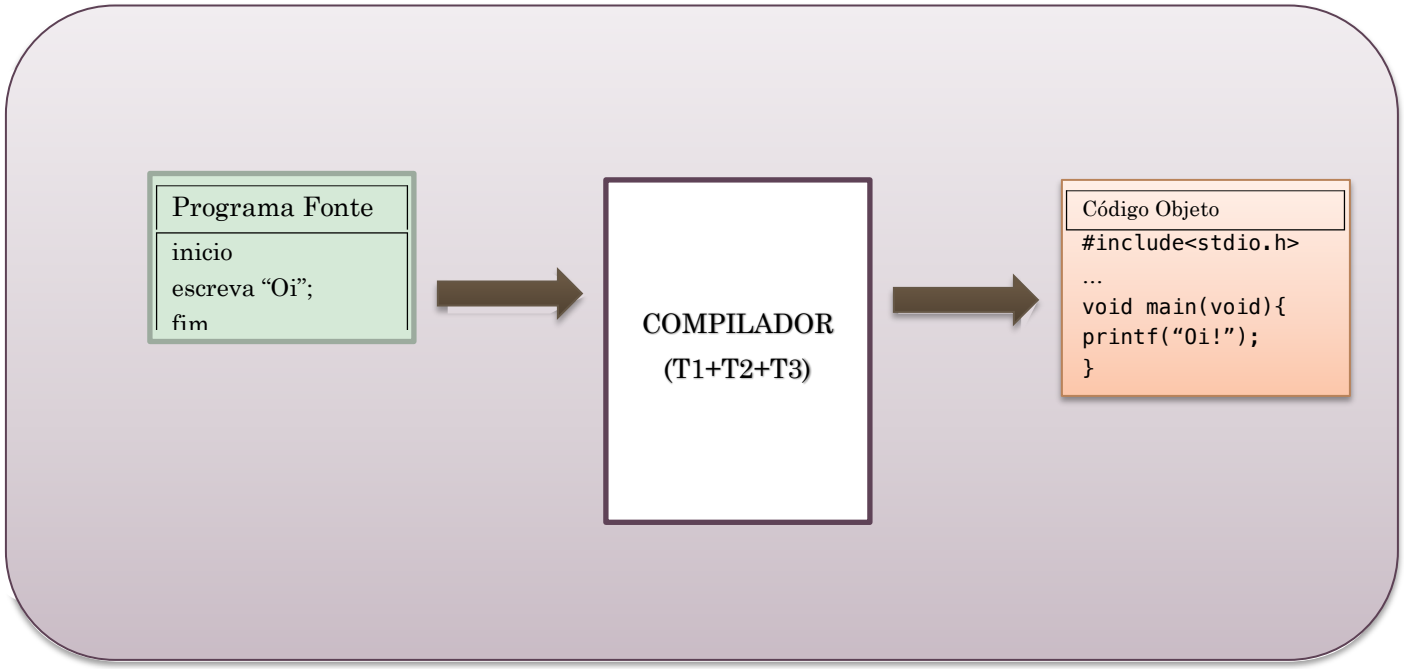
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Trabalho PRÁTICA - Regras Gerais para T1, T2 e T3

PROJETO: Desenvolvimento de um compilador



Profa. Dra. Deborah Fernandes

1 . Descrição

A atividade prática, estudo de caso dividido em três etapas T1 (Implementação do Trabalho 1), T2 (Implementação do Trabalho 2) e T3 (Implementação do Trabalho 3) em Compiladores, é um componente para a avaliação e desenvolvimento dos conhecimentos desenvolvidos nas disciplinas ofertadas para Ciência da Computação e Engenharia de Computação - Compiladores e Compiladores 1. As notas de cada trabalho, avaliação oral e entregas de complementares estão descritas no Plano de curso da disciplina disponível na plataforma Turing.

