Objetivo

O objetivo do projeto deste semestre será a consolidação dos conceitos de verificação de tipos e geração de código vistos ao longo da disciplina. Por simplicidade as duas entregas poderão ser executadas em separado, conforme discutido nas aulas.

Execução

- O trabalho pode ser realizado individualmente ou em grupos de até 3 alunos
- Trabalhos que não compilarem/executarem serão desconsiderados
- As entregas de trabalhos são feitas exclusivamente pelo envio de arquivo zipado (zip ou rar) na sala de entrega disponibilizada no sistema moodle. Não serão aceitas quaisquer outras formas de entrega
- Os trabalhos que configurarem algum tipo de fraude (cópia) serão anulados e atribuída a nota zero a todos os alunos envolvidos
- quais quer questões, sobre a execução ou entendimento do trabalho, devem ser submetidas (e serão acordadas) no forum criado para discussão do trabalho prático. Não serão respondidas questões sobre o trabalho por mail.

Entrega (conteúdo do arquivo postado no moodle)

1. Especificações JFlex e (eventualmente) ByaccJ implementadas e todos os arquivos fontes necessários à compilação e execução do projeto.
   Obs. o grupo pode optar por utilizar a versão C++, ou C, dos geradores
2. "*Programas teste*" que demonstrem o que foi implementado
3. ***Todos*** os arquivos devem estar claramente identificados com o nome, numero de matrícula e e-mail de cada um dos componentes do grupo.
4. Relatório, também identificado, contendo instruções de compilação, plataforma utilizada para o desenvolvimento do trabalho, itens da descrição que não foram implementados e o porquê, e se for o caso, o que foi implementado além do que foi solicitado. Este relatório deverá estar em formato texto ou html.

Data de entrega e apresentação

- Dia 30 de novembro (até as 08h), conforme cronograma da disciplina

Parte 1 - **Verificação de tipos** (55% da nota do TF)

A partir do exemplo de verificação semântica trabalhada em aula, acrescentar regras semânticas que construam um sistema de tipos capaz de permitir a declaração de funções e *structs*. Realizar as verificações semânticas básicas segundo as regras da linguagem ANSI C. Minimamente as seguintes verificações devem ser realizadas:

- todo identificador deve ser declarado antes do seu uso (dentro do escopo corrente)

- não é possível re-declarar um identificador (dentro de um mesmo escopo)
- os tipos válidos da linguagem são inteiro (“integer”), ponto flutuante (“double”); lógico (“boolean”) e literais (“strings”). Como tipo de retorno de funções acrescenta-se o tipo “void”;
- na ativação de subprogramas (funções), o número de parâmetros deve ser igual, os tipos devem ser compatíveis aos tipos da declaração do subprograma e o tipo de retorno compatível com a expressão/atribuição onde estiver sendo utilizada.
- Não é possível declarar *structs* “aninhadas” mas deve ser possível definir variáveis do tipo *struct* como campos de outra *struct*, bem como realizar a validação de tipos desses casos.

No caso de dúvidas sobre a sintaxe/semântica das funções/structs utilizar o padrão da linguagem “C”, ou postar no fórum de dúvidas da disciplina para ficar acordado com o grupo.

Parte 2 **- Geração de código** (55% da nota do TF)*

A partir do exemplo de geração de código (especificação ByaccJ) disponibilizado na página da disciplina, complementar esta especificação com as seguintes estruturas (com a respectiva geração de código). Note que os primeiros 3 item são obrigatórios (e relativamente triviais) e obrigatórios para poder testar os casos disponibilizados para o comando “for”.

1. transformar o “comando” de atribuição em uma “expressão” de atribuição
2. acrescentar operadores de pré e pós-incremento e atribuição por adição (+=)
3. acrescentar os comandos “break” e “continue”
4. acrescentar o comando “for”
5. Acrescentar geração de código para o uso de “arrays” (apenas uma dimensão)

No caso de dúvidas sobre a sintaxe/semântica de execução das estruturas utilizar o padrão da linguagem “C”, ou postar no fórum de dúvidas da disciplina para ficar acordado com o grupo.

* Sim, o trabalho tem 110% de nota.