

Раздел «Язык Си» . CoffeeFunctionR :

- Функция вызывает другую функцию.
- Рекурсивный вызов функции
  - Примеры рекурсивных программ
    - Факториал
    - Обратная строка
  - Задачи.
    - Задача 1.
    - Задача 2.
    - Задача 3.
    - Задача 4.
    - Задача 5.
    - Задача 6\*
    - Контекст

Функция вызывает другую функцию.

Пусть имеются три функции: `int getS()` – получить расстояние, `int convToM(int)` – перевести в метры, `float getV(int)` – вычислить скорость.

Вызов функций в программе:

```
int convToM( int s){
// умеет переводить в метры
return s;
};

int getS(){
// как-то получили расстояние s

// вызов функции конвертирования
// и возврат
return convToM(s);
};

float getV( int time){
// что-то делает для вычислений
v = getS() / time;
return s;
};

int main(){
// что-то выполняем
// вызов функции
v = getV(45);
};
```

Не все функции могут сразу отработать. Некоторые требуют вызова других функций и вынуждены ждать пока те не закончат свою работу. Все функции помещаются в СТЕК. Там находятся ВСЕ локальные переменные функции и информация какую инструкцию нужно выполнять когда завершится работа вызванной функции.

Все вызванные функции занимают место в СТЕКЕ.

СТЕК – это раздел памяти. Его размеры ограничены. `main()`, затем `getV()`, затем `getS()` и последней `convToM()` Первой помещается функция

<pre>int convToM( int s){ // умеет переводить в метры return s; };</pre>	Выполняется самая "глубокая" функция
<pre>int getS(){ // как-то получили расстояние s  // вызов функции конвертирования // и возврат return convToM(s); };</pre>	Ждет когда convToM() вернет значение
<pre>float getV( int time){ // что-то делает для вычислений v = getS() / time; return s; };</pre>	Ждет пока getS() вернет значение
<pre>int main(){ // что-то выполняем // вызов функции v = getV(45); };</pre>	ждет пока getV() вернет значение

В данном примере все функции завершают свою работу по собственным правилам, которые не зависят от той функции, которая их вызывала.

Рекурсивный вызов функции

Поиск

Поиск

Раздел «Язык Си»

Главная

Зачем учить C?

Определения

Инструменты:

Поиск

Изменения

Index

Статистика

Разделы

Информация

Алгоритмы

Язык Си

Язык Ruby

Язык Ассемблера

EI Judge

Парадигмы

Образование

Сети

Objective C

Logon>>

Можно организовать вызов не другой функции, а ТОЙ ЖЕ САМОЙ функции.

Все вызванные функции также будут помещаться в СТЕК и занимать в нем место.

Однако, если все РАЗНЫЕ функции завершались по своим правилам, то для всех вызовов рекурсивно функции правила работы и завершения ОДИНАКОВЫЕ.

Рассмотрим два примера:

### Сказка про белого бычка

Бесконечная сказка

```
#include <stdio.h>
#include <stdlib.h>

void kozlik(){
    char buf[100];
    printf("Рассказать тебе сказку про белого бычка?\n");
    // считать строку
    scanf("%s",buf);
    // напечатать строку
    printf("Все говорят: %s.\n", buf);
    // вызов той же самой функции
    // она, конечно, начнет работать сначала
    // и рпять задаст вопрос.
    // нет никаких условий, чтобы перестать вызывать функцию
    // (см. код этой функции)
    // Значит она будет вызываться "бесконечно".
    // То есть пока не кончится память в стеке или пока
    // не лопнет терпение пользователя :)
    kozlik();
};

int main(){
    kozlik();
    return 0;
}
```

"Щадающий" вариант (с завершением)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void kozlik(){
    char buf[100];
    printf("Рассказать тебе сказку про белого бычка?\n");
    // мы хотим получать ЦЕЛЫЕ строки
    // а scanf() считает, что строка заканчивается разделителем
    // fgets() читает ВСЮ строку до \n и в конце оставляет \n
    // указываем сколько максимально можем прочитать символов за раз
    // и откуда читаем: stdin - значит с клавиатуры
    fgets(buf,100, stdin);
    // узнаем длину прочитанной строки (в символах)
    // функция strlen()
    int z = strlen(buf);
    // поменяем \n на конец строки \0
    buf[z - 1] = '\0';

