gcc

Calculadora

Como exemplo foi feito uma calculadora A calculadora e simples fazendo somente as operacoes basicas de soma, subtracao, multiplicacao e divisao Ela tambem pode fazer operacoes de potenciacao

Arquivo Flex da Calculadora

No arquivo flex da calculadora foi dividido em:

- * caracteres da linguagems

Arquivo Flex da Calculadora

No arquivo flex da calculadora foi dividido em:

- * caracteres da linguagems
- * significado dos lexemas

Arquivo Bison da Calculadora

No bison foi feito primeiro uma definicao dos tokens da linguagem
Depois a precedencia das operacoes indicando quais sao os nos que ficaram na raiz da arvore E tambem qual o metodo de derivacao da arvore
Logo em seguida e definida a BNF da calculadora

Árvore Sintática da Calculadora

*Arvore sintatica e uma abstracao de como e feita a verificacao da derivacao das expressoes de acordo com a BNF.
*A arvore sintatica cria todoas as possiveis combinacoes de expressoes que sao validas na linguagem

Árvore Sintática da Calculadora

EXEMPLIFICAR A ARVORE COM OS EXEMPLOS
1
1+3
1+3*4
1+3*4-2

Árvore Sintática da Calculadora

INPUT -¿ LINE -¿ EXPRESSION -¿ NUMBER -¿ 1
INPUT -¿ LINE -¿ EXPRESSION -¿ EXPRESSION + EXPRESSION -¿ NUMBER + EXPRESSION -¿ NUMBER + NUMBER -¿ 1 + 3

Árvore Sintática da Calculadora

EXEMPLIFICAR A ARVORE COM OS EXEMPLOS
(1+3)*4-2
(1+3)*(4-2)
(1+3)*4(-2)