KUALA

ATHENA SARANTÔPOULOS (2652)
JOSE GRIGORIO NETO (3046)
THIAGO OLIVEIRA (3037)
VITOR HUGO (3049)

Sumário

Introdução

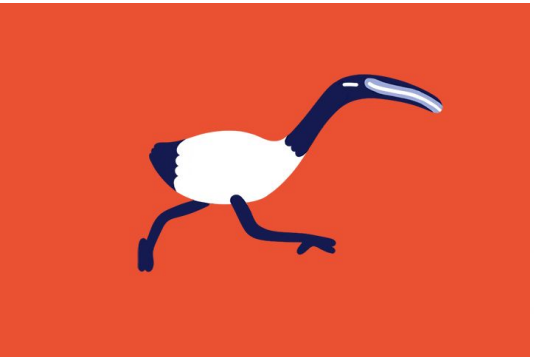
- Criar uma linguagem;
- Criar a gramática;
- Criar o compilador.

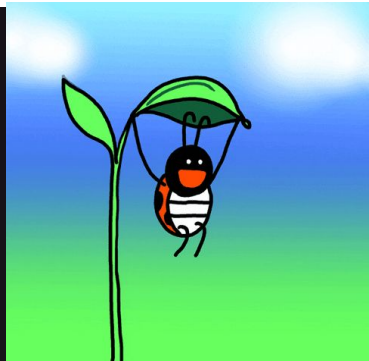
Sobre a Linguagem

- Kuala;
- Origem por ser composta por nomes de animais;
- Inspirada em C;

Dados primitivos

- int = ibis;
- float = frog;
- string = snake;
- bool = bug;





Comandos

- if = iguana;
- for = fox;
- while = whale;
- else = eel.



Palavras reservadas

- iguana;
- fox;
- whale;
- eel;

- ibis;
- viper;
- snake;
- frog;

- bug;
- kuala;
- rabbit;
- ʕ •ᴥ•ʔ .

Exemplo de código

```
kuala(){
    //kuala hj amanha e sempre

    ibis int = 0;
    ibis array = [0,2,3,4]
    frog float = 1.0;
    bug bool = 1;
    snake string = 'desenho do koala*';

    func(string)


    fox(ibis i=0;i<=10;i++){
        iguana(i<=2){
```

Exemplo de código

```
15              print(i+i);
16          }eel iguana(i<=5){
17              print(i-i);
18          }eel iguana(i<=8){
19              print(i*i);
20          }eel{
21              print(i/i);
22          }
23
24          iguana( 2 == i && i+2 == 4){
25              print("Logic yeahh!!!");
26          }
27
28          iguana(3==i || i+1==3){
29              print('Logic !!hhaey');
30          }
31
32          whale(i==2){
33              print("OooOoOOOOOOoooo0000");
34          }
35
36      }
37
38  }
```

Exemplo de código

```
39
40  viper func(snake str){
41      print("THIS IS A FUNCTION " + str);
42  }
43
44  /*
45
46      __           __
47   /"  "\       /"  "\
48  (  (\  )___(  /)  )
49   \                 /
50   /                 \
51  /     () ___ ()     \
52  |       (   )       |
53   \       \_/       /
54     \...__!__.../
55           "
56  MELHOR LINGUAGEM DO MUNDO.
```

Gramática

- Paradigma;
- Baseada em C;
- Tipos primitivos;
- Tipada;

Gramática

- Várias funções;
- Kuala main;
- Main não tem tipo;

```
program : funct_list kuala funct_list
        | funct_list kuala
        | kuala funct_list
        | kuala
        ;
```

Gramática

- Permite definições;
- Permite blocos;
- Expressões e comandos;

```
param_dclr  : type_consts id { $2->sym->type = $1; $$ = $2;}
            | type_consts id '[' ']' {$2->sym->type = $1; $$ = $2;}
            ;
```

Gramática

```
stmt        : IF '(' logic ')' block {$$ = makeAST(IF_STATE
            | IF '(' logic ')' stmt {$$ = makeAST(IF_STATEME
            | ELSE IF '(' logic ')' block {$$ = makeAST(ELSE
            | ELSE block {$$ = makeAST(ELSE_STATEMENT,$2,0);
            | WHILE '(' rel ')' block {$$ = makeAST(WHILE_ST
            | FOR '(' expr_statement rel ';' attr ')' block
            | FOR '(' expr_statement rel ';' attr ')' stmt {
            | block
            | expr_statement stmt {$$ = makeAST(BLOCK_STATE
            | expr_statement
            | error  // continue parsing
            ;
```

Gramática

- Permite operações;
- Atribuição.

