UNIVERSIDADE FEDERAL DE ITAJUBÁ

Arthur Gonçalves Marton - 2023009270
Felipe Rodrigues Malizia - 2023001414
Vicenzo Cenni - 2023001915

CRIAÇÃO E DESENVOLVIMENTO DE UM ANALISADOR LÉXICO E UM ANALISADOR SINTÁTICO PARA UMA LINGUAGEM DE PROGRAMAÇÃO

Trabalho da disciplina ECOM06A (Compiladores), apresentado à professora Thatyana de Faria Piola Seraphim, na Universidade Federal de Itajubá.

ITAJUBÁ
2025

Tokens

ANTIGO LEXEMA	LEXEMA NOVO	TOKEN
.	dota	TK_DOT
:	dotabb	TK_DOTB
;	dotasword	TK_DEL
,	sword	TK_SW
(	shield	TK_SHI
)	dleihs	TK_IHS
[	sqrShield	TK_SHI
]	dleihSrqs	TK_IHS
{	rise	TK_OKEY
}	fall	TK_IKEY
begin	rise	TK_OKEY
end	fall	TK_IKEY
int	alpha	TK_INTT
float	sigma	TK_FLOT
char	beta	TK_CHART
=	receive	TK_ATR
==	win	TK_COMP
>=	higher	TK_COMP
<=	lower	TK_COMP
>	high	TK_COMP
<	low	TK_COMP
!=	defeat	TK_COMP
+	buff	TK_MATH
-	nerf	TK_MATH
*	pwrup	TK_MATH
/	sliced	TK_MATH

mod	afk	TK_MATH
cout	redpill	TK_OUT
cin	bluepill	TK_IN
>>	upload	TK_MAD
<<	download	TK_MED
if	straif	TK_IF
else	dodge	TK_ELSE
while	grind	TK_WHI
and	coop	TK_LOG
or	cope	TK_LOG
not	denial	TK_LOG
(0 - 9)	(0 - 9)	TK_NUM
“	“	TK_ASP
‘	‘	TK_ASP
(a-z)	(a-z)	TK_LET
(A-Z)	(A-Z)	TK_LET
	-	TK_SIGN
	variável alpha	TK_INT
	variável sigma	TK_FLO
	variável beta	TK_CHAR
	variável	TK_VAR

Expressões Regulares dos Tokens:

Definições Gerais:

TK_LET = [a-zA-Z]
TK_NUM = [0-9]
TK_SIGN = “-”
TK_ASP = “ “ “ | “ ‘ “

Símbolos Especiais:

TK_DOT = “dota”
TK_DOTB = “dotabb”

TK_DEL = “dotaSword”
TK_SW = “sword”
TK_SHI = “shield” | “sqrShield”
TK_IHS = “dleihs” | “sqrShield”

Tipos de Dados:

TK_INT = (TK_SIGN)? . (TK_NUM)+
TK_FLO = (TK_SIGN)? . (TK_NUM)+ (TK_DOT . (TK_NUM)+)?
TK_CHAR = TK_ASP . (TK_LET | TK_NUM) . TK_ASP
TK_VAR = TK_LET . (TK_NUM | TK_LET)*
TK_INTT = “alpha”
TK_FLOT = “sigma”
TK_CHART = “beta”

Atribuição:

TK_ATR = “receive”

Operadores de Comparação:

TK_COMP = “win” | “defeat” | “high” | “low” | “higher” | “lower”

Operadores Lógicos:

TK_LOG = “coop” | “cope” | “denial”

Operadores Aritméticos:

TK_MATH = “buff” | “nerf” | “pwrup” | “sliced” | “afk”

Blocos de Comandos:

TK_OKEY = “rise”
TK_IKEY = “fall”

Entrada e Saída:

TK_OUT = “redpill”
TK_IN = “bluepill”
TK_MAD = “upload”
TK_MED = “download”

Comandos Condicionais:

