



TRABALHO I
Compiladores
Prof. Lucas Ismaily

INFORMAÇÕES IMPORTANTES

O Trabalho I contém três opções para escolher. As opções são mutuamente exclusivas, isto é, somente é possível escolher **exatamente uma opção**. A data máxima de entrega do trabalho é **21/05/2025**. Porém, recomendo fortemente que entreguem antes, para evitar imprevistos. **Atenção:** findado o prazo de envio, todos os grupos que não enviaram receberão automaticamente nota **zero**. A entrega será **somente** via e-mail (ismailybf@ufc.br, assunto: Trabalho I - Compiladores), numa pasta zipada contendo todos os arquivos, e se preciso, instrução para execução. Também deve conter um arquivo contendo todos os nomes do membros da equipe.

Trabalho pode conter no mínimo 3 e máximo 5 alunos. Sejam honestos com vocês e comigo. Qualquer fraude será punida com nota zero para todos os envolvidos.

PARA AS OPÇÕES 1 E 3 DO TRABALHO CONSIDERE A LINGUAGEM LangB

A **LangB** é uma linguagem minimalista e estruturada inspirada na **Linguagem C**, com regras claras e sintaxe simplificada para facilitar o aprendizado e a implementação de compiladores. O alfabeto da linguagem é formado por números, letras do alfabeto português (sem acentuação), os caracteres reservados da linguagem além dos símbolos: ‘!,@,#,\$,%,&,?,/, _ e |’.

Características da LangB:

1. Tipos de Dados

- Apenas dois tipos são suportados:
 - num (equivalente a int para números inteiros).
 - text (equivalente a string para cadeias de caracteres).
 - true.
 - false.

2. Operadores Suportados

- Operações matemáticas básicas: +, -, *, /.
- Operadores de comparação: >, <.
- Operador de atribuição: =.



- **Não** há operadores bit a bit.

3. Estrutura do Código

- Cada instrução deve terminar com ;.
- Declaração de variáveis segue o formato:

```
num x = 10;  
text y = "hello";
```

- Nome de variáveis devem ter tamanho de no máximo 30 caracteres e **não podem** iniciar com seguintes caracteres: !, @, #, \$, %, &, ?, /, |.
- **Não há funções, blocos de código ou estruturas de controle** como loops e condicionais.

4. Entrada e Saída

- A única forma de saída é a instrução show, que exibe o valor de uma variável.

```
show x;  
show y;
```

- Não há entrada de dados ou interação com o usuário.

5. Ausências na Linguagem

- Não há estruturas de repetição (for, while).
- Não há condicionais (if, else).
- Não há funções ou modularização.
- Não há suporte para bibliotecas externas ou importação de código.
- Não há ponteiros ou gerenciamento avançado de memória.

Exemplo de Código em LangB

```
show 2>2  
num a = 5;  
text mensagem = "Oi!";  
show mensagem;  
show a;
```

Saída esperada:

```
false  
Oi  
5
```

Essa linguagem mantém a estrutura do C, mas com palavras-chave simplificadas para tornar sua



compreensão mais acessível.

Opção 1 – Análise Léxica

1. (2,0 pontos) Crie *Tokens* apropriados e para cada *Token* faça uma Expressão Regular para a Linguagem **LangB**.
2. (2,0 pontos) Implemente um algoritmo que recebe como entrada todas as Expressões Regulares da Questão anterior e retorna um único Autômato Finito Não-Determinístico (NFA).
3. (3,0 pontos) Implemente um algoritmo que recebe como entrada um Autômato Finito Não-Determinístico (NFA) e retorna um Autômato Finito Determinístico (DFA). A forma de representação dos Autômatos é livre, ou seja, você pode representá-los como matriz, lista, dicionário etc.
4. (3,0 pontos) Utilizando o DFA da Questão 3, implemente um analisador léxico para a Linguagem **LangB**. Além do código, é preciso entregar um arquivo .txt contendo a lista de *tokens* utilizados e o que eles representam. O arquivo tem o seguinte formato: cada linha contém duas informações separadas por espaço, sendo a primeira posição o *token* e a segunda o que ele representa. Se o *token* representa mais de uma entidade, separe-os por vírgula.

Entrada

A entrada é composta por um código fonte de um programa qualquer escrito em **LangB**.

Saída

Para cada entrada, seu programa deve produzir uma sequência de *Tokens* ou a palavra ERRO, caso a entrada tenha erro léxico.

Exemplo

Entrada

```
int a = 0 ;  
int b = 5 + a ;  
text c = "teSte" ;
```

Saída

```
INT VAR EQ NUM SEMICOLON  
INT VAR EQ NUM ADD VAR SEMICOLON  
TEXT VAR EQ CONST SEMICOLON
```