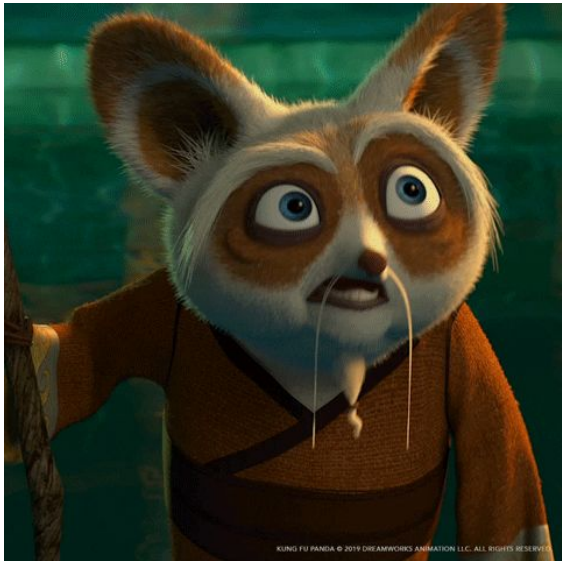
```
logic_op    : OR_OP {$$ = OR;}
            | AND_OP {$$ = AND;}
            ;

rel         : rel_oper relop rel_oper
            | rel_oper
            ;

relop       : LE_OP {$$=LE_OP_AST;}
            | EQ_OP {$$=EQ_OP_AST;}
            | LT_OP {$$=LT_OP_AST;}
            | GE_OP {$$=GE_OP_AST;}
            | GT_OP {$$=GT_OP_AST;}
            | DIF_OP {$$=DIF_OP_AST;}
            ;
```

Analisador Léxico

- Reconhece:
  - espaço;
  - caracteres especiais;
  - string;
  - identificadores;
  - números;
  - decimais;
- Sensitive case;
- Comentários

Analisador Semântico

- Atribuição de função;
- Tipos de operação;
- Definição de variável;



Analisador Semântico

```
Semantica - Valores de Operação incompatives; linha: 18
Semantico - A variavel 'salario' não foi declarada; linha: 18
 Semantico - Atribuição imcompativel; linha: 18
 Semantico - A variavel 'empregado' não foi declarada; linha: 19
 Semantico - Atribuição imcompativel; linha: 19
```

Representação Intermediária

- Árvore sintática abstrata
  - ibis a = 10 + (3 - 4) * 6;

```
(EX_EQ 'a' (PLUS_OP 10 (MUL_OP (MINUS_OP 3 4) 6)))
```

Dificuldades Encontradas

- Tabela de Símbolos;
- Criação da Gramática;
- Diferenças entre análise Semântica e Sintática;
- Geração de código Intermediário;
- A linguagem C;
- Todo o resto (?)

Execução

Conclusão

- Trabalho, mas rico em conhecimento;
- Cada parte ajudou na construção de outra parte;
- Contato maior com a “caixa preta” do compilador.

Perguntas?

Referências Bibliográficas

Lex - A Lexical Analyzer Generator . Disponível em:http://dinosaur.compilertools.net/lex/. Acesso em: 23 de out.

Yacc: Yet Another Compiler-Compiler . Disponível em: http://dinosaur.compilertools.net/yacc/ . Acesso em: 23 de out.

Tonius. Hash. Disponível em: https://gist.github.com/tonious/1377667. Acesso em: 02 de dez. de 2020.

lu1s. dragon-book-source-code. Disponível em: https://github.com/lu1s/dragon-book-source-code . Acesso em: 02 de dez. de 2020.