UFRGS - Informática - INF01147 - Compiladores - 2020/1
Turma A - Prof. Marcelo Johann

Trabalho Prático - Instruções para Entrega da Etapa 1: Análise Léxica e Inicialização de Tabela de Símbolos

Resumo:

Este texto define as regras, formatos e procedimentos necessários para a entrega da primeira etapa do trabalho de desenvolvimento do compilador.

Definições neste texto:

Os tópicos tratados são os seguintes:

a. formato de entrega;
b. prazo;
c. organização do código;
d. recomendações;

Formato de Entrega

Cada aluno deve compactar o conteúdo do seu diretório de trabalho executando o comando "`tar cvzf etapa1.tgz .`" dentro do mesmo diretório. Não utilizem outros programas, formatos, comandos, nomes ou organizações de diretórios para esse operação. Será gerado um arquivo chamado `etapa1.tgz`, o qual deve ser copiado para o servidor *http* através do comando "`sftp username@html.inf.ufrgs.br`", deixando-o acessível no diretório `public_html` (Executem "`cd puclic_html`" e depois "`put etapa1.tgz`"). Vocês devem verificar as permissões de acesso ao arquivo, corrigi-las se necessário com o comando "`chmod 744`", e recomenda-se que testem o acesso à partir de um navegador.

O seu diretório de trabalho deve conter um arquivo `Makefile` com todos os comandos necessários à compilação do seu programa, automatizados através da chamada `make`, incluindo a invocação de `lex` ou `flex` e `gcc`. A execução de `make` deve gerar um executável no mesmo diretório de trabalho chamado apenas e exatamente "`etapa1`". O arquivo `Makefile` também deve conter uma definição `clean`, que será usada para apagar os arquivos intermediários e executável e permitir uma compilação nova completa.

Prazos

A etapa 1 do trabalho deve ser colocada no servidor *http* até as 12 horas do dia 26/08/2020, de acordo com o cronograma desse semestre, para que o trabalho seja avaliado de forma automática e pelo professor na parte da tarde dessa data.

Organização do Código

Existe uma certa flexibilidade na organização do código fonte, mas há uma série de regras que devem ser observadas para permitir os testes corretos.

a. Somente as ferramentas *make*, *lex* ou *flex* e *gcc* devem ser utilizadas;
b. código de *tokens.h* não deve ser modificado, caso contrário os valores retornados não poderão ser verificados;
c. Deve haver um arquivo *main.c* separado contendo unicamente a função *main* com as suas chamadas de teste para *yylex*, e nenhuma outra funcionalidade do seu analisador deve estar implementada nesse arquivo ou na *main*.
d. O código do seu arquivo *main.c* não pode incluir outros fontes seus com a diretiva `#include` para completar a compilação, pois ele será substituído. Você pode fazer o contrário, incluir *main.c* no restante do código, ou compilar os fontes separadamente para código objeto (`gcc -c`);
e. Você deve implementar uma função chamada `void initMe(void)` de onde todo e qualquer código de inicialização necessário (por exemplo, inicialização da tabela *hash*) deve ser chamado. Essa função será chamada pela *main* testadora substituída pelo professor. Você precisa implementar essa função mesmo que encontre outra forma de inicialização que não dependa dela, pois ela será chamada pelos testes automatizados.
f. Não use nenhuma estrutura hierárquica de diretórios. Todos os seus arquivos, incluindo fontes, teste, *Makefile*, executável, devem estar no mesmo diretório de trabalho. Eles serão compactados pelo seu comando *tar* e descompactados por um comando correspondente dentro de um único diretório onde serão testados.

Recomendações

Executem vários testes. Verifiquem a conformação com cada uma das regras desse formato e da especificação da etapa. Verifiquem se o trabalho pode ser compilado e rodado em outro sistema fora o usado para desenvolvimento. Verifiquem o acesso ao arquivo disponibilizado no servidor *http*. Incluam os nomes dos componentes do grupo em comentários no início de todos os códigos fonte do seu trabalho. Retirem dúvidas com o professor antes do prazo final.

Porto Alegre, 19 de Agosto de 2020