

Лабораторная работа 20 (2 часа) Конструирование программного обеспечения

Расширение функции преобразования выражений с вызовом функций в форму обратной польской записи

1. Используйте материалы лекций (Алгоритмы синтаксического разбора. Польская запись) и результат лабораторной работы № 18.
2. Реализуйте расширенный алгоритм построения обратной польской записи (лекция № 13) в функции **PolishNotation** и протестируйте на различных выражениях (использовать все арифметические выражения контрольного примера, включая вызовы функций).
3. Ознакомьтесь со спецификацией функции **PolishNotation**, представленной ниже.

```
#include <iostream>
#include <locale>
#include "LT.h"      // таблица лексем
#include "IT.h"      // таблица идентификаторов

#define EXP1 28      // позиция первого выражения
#define EXP2 50      // позиция второго выражения
#define EXP3 66      // позиция третьего выражения
// и т.д.
bool PolishNotation( // построение польской записи в таблице лексем
    int          lextable_pos, //позиция выражения в lextable
    LT::LexTable& lextable,    //таблица лексем
    IT::IdTable&  idtable      //таблица идентификаторов
);
// PolishNotation() == true построение польской записи выполнено успешно
// PolishNotation() == false построение польской записи не выполнено
int _tmain(int argc, _TCHAR* argv[])
{
    setlocale(LC_ALL, "rus");
    LT::LexTable lextable; //таблица лексем
    IT::IdTable idtable;  //таблица идентификаторов
    std::cout <<28<<" : польская запись построена"<<std::endl;
    // заполнение таблиц lextable и idtable см. лаб. 14
    if (PolishNotation(EXP1, lextable, idtable))
        std::cout <<EXP1<<" : польская запись построена"<<std::endl;
    else std::cout <<EXP1<<" : польская запись не построена"<<std::endl;
    if (PolishNotation(EXP2, lextable, idtable))
        std::cout <<EXP2<<" : польская запись построена"<<std::endl;
    else std::cout <<EXP2<<" : польская запись не построена"<<std::endl;
    if (PolishNotation(EXP3, lextable, idtable))
        std::cout <<EXP3<<" : польская запись построена"<<std::endl;
    else std::cout <<EXP3<<" : польская запись не построена"<<std::endl;
    //...
    return 0;
};
```

```

// пример
// PolishNotation(29, lextable, idtable);
//      входная      выходная
//      таблица лексем      таблица лексем
//      ...
// 25 ...      25 ...
// 26 ...      26 ...
// 27 i        27 i
// 28 =        28 =
// 29 i        29 i
// 30 (        30 i
// 31 i        31 i
// 32 ,        32 @₃
// 33 i        33 i
// 34 ,        34 v
// 35 i        35 ;
// 36 )        36 null
// 37 v        37 null
// 37 i        37 null
// 37 ;        37 null
// 37 ...      37 ...
// 37 ...      37 ...
// 38 ...      38 ...
// 39 ...      39 ...
// 40 ...      40 ...
// 41 ...      41 ...
//

```

4. Преобразования выражений выполняются после синтаксического разбора. Функция **PolishNotation** принимает три параметра (см. спецификацию): позицию выражения (первого символа выражения) в таблице лексем (по значению); таблицу лексем (по ссылке, заполнена в лабораторной 17), таблицу идентификаторов (по ссылке, заполнена в лабораторной 17), возвращает **true** (если преобразование выражения в польскую запись выполнено успешно) или **false** (преобразование не выполнено) и выводит выражение, полученное в результате преобразования, в консоль. Кроме того, функция **PolishNotation** изменяет таблицу лексем: вместо исходного выражения, записывается выражение в ПОЛИЗ, полученное в результате преобразования. Элементы таблицы лексем, которые остались свободными (длина польской записи всегда меньше или равна исходной) заполняются символом-заполнителем (любой символ из числа запрещенных).
5. Протестируйте **PolishNotation** на *всех* выражениях из контрольного примера лабораторной работы 17.