

APS LOGCOMP

Orientador: Raul Ikeda

Aluno: Fernanda Pereira



Motivação da APS

- Criar uma linguagem simples, expressiva e autoral
- Evitar a mesmice de copiar sintaxes como C/Java
- Pensar em comandos que facilitam a leitura e a intenção do código
- Tornar decisões e laços mais declarativos

Características da Linguagem

- Tipagem explícita (`int`, `text`, `bool`)
- Declarações com valor obrigatório: `let x: int := 10;`
- Comando `decide(cond, A, B)` para decisões como expressão
 - Laços com `loop i in range(a, b) { ... }`
 - Impressão com `show(...)`
 - `watch x;` para debugar valor de variáveis
 - `yield` para indicar um retorno intermediário

Diferente de
um `if`
clássico,
`decide`
funciona
como uma
****expressão****.

```
loop_statement =  
    "loop" identifier "in" "range" "("  
        expression "," expression ")" " {"  
            statement } "};
```

```
if_expression =  
    "decide" "(" condition ","  
        expression_if_true ","  
        expression_if_false ")";  
    ...
```

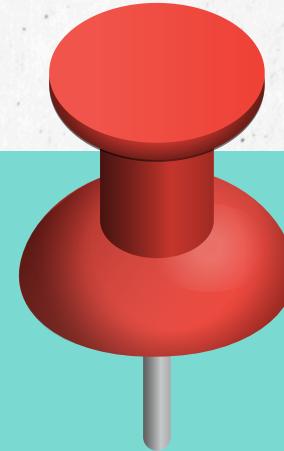


Exemplo de Código

```
let nome: text := "Fernanda";
show("Bem-vinda, " ++ nome);

let pontos: int := 15;
watch pontos;

let status: int := decide(pontos >= 10, 1, 0);
yield status;
```



Resposta no Terminal:

```
Bem-vinda, Fernanda
[watch] pontos = 15
[yield] 1
```

Conclusão

- Linguagem 100% autoral e funcional
- Aprendi a estruturar EBNF, criar lexer/parser e interpretar AST
- Flex + Bison + C = muito poder, mas também muita atenção
- Próximos passos: interpretar `bool` e strings reais.