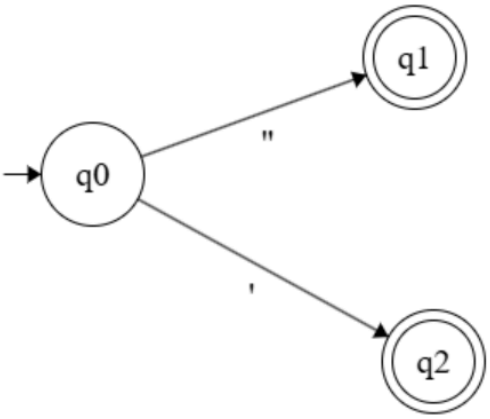
TK_IF = “straif”
TK_ELSE = “dodge”

Comando de Repetição:

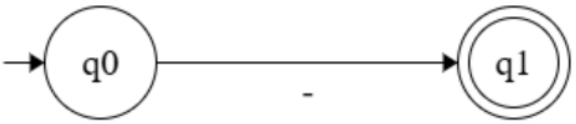
TK_WHI = “grind”

Autômatos

Definições Gerais:

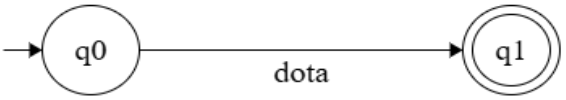


Token: TK_ASP

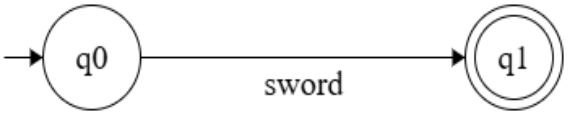


Token: TK_SIGN

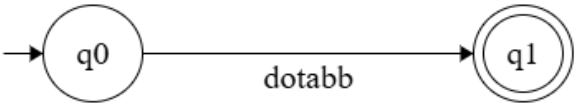
Símbolos Especiais:



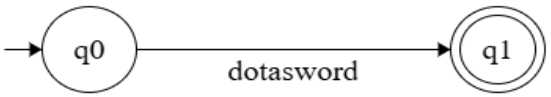
Token: TK_DOT



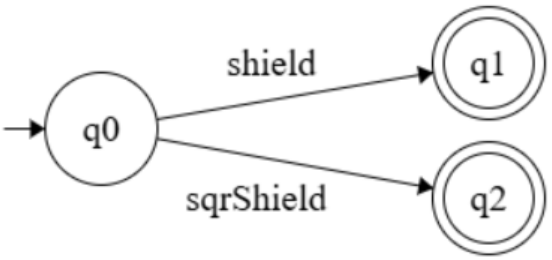
Token: TK_SW



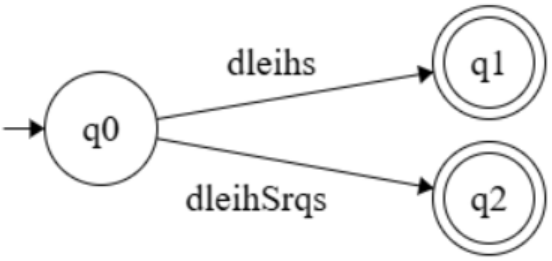
Token: TK_DOTB



Token: TK_DEL

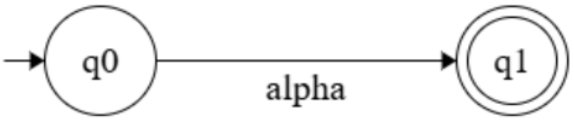


Token: TK_SHI

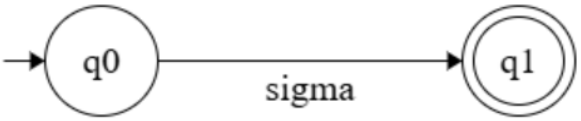


Token: TK_IHS

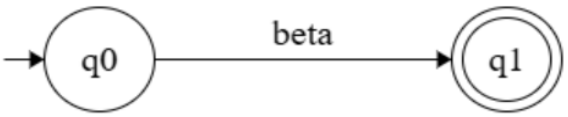
Tipos de Dados:



