Trabalhos de Compiladores

Os trabalhos devem ser desenvolvidos prioritariamente em dupla ou trio.
Todos deverão ser apresentados no horário de aula.
Os trabalhos poderão ser apresentados com uma semana de atraso (exceto o último), porém nesta situação receberão nota máxima de 7.0. O último trabalho somente poderá ser entregue na data definida.

Trabalho 1 (T1) – Analisador Léxico (Peso: 30% da M1)
Apresentação em: 20/08/2025

Requisitos

Deverão ser construídas expressões regulares para identificação dos tokens de uma linguagem de programação. Deverá ser utilizada a ferramenta GALS, a qual será utilizada também para apresentação.
Os tokens a serem reconhecidos deverão ser, no mínimo:

- (0.5 ponto) Palavras Reservadas (se, entao, senao, end, int, float, ...)
- (1 ponto) Operadores Aritméticos (+, - , ...)
- (1 ponto) Operadores Relacionais (>, <, >=, <=, ==, !=)
- (1 ponto) Operadores Lógicos (&&, ||, !)
- (1 ponto) Operadores Bit a Bit (>>, <<, &, |, ~, ^)
- (1 ponto) Identificadores (nomes de variáveis, sub-rotinas e programas): **Deverão** possibilitar nomes que contenham números e o caractere "_".
- (1 ponto) Literais: Valores constantes do tipo Inteiro (Decimal, Binário e Hexadecimal), Real, Caractere e String entre aspas (simples para caractere e dupla para string)
- (1 ponto) Comentários de uma linha (//)
- (2 ponto) Comentário de múltiplas linhas (**DEVE ser este:** /*... */, depois da avaliação pode ser alterado)
- (0.5 ponto) Delimitadores e pontuadores (ponto, ponto e vírgula, dois pontos, colchetes, etc.)

Trabalho 2 (T2) – IDE e Analisador Sintático (Peso: 70% da M1)
Apresentação em: 17/09/2025

Requisitos

(1 ponto) Deverá ser construído um editor de programas integrado ao Compilador, uma IDE – Integrated Development Environment que possibilite:

- Digitar um programa (código fonte) – **a fonte deve ter tamanho 14**;
- Realizar a **Análise Sintática** do programa (Compilar);
- Exibir mensagens de erro e depuração – **fonte tamanho 14**.

O analisador sintático deverá ser construído utilizando a ferramenta GALS para definição da Gramática da Linguagem e deverá conter as seguintes instruções (no mínimo):

- (1,5 ponto) Declaração de variáveis e vetores c/ suporte a declaração de múltiplos nomes (múltiplos ids, 2 ou mais variáveis e vetores)
- (1 ponto) Blocos de instruções com início e fim (escopos)
- (2 pontos) Três tipos de Laços de Repetição (pré-testado, pré-testado com variáveis de controle [inicialização e pós-operação], e pós-testado)
- (0,5 pontos) Desvio condicional simples (sem else)
- (1 pontos) Desvio condicional composto (com else)
- (0,5 pontos) Entrada de dados (leia, cin,...) com variáveis e vetores;
- (0,5 pontos) Saída de dados (escreva, cout, System.out.print) com variáveis, vetores e literais;
- (1,5 pontos) Atribuição com variáveis, vetores recebendo expressões com variáveis, vetores, literais e chamadas de sub-rotinas;
- (1,5 pontos) Definição de sub-rotinas (procedimentos e funções) com passagem de parâmetros (com suporte a variáveis, vetores e literais) e chamada de sub-rotinas como operação;
- Expressões aritméticas, relacionais e lógicas sem restrição de tamanho e com possibilidade de uso de parênteses (pegar pronto no material e acrescentar suporte a variáveis, vetores e sub-rotinas)

Trabalho 3 (T3) – Análise Semântica - Tabela de Símbolos, Escopos e Compatibilidade de tipos (Peso: 50% da M2)
Apresentação em: 22/10/2025

Requisitos

Deverão ser acrescentadas ações semânticas a gramática, e estas deverão ser implementadas para possibilitar a realização da gerência da tabela de símbolos e demais verificações semânticas. Os seguintes quesitos serão avaliados:

OBS: Identificadores são: variáveis, vetores e funções.

- (2 pontos) Inserir identificadores na tabela de símbolos com os respectivos tipo, modalidade (variável, vetor, parâmetro ou função) e escopo;
- (2 pontos) Verificar se um identificador está declarado no escopo em que é usado no programa;
- (1 pontos) Garantir a unicidade dos identificadores em um escopo;
- (1 ponto) Avisar se os identificadores são declarados e não usados;
- (1 ponto) Avisar (**não restringir**, apenas avisar) se os identificadores estão sendo usados sem estar inicializados (uso de lixo de memória);
- (1 ponto) Acrescentar na interface da IDE um componente para visualização da tabela;
- (2 pontos) Verificar a compatibilidade de tipos nas expressões (por exemplo, evitar multiplicação de strings) e nas atribuições.

Trabalho 4 (T4) – Geração de Código I (Peso: 50% da M2)
Apresentação em: 12/11/2025

Requisitos

- Realizar a geração de código do programa usando o conjunto de instruções do processador BIP. O Código gerado deve executar corretamente no Bipide 3.0;
- As seguintes instruções devem ter seu código gerado:

- (1 ponto) Declarações de variáveis e vetores (.data)
- (1 ponto) Entrada de dados (com variáveis e vetores)
- (1 ponto) Saída de dados (com inteiros, variáveis e vetores)
- (3 ponto) Atribuições para variáveis e vetores
- (3 ponto) Operações aritméticas (soma e subtração e operações bit a bit com variáveis e vetores)
- (1 ponto) Deverá haver na interface do editor um local para exibição do código assembly gerado (asm);

Trabalho 5 (T5) – Geração de Código II (Peso: 60% da M3)
Apresentação em: 03/12/2025

Requisitos

- Realizar a geração de código do programa usando o conjunto de instruções do processador BIP;
- As seguintes instruções devem ter seu código gerado:

- (1 ponto) Suporte a operações relacionais
- (1 ponto) Desvio condicional simples (sem else)
- (1,5 ponto) Desvio condicional composto (com else)
- (1 ponto) Laço de repetição com teste lógico no inicio (while)
- (1 ponto) Laço de repetição com teste lógico no fim (do..while)
- (2,5 ponto) Laço de repetição com variável de controle (for)
- (2 pontos) Suporte para aninhamentos (ex: laço dentro de laço, desvio dentro de laço, etc...);

Trabalho 6 (T6) – Geração de Código III (Peso: 40% da M3)
Apresentação em: 10/12/2025

Requisitos

- Realizar a geração de código do programa usando o conjunto de instruções do processador BIP;
- As seguintes instruções devem ter seu código gerado:

- (1 ponto) Chamada de sub-rotinas (gestão de rótulos e calls)
- (3 pontos) Passagem de parâmetros (por cópia)
- (2 pontos) Retorno de funções (permitir em expressões e atribuições)
- (3 pontos) Compatibilidade entre parâmetros (tipo, ordem e quantidade)
- (1 pontos) Gerar mensagens de aviso e erro significativas para o usuário (ex.: (*i*) a função "teste" esperava 2 parâmetros e foram passado X parâmetros; (*ii*) a rotina "coisa" não existe,...)