A disciplina de compiladores preocupa-se em estudar técnicas e teorias com a finalidade de proporcionar o conhecimento para a construção de um compilador. Para tal, durante o semestre investigar-se-á seus componentes sobre aspectos teóricos e práticos em um estudo de caso. Esse estudo envolverá o desenvolvimento (implementação) de um compilador que receberá como entrada um arquivo fonte na linguagem de programação *Mgol* (linguagem desenvolvida para o estudo de caso em questão), realizará as fases de análise e síntese (T1,T2 e T3) e gerará um arquivo objeto em linguagem C. O arquivo final deverá ser compilável em compilador C, ou seja, o código gerado deverá estar completo para compilação e execução.

A Figura 1 apresenta o modelo de arquitetura do compilador que será desenvolvido durante o semestre. Os módulos a serem implementados contemplam:

- T1 – Implementação do Trabalho 1: desenvolvimento do analisador léxico e da tabela de símbolos;
- T2 - Implementação do Trabalho 2: desenvolvimento do analisador sintático ascendente SLR(1) para verificação de sintaxe com dados obtidos do analisador léxico (T1) e também a recuperação do erro com reestabelecimento da análise;
- T3 – Implementação do Trabalho 3: desenvolvimento do analisador semântico e geração de código final a partir do método tradução dirigida pela sintaxe (conexão com T2).

2. Regras para o desenvolvimento

2.1. **SOBRE o código:**

- O trabalho (códigos fonte e executáveis) será entregue EXCLUSIVAMENTE via *Plataforma Turing* na data definida pelo professor. Para cada dia de atraso serão descontados 0,3 (por dia) até o dia da avaliação.
- O(s) aluno (s) poderá(ão) escolher a linguagem de programação que será utilizada para desenvolver o trabalho e deverão permanecer no uso desta linguagem até a finalização do projeto.

- No dia e horário destinado a avaliação, o aluno deverá baixar o código que foi depositado na Turing e este será utilizado para a arguição.
- A arguição será realizada sobre o código depositado na Plataforma TURING.
- O código depositado na Turing a ser avaliado em sala junto ao professor não poderá conter comentários. Se necessário, crie um manual de instruções que não poderá ser utilizado na avaliação. Este será utilizado pela equipe, somente.
- NÃO SERÁ PERMITIDO o uso de geradores de analisadores léxicos, sintáticos ou *Regex* para solucionar o problema proposto em cada especificação de trabalho.
- Cópias de trabalhos (códigos) de colegas ou de semestres anteriores terão nota 0,0.
- Somente serão avaliados os códigos de programas que estejam executando e com as principais funcionalidades implementadas e funcionand**o**.

2.2. **Sobre horário e local da avaliação:**

- As avaliações serão realizadas com horário marcado (dentro dos horários de aula regulares da disciplina ou em outro acordado com o(s) aluno(s)) e comunicados aos alunos com antecedência de até 2 dias antes da apresentação.
- As avaliações serão realizadas em sala de laboratório (a ser confirmada pela professora), poderão também ser realizadas na sala de aula ou sala do professor.
- Os alunos deverão comparecer com 10 minutos de antecedência ao horário agendado.