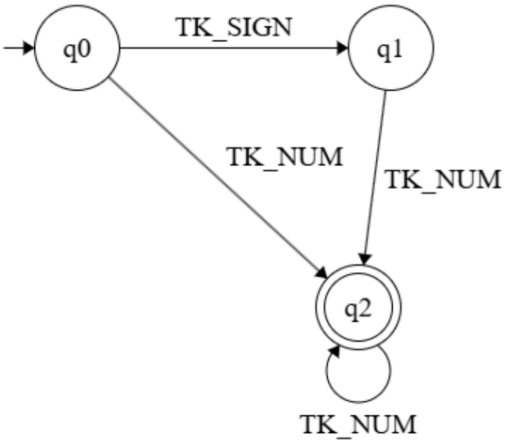
Token: TK_INTT



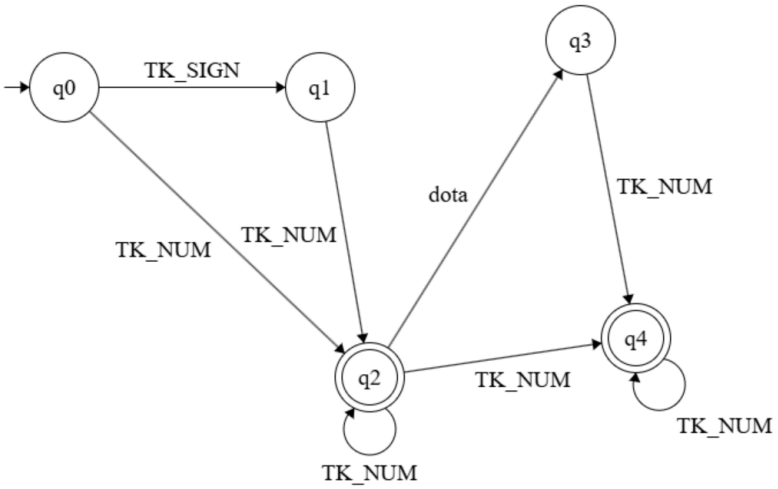
Token: TK_FLOT



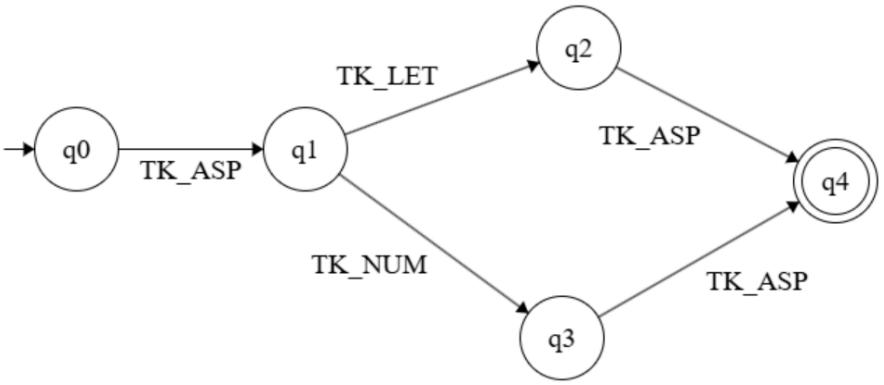
Token: TK_CHART



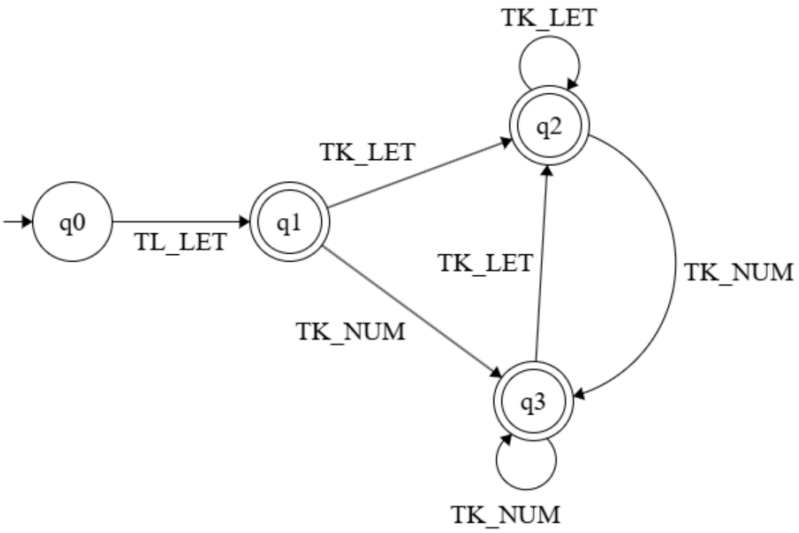
Token: TK_INT



