**Análise Sintática Preditiva - Parte 2**

**Exercício Individual**

**Nome**: João Lucas Lima de Melo

Considere a gramática **G1:**

S → E$

1. E → E + T
2. E → E - T
3. E → T
4. T → T \* F
5. T → T / F
6. T → F
7. F → a
8. F → (E)
9. Definir uma gramática **G1'** a partir de G1 em que a recursividade à esquerda foi substituída por **iteração**.

**G1’**:

S → E$

1. E → T {+T | -T}
2. T → F {\*F | /F}
3. F → a
4. F → (E)
5. Mostrar a derivação mais à esquerda (leftmost) da cadeia **a + a \* a** com **G1'**.

E → T

**T** → T+ T

**T** + T → F + T

**F** + T → a + T

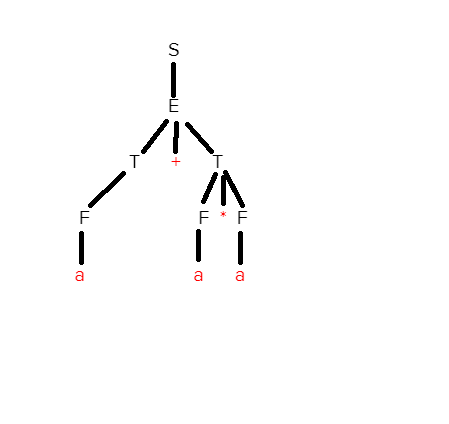
a + **T** →a + F

a + **F** → a + F \* F

a + **F** \* F → a + a \* F

a + a \* **F** → a + a \* a

1. Desenhar a árvore sintática correspondente para a derivação de B.



1. Escreva as funções para a análise sintática descendente recursiva a partir da gramática **G1'** (siga o modelo de pseudo-código usado nos slides da aula).

void S(){

E();

}

void E(){

T();

enquanto tok pertence {+, -}{

eat(tok);

T();

}

}

void T(){

F();

enquanto tok pertence {\*, /}{

eat(tok);

F();

}

}

void F(){

if tok == ‘(‘{

eat( ‘(‘ );

E();

eat( ’)’ );

}else{

eat(a)

}

}