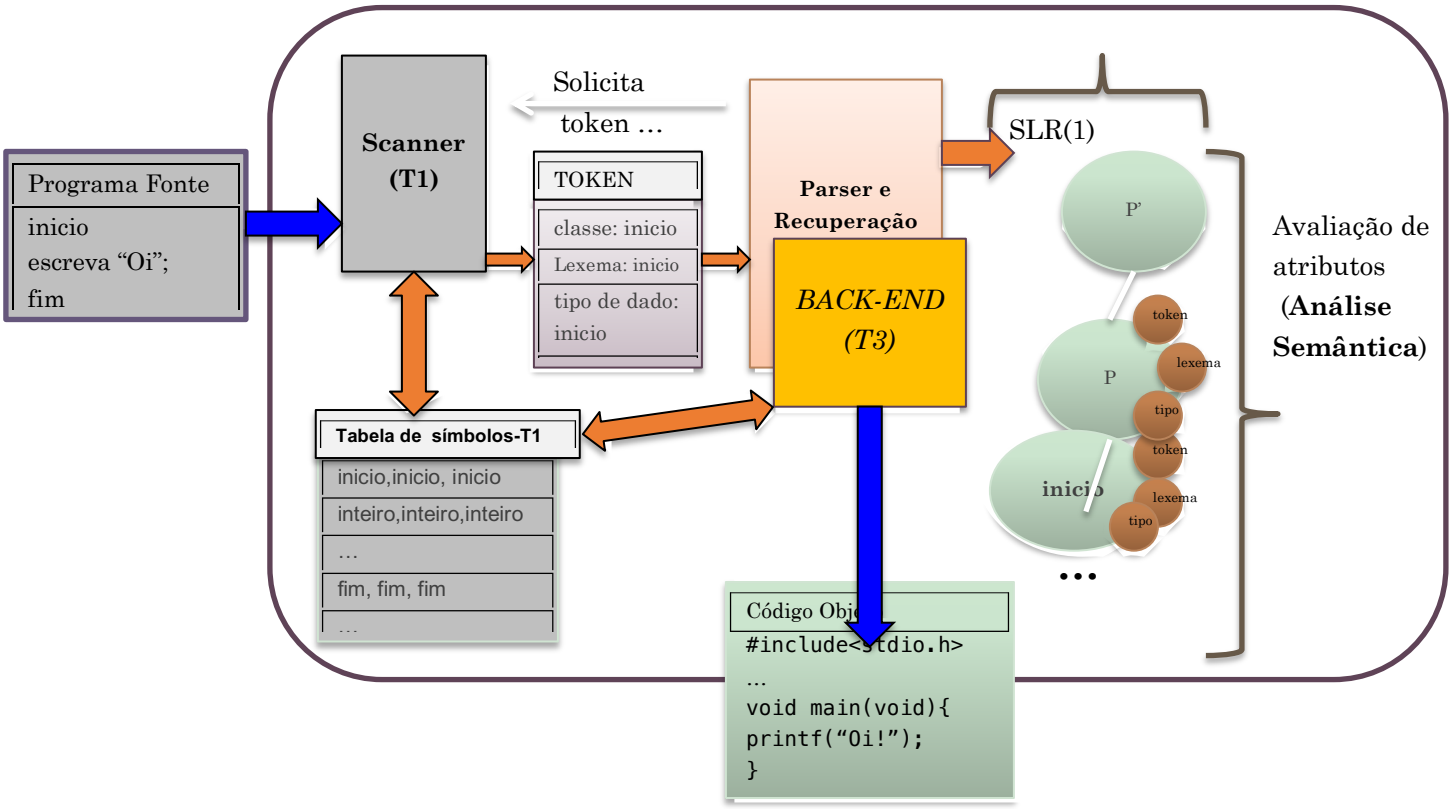
2.3. Sobre o formato da avaliação:

- Arguição oral: O professor arguirá o aluno ou escolherá qualquer aluno componente da equipe para responder questões sobre o desenvolvimento do trabalho, implementação baseada nas descrições dos trabalhos T1, T2 e T3. Não será realizada na forma de seminário.
- Em caso de equipe, TODOS deverão estar presentes no horário da avaliação.
  - A nota será a mesma para TODOS os alunos.
  - Se um aluno não souber explicar quaisquer questões sobre o trabalho ao ser arguido, em caso de equipe, a nota será a mesma para TODOS.

2.4. Sobre dúvidas:

- O aluno poderá tirar suas dúvidas sobre o trabalho em horários de atendimento extraclasse, em horários determinados pelo professor em sala, através de e-mail ou fórum e chat da disciplina até a data determinada para a entrega.

Figura 1 – Arquitetura Geral do Compilador Completo (T1+T2+T3).



3 - Entregáveis

1 – **Atividades complementares aos trabalhos**: Tarefa INDIVIDUAL. Entregar na data determinada pelo professor, EXCLUSIVAMENTE via plataforma Turing**. A entrega atrasada dessas atividades não contabilizará nota.** Atividades:

- T1.1 - Atividade 1 complementar ao trabalho 1 (T1) – Autômato finito determinístico e questões sobre a análise léxica.
- T2.1 - Atividade 1 complementar ao trabalho 2 (T2) – Conjuntos First/Follow e tratamento de erro.
- T2.2 - Atividade 2 complementar ao trabalho 2 (T2) – Autômato LR(0).