Token: TK_FLO

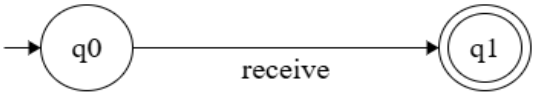


Toke: TK_CHAR



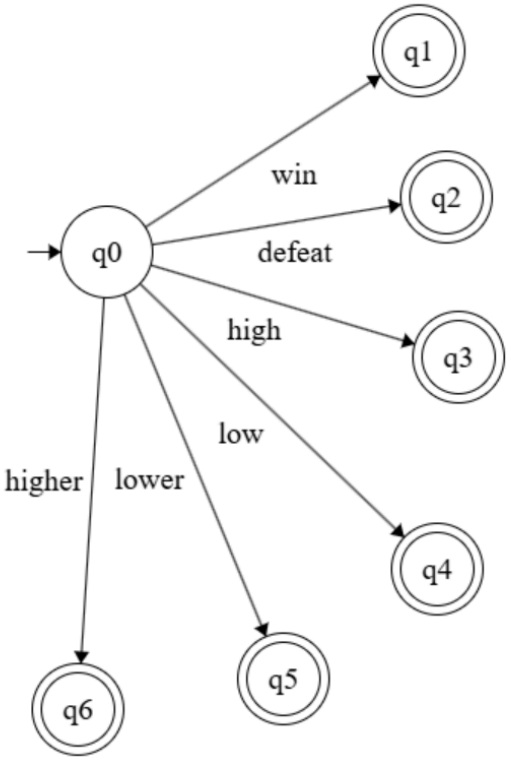
Token: TK_VAR

Atribuição:



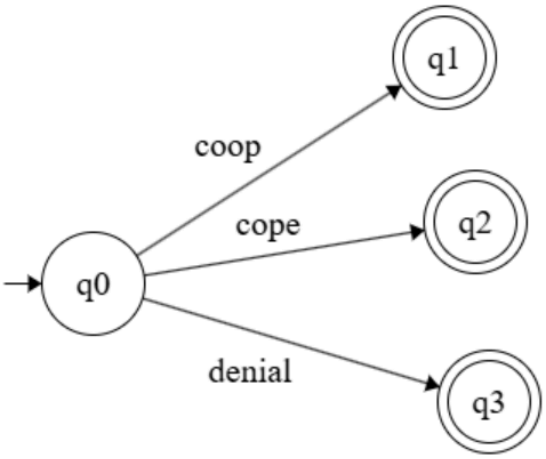
Token: TK_ATR

Operadores de Comparação:



