



Иногда для того чтобы лучше понять как работают процедуры и функции модуля 1С, лучше представить цепочку их вызовов (стек вызовов) в виде кластеров имен функций.

Посмотрим работу программы на простом примере. Модули определения процедур и функций в 1С имеют примерно такую структуру:

```
Процедура Проц6()  
    Проц5();  
КонецПроцедуры
```

```
Процедура Проц5()  
    Проц4();  
КонецПроцедуры
```

Кластеризация имен функций

Автор: А.Волос - Обновлено 30.01.2015 06:07

```
Процедура Проц4()  
  Проц3();  
КонецПроцедуры
```

```
Процедура Проц3()  
  Проц2();  
КонецПроцедуры
```

```
Процедура Проц2()  
  Проц1();  
КонецПроцедуры
```

```
Процедура Проц1()  
  Сообщить("Привет!!!");  
КонецПроцедуры
```

```
////////////////////  
Процедура Проц33()  
  Проц1();  
  Проц3();  
  Проц6();  
КонецПроцедуры
```

```
Процедура Проц35()  
  Проц1();  
  Проц2();  
  Проц4();  
КонецПроцедуры
```

После обработки этого файла программой результат будет таким:

Кластеризация имен функций

Автор: А.Волос - Обновлено 30.01.2015 06:07

+ Проц6

-- Проц5
- -- Проц4
- - - Проц3
- - - - Проц2
- - - - - Проц1

+ Проц5
-- Проц4
- -- Проц3
- - - Проц2
- - - - Проц1

+ Проц4
-- Проц3
- -- Проц2
- - - Проц1

+ Проц3
-- Проц2
- -- Проц1

+ Проц2
-- Проц1

+ Проц1

Кластеризация имен функций

Автор: А.Волос - Обновлено 30.01.2015 06:07

```
+ Проц33
-- Проц1
-- Проц3
- - - Проц2
- - - - Проц1
-- Проц6
- - - Проц5
- - - - Проц4
- - - - - Проц3
- - - - - - Проц2
- - - - - - - Проц1
```

```
+ Проц35
-- Проц1
-- Проц2
- - - Проц1
-- Проц4
- - - Проц3
- - - - Проц2
- - - - - Проц1
```

Текст программы:

```
{codecitation style="brush: php;"}  
  
<?php
```

```
$mas_names = array();

//-----
// откроем файл для вывода ошибок
$filename = "error.txt";
if(!($errorFile = fopen($filename, "w"))){
    print("$filename' could not be createdn");
    exit;
}
//-----
// откроем файл для вывода
$filename = "fileOut.txt";
if(!($fileOut = fopen($filename, "w"))){
    fputs($errorFile, "$filename' could not be createdn ");
    exit;
}
//-----
// откроем файл для чтения
$filename = "data.txt";
if(!($myFile = fopen($filename, "r"))){
    fputs($errorFile, "$filename' could not be openedn ");
    exit;
}

$kn=0;
//-----
//считаем построчно из файла data.txt в файл fileOut.txt
while(!feof($myFile)){
    //читаем строку из файла
    $myLine = fgets($myFile, 1080);
    $str_temp = " $myLine";

    if((strpos($str_temp, "Функция") || strpos($str_temp, "Процедура")) && strpos($str_temp,
"(") ) { //лексема найдена

        $str_temp = trim($myLine);

        // разбивает строку в массив по пробелу.
        $mas_str1 = explode(" ", $str_temp);
```

Кластеризация имен функций

Автор: А.Волос - Обновлено 30.01.2015 06:07

```
        // берет второй элемент массива (в нем должно присутствовать имя
функции или процедуры).
        $str_temp3 = $mas_str1[1];

        // разбивает его (этот элемент-строку) в массив по открывающей скобке.
        $mas_str2 = explode("(", $str_temp3);

        // берет первый элемент массива - в нем должно быть чистое имя функции
        $str_temp4 = $mas_str2[0];

        // запоминает в переменной имя процедуры или функции,
        $fun_name = trim($str_temp4); //убрать пробелы по краям

        // все найденные имена сохраняет в массиве.
        $mas_names[$kn] = $fun_name;

        $kn++;

    } //end if лексема найдена
}

//массив имен процедур и функций создан. Выведем его.
//printMassiv($mas_names);

////////////////////////////////////
// закроем файлы
fclose($myFile);
//-----
// откроем файл для чтения
$filename = "data.txt";
if(!($myFile = fopen($filename, "r"))){
    fputs($errorFile, "'$filename' could not be openedn" );
}
```

