**Начальное условие лабораторной работы:**

**Типы токенов**

* +
* -
* \*
* /
* (
* )
* **id**
* **num**

**Грамматика**

E → E+T | E-T | T  
T → T\*F | T/F | F  
F → **id** | **num** | (E)

Начало работы

1)Расширение грамматики (для удаления левосторонней рекурсии):

E → E’  
E’ → +TE’ | -TE’ | e  
T → T’

T’ → \*FT’ | /FT’ | e

F → id | num | (E)

2)Построение множеств FIRST (множество токенов, с которых может начинаться левая часть правила грамматики) и FOLLOW (множество токенов, которые могут стоять после первого символа правой части грамматики)

|  |  |  |
| --- | --- | --- |
|  | FIRST | FOLLOW |
| E | {id, num, (} | {$, )} |
| E’ | {+, -, )} | {$, )} |
| T | {id, num, (} | {+, -, $, )} |
| T’ | {+, -, \*, /, )} | {+, -, $, )} |
| F | {id, num, (} | {\*, /, +, -, ), $} |

Где $ - символ конца ввода строки

3)Построение таблицы переходов (показывает, какое правило применить в зависимости от встреченного символа) (столбцы – токен, строки – какое правило применить):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | id | num | + | - | \* | / | ( | ) | $ |
| E | TE’ | TE’ | -------- | -------- | -------- | -------- | TE’ | -------- | e |
| E’ | -------- | -------- | +TE’ | -TE’ | -------- | -------- | -------- | e | e |
| T | FT’ | FT’ | -------- | -------- | -------- | -------- | FT’ | -------- | e |
| T’ | -------- | -------- | e | e | \*FT’ | /FT’ | -------- | e | e |
| F | id | num | -------- | -------- | -------- | -------- | (E) | -------- | e |

Например, для выражения id + num$ выполнятся следующие переходы (начальное состояние E):

1. Текущий символ = id, переходим по правилу E → TE’
2. Текущий символ = id, переходим по правилу T → FT’
3. Текущий символ = id, переходим по правилу F → id, текущий символ = +
4. Текущий символ = +, переходим по правилу T’ → e
5. Текущий символ = +, переходим по правилу E’ → +TE’, текущий символ = num
6. Текущий символ = num, переходим по правилу T → FT’
7. Текущий символ = num, переходим по правилу F → num, текущий символ = $
8. Текущий символ = $, переходим по правилу T’ → e
9. Текущий символ = $, переходим по правилу E’ → e
10. Текущий символ = $, переходим по правилу E’ → e
11. Конец

Если выражение будет неверным, например +id, то перехода из Е по + не существует, и программа возвращает ложь (строка не поддерживается). Аналогично программа работает и для других ошибок.

В программе 3 файла: Source.cpp, Header.h, Header.cpp

Пояснения к программе в самой программе в виде комментариев.