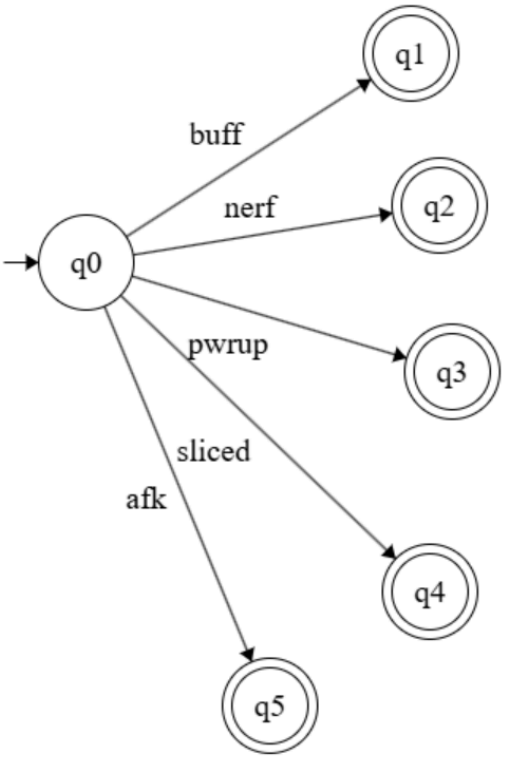
Token: TK_COMP

Operadores Lógicos:



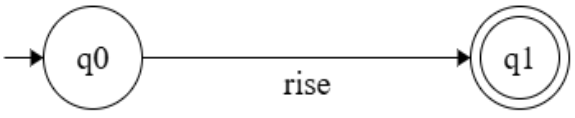
Token: TK_LOG

Operadores Aritméticos:

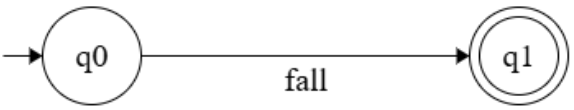


Token: TK_MATH

Blocos de Comandos:

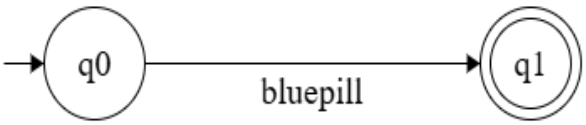


Token: TK_IKEY

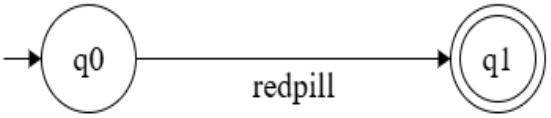


Token: TK_OKEY

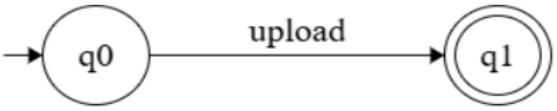
Entrada e Saída:



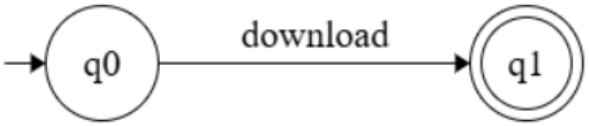
Token: TK_IN



Token: TK_OUT

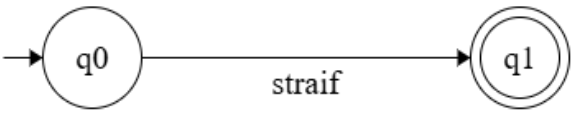


Token: TK_MAD

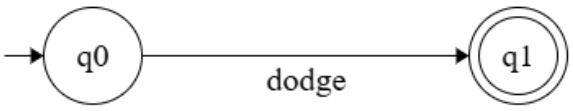


Token: TK_MED

Comandos Condicionais:

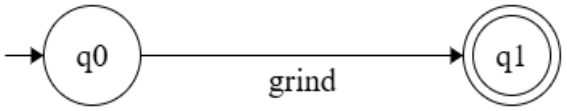


Token: TK_IF



Token: TK_ELSE

Comando de Repetição:



Token: TK_WHI

Regras das Produções:

MAIN -> TK_IKEY EXPRESSAO TK_OKEY

COMPARATIVA-> TK_VAR TK_COMP TK_VAR
| TK_VAR TK_COMP TK_INT
| TK_VAR TK_COMP TK_FLO
| TK_VAR TK_COMP TK_CHAR
| TK_FLO TK_COMP TK_VAR
| TK_INT TK_COMP TK_VAR
| TK_CHAR TK_COMP TK_VAR
| ARITMETICA TK_COMP TK_VAR
| TK_VAR TK_COMP ARITMETICA