Кластеризация имен функций

Автор: А.Волос - Обновлено 30.01.2015 06:07

```
    exit;
}
////////////////////////////////////
////////// Повторное чтение файла //////////
////////////////////////////////////
//-----
//-----
//считаем построчно из файла data.txt в файл fileOut.txt
$flag = false;
$kn = 0;
$fun_name = "";

$index = 0;

while(!feof($myFile)){
    //читаем строку из файла
    $myLine = fgets($myFile, 1080);
    $str_temp = " $myLine";

    if($flag = true && !$fun_name == ""){ //эта ветка выполняется после того, как найдено
начало функции

        //в каждой строке будем искать элемент массива, если находим, то этот
элемент складываем в локальный массив
        foreach ($mas_names as $value) { //скан индексного массива имен функций
            if(strpos($str_temp, $value)) { //в этой строке есть один из элементов
индексного массива (есть имя одной из функций)
                if($fun_name != $value){ //если элемент индексного массива не равен
имени вызывающей функции

                    //создадим ассоц массив с ключом - элементом массива
                    $ass_mas[$value] = array();

                    //добавим созданный ассоц массив в индексный массив
                    array_push($mas_n[$fun_name], $ass_mas);
                    array_shift($ass_mas); //удалим элемент
```

```
                break;
            }
        }
    }

    if(strpos($str_temp, "КонецФункции") || strpos($str_temp, "КонецПроцедуры") )
    { //лексема найдена

        $flag = false;
        $fun_name = "";
    }
}

if((strpos($str_temp, "Функция") || strpos($str_temp, "Процедура")) && strpos($str_temp,
"(") ) { //лексема найдена

    $flag = true;
    $str_temp = trim($myLine);

    // разбивает строку в массив по пробелу.
    $mas_str1 = explode(" ", $str_temp);

    // берет второй элемент массива (в нем должно присутствовать имя функции
или процедуры).
    $str_temp3 = $mas_str1[1];

    // разбивает его (этот элемент-строку) в массив по открывающей скобке.
    $mas_str2 = explode("(", $str_temp3);

    // берет первый элемент массива - в нем должно быть чистое имя функции
    $str_temp4 = $mas_str2[0];

    // запоминает в переменной имя процедуры или функции,
```



```
$fun_name = trim($str_temp4); //убрать пробелы по краям

$mas_n["$fun_name"] = array();

    }
    $index++;
}

echo("<br><br>");

$_index = 0;
$mas_out = array();

showStek($mas_n, $mas_out, $_index);

// закроем файлы
fclose($myFile);
fclose($fileOut);
fclose($errorFile);

//
function showStekReverce($mass, $element, $_index){

    $_index++;
    $strr = "-- $strr ";

    for($i = 0; $i<$_index;$i++){
        $strr = "- $strr ";
    }

    foreach($mass as $elem=>$key){//сканируем $mass - - входящий ассоц массив в поиске совпадения
```

Кластеризация имен функций

Автор: А.Волос - Обновлено 30.01.2015 06:07

```
$kl = count($key);//число элементов массива
if($kl != 0){//если массив не пустой

    if($element == $elem){//совпадение! В $mass - входящем массиве есть ключ,
совпадающий с переданным в параметре

        foreach($key as $el=>$ke){//сканируем индексный массив второго уровня
            foreach($ke as $e=>$k){//сканируем $ke - ассоц массив второго уровня
                //
                echo("$strr $e <br>");
                showStekReverse($mass, $e, $_index);
            }
        }
    }
}

//
function showStek($mass, $mas_out, $_index){

    foreach($mass as $elem=>&$key){//сканируем $mass - входящий ассоц массив!

        echo(" <br>");
        echo("+ $elem <br>");
        $kl = count($key);//число элементов массива
        if($kl != 0){//если массив не пустой

            foreach($key as $el=>&$ke){//сканируем индексный массив второго уровня
                foreach($ke as $elem3=>&$key3){//сканируем $ke - ассоц массив второго уровня!
                    echo("-- $elem3 <br>");
                    showStekReverse($mass, $elem3, $_index);
                }
            }
        }
    }
    return;
}

?>
```

{/codecitation}

Как пользоваться этой программой:

1. Копируете текст файла или модуля 1С с определениями процедур и функций - в файл data.txt.

Файл data.txt должен находиться в одном каталоге со скриптом программы.

2. Запускаете скрипт программы. Результат будет выведен на экран.