- T3.1- Atividade 1 Complementar ao trabalho 3 (T3) – Regras semânticas adicionais e tradução dirigida pela sintaxe.

2 – **Implementação – EQUIPES- Entrega de código** – Os códigos para T1, T2 e T3 deverão ser entregues EXCLUSIVAMENTE via plataforma Turing, na data determinada pelo professor. Caso seja realizado em equipe, apenas um componente deverá depositá-lo na plataforma. Se forem entregues mais de um arquivo, utilizar a compactação .zip. O NOME do código deverá seguir o padrão:

- Para o T1: ***T1-NomeAluno1-NomeAluno2-20232.extensão***.
- Para o T2: ***T2-NomeAluno1-NomeAluno2-20232.extensão***.
- Para o T3: ***T3-NomeAluno1-NomeAluno2-20232.extensão***.

Exemplo para duplas: T1-DeborahFernandes-FulanoPrado20232.zip

Exemplo para trabalho individual: T1-DeborahFernandes20232.zip

4 – O que fazer?

A cada liberação de descrição de trabalho T1, T2 e T3:

- Ler a descrição;
- Atentar e compreender as solicitações;
- Tirar dúvidas com o professor e ou monitor (e-mail, chat, fórum, atendimento extraclasse);
- Implementar cada item elencado, exatamente como solicitado;
- Compreender a teoria que está sendo implementada na prática;
- Produzir um código que possa ser executado e que atenda às principais funcionalidades (mínimo para ser avaliado).

5 – Resultado final

A cada edição do trabalho teremos um resultado final esperado conforme abaixo:

- T1: a leitura do arquivo fonte e produção de tokens conforme definição do T1 para a análise léxica e tabela de símbolos.
- T2: Obtenção dos tokens do trabalho T1, produção da árvore sintática através do modelo de análise sintática definido na descrição do trabalho 2 e implementação de rotina de tratamento e recuperação do erro sintático.

- T3: Realização de análise semântica e produção de código final em conjunto com a análise sintática implementada no trabalho 2.

Ao final de todos os três trabalhos práticos da disciplina (modelo exposto de maneira simplificada através da Figura 1), teremos como sistema e resultado do estudo de caso, um pequeno compilador que compilará o programa fonte (linguagem Mgol), Fonte.ALG na Figura 2 (a) em PROGRAMA.C da Figura 2(b).

Figura 2 – Entrada e saída do Compilador Completo (T1+T2+T3).

(a) Fonte. alg

```
inicio
   varinicio
      literal A;
      inteiro B, D;
      real C;
   varfim;
   escreva "Digite B:";
   leia B;
   escreva "Digite A:";
   leia A;
   se(B>2)
   entao
      se(B<=4)
      entao
         escreva "B esta entre 2 e 4";
      fimse
   fimse
   B<-B+1;
   B<-B+2;
   B<-B+3;
   D<-B;
   C<-5.0;
   escreva "\nB=\n";
   escreva D;
   escreva "\n";
   escreva C;
   escreva "\n";
   escreva A;

fim
```

(b) Programa.c

```
include<stdio.h>
typedef char literal[256];
void main(void)
{       /*----Variaveis temporarias----*/
        int T0;
        int T1;
        int T2;
        int T3;
        int T4;
        /*------------------------------*/
        literal A, E;
        int B;
        int D;
        double C;
        printf("Digite B");
        scanf("%d",&B);
        printf("Digite A:");
        scanf("%s",A);
        T0=B>2;
        if(T0)
        {
           T1=B<=4;
           if(T1)
           { printf("B esta entre 2 e 4");
           }
        }
        T2=B+1;
        B=T2;
        T3=B+2;
        B=T3;
        T4=B+3;
        B=T4;
        D=B;
        C=5.0;
        printf("\nB=\n");
        printf("%d",D);
        printf("\n");
        printf("%lf",C);
        printf("\n");
        printf("%s",A);
}
```