LOGICO -> TK_VAR TK_LOG TK_VAR
| TK_VAR TK_LOG TK_VAR LOGICO
| TK_LOG TK_VAR
| TK_LOG TK_VAR LOGICO
| TK_VAR TK_LOG TK_INT
| TK_INT TK_LOG TK_VAR

ATRIBUICAO -> TK_INTT TK_VAR TK_ATR TK_INT
| TK_CHART TK_VAR TK_ATR TK_CHAR
| TK_FLOT TK_VAR TK_ATR TK_FLO
| TK_INTT TK_VAR TK_ATR ARITMETICA
| TK_CHART TK_VAR TK_ATR ARITMETICA
| TK_FLOT TK_VAR TK_ATR ARITMETICA
| TK_INTT TK_VAR TK_ATR LOGICO

ARITMETICA -> TK_VAR TK_MATH TK_VAR
| TK_VAR TK_MATH ARITMETICA
| TK_SHI ARITMETICA TK_IHS
| TK_INT TK_MATH ARITMETICA
| TK_FLO TK_MATH ARITMETICA

```
                |  TK_VAR TK_MATH TK_INT
                |  TK_VAR TK_MATH TK_FLO
                |  TK_INT TK_MATH TK_VAR
                |  TK_FLO TK_MATH TK_VAR

COND -> LOGICO
        | COMPARATIVA
        | LOGICO COND
        | COMPARATIVA COND
        | TK_SHI COND TK_IHS

NOCONDICAO -> TK_ELSE TK_IKEY EXPRESSAO TK_OKEY

CONDICAO -> TK_IF TK_SHI COND TK_IHS TK_IKEY EXPRESSAO TK_OKEY
                | CONDICAO NOCONDICAO

REPETICAO -> TK_WHI TK_SHI COND TK_IHS TK_IKEY EXPRESSAO TK_OKEY

EXPRESSAO -> ATRIBUICAO TK_DEL
                | ATRIBUICAO TK_DEL EXPRESSAO
                | CONDICAO
                | CONDICAO EXPRESSAO
                | REPETICAO
                | REPETICAO EXPRESSAO
                | IN TK_DEL
                | IN TK_DEL EXPRESSAO
                | OUT TK_DEL
                | OUT TK_DEL EXPRESSAO


IN -> TK_IN TK_MED TK_VAR
    | TK_IN TK_MED TK_VAR IN
    | TK_MED TK_VAR
    | TK_MED TK_VAR IN

OUT -> TK_OUT TK_MAD TK_VAR
        | TK_OUT TK_MAD TK_VAR OUT
        | TK_MAD TK_VAR
        | TK_MAD TK_VAR OUT
        |TK_OUT TK_MAD TK_CHA
        | TK_OUT TK_MAD TK_CHA OUT
        | TK_MAD TK_CHA
        | TK_MAD TK_CHA OUT
```

Exemplo de Aplicação de Código:

```
rise

alpha i  receive 0 dotasword
sigma var receive 1 buff 2 dotasword
alpha gon receive 1 dotasword
beta sir receive ‘i’ dotasword

grind shield i lower 12 dleihs rise
        redpill upload sir upload ‘=’ upload i dotasword
        straif sqrShield shield var win 3 dleihs coop gon dleihSrqs rise
                redpill upload ‘n’ upload ‘i’ upload ‘c’ upload ‘e’ dotasword
                alpha gon receive denial gon dotasword
                fall
        dodge rise
                bluepill download var dotasword
                alpha gon receive denial gon dotasword
                fall
        alpha i receive var buff 1 dotasword
        fall
redpill upload ‘f’ upload ‘i’ upload ‘m’ dotasword

fall
```