    //=====постараемся завершить функцию:
    // сравним строки strcmp()!!!
    if ( strcmp( buf, "отзынь на три лапти") == 0 )
    // если строки равны - прерываем функцию return
        return;
    // если нет, продолжаем вызывать kozlik()
    printf("Все говорят: %s.\n", buf);
    kozlik();
};

int main(){
    // первый вызов kozlik()
    kozlik();
    return 0;
}
```

### Примеры рекурсивных программ

#### Факториал

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
long long fact(int n){
    long long f;
    printf("n=%d\n", n);
    // если n==1 или n==0,
    // завершаем функцию и возвращаем 1
    if ( n == 1 || n == 0){
        // печать для наглядности
        printf("res: 1\n");
        return 1;
    }
    // рекурсивный вызов функции
    // чтобы не работать бесконечно,
    // аргументы функции при вызовах должны изменяться
    // и не повторяться
    f = fact(n - 1) * n;
    // печать "для наглядности"
    printf("res: %lld\n", f);
    return f;
};

int main(){
    int n;
    scanf("%d", &n);
    printf("%lld\n", fact(n) );
    return 0;
}
```

#### Обратная строка

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void backStr(char * z){
    int len = strlen(z);
    char *p;
    // указатель на начало строки
    p = z;
    // если достигнут конец строки
    // завершим функцию
    if ( p[0] == '\0' ){
        return;
    }
    // рекурсивный вызов для указателя на
```

```
// СЛЕДУЮЩУЮ букву
backStr(p + 1);
// когда завершена функция, можно печатать буквы
printf("%c",p[ 0 ]);
return;
};

int main(){
char buf[100];
scanf("%s",buf);
backStr(buf);
return 0;
}
```

## Задачи.

### Задача 1.

Написать РЕКУРСИВНУЮ функцию вычисления НОД.

### Задача 2.

Написать рекурсивную и нерекурсивную функцию вычисления чисел Фибоначчи. Каждая функция должна напечатать само число и количество итераций, требуемых для вычислений.

### Задача 3.

Написать рекурсивную функцию, которая инвертирует заданное число ( $N \leq 2^{32} - 1$ ).

Программа должна напечатать это число, деленное на 2.

### Задача 4.

В символьном массиве *NxM* точка – белая клетка, \* – черная. Черными клетками нарисована фигура. При этом все клетки фигуры соприкасаются хотябы с одной другой клеткой этой фигуры по стороне клетки. Написать рекурсивную функцию, которая закрашивает фигуру заданным цветом (символом, отличным от \*).

### Задача 5.

В символьном массиве *NxM* точка – белая клетка, \* – черная. Черными клетками нарисованы несколько фигур (см. задачу 3). При фигуры не соприкасаются и не пересекаются . Написать функцию, которая закрашивает каждую фигуру своим цветом, не совпадающим с цветом других фигур и \*.

### Задача 6\*

Молоко привезли в цистерне.

В кувшин помещается *n* литров молока, в банку – *m* литров. Других способов отмерить у продавца нет.


Необходимо отлить ТОЧНОЕ количество молока покупателю, используя только кувшин и банку. Молоко из кувшина и банки в цистерну переливать можно, между кувшином и банкой – тоже можно. Нельзя переливать молоко из тары покупателя.

Налить молоко нужно за МИНИМАЛЬНОЕ количество операций.

## Контеcт

- [контеcт](#) (экспериментальный контеcт, только для группы Овсянниковой Т.В.)

-- [TatyanaOvsyannikova2011](#) – 09 Nov 2016

Attachment	Action	Size	Date	Who	Comment
 funct.pdf	<a href="#">manage</a>	509.3 K	09 Nov 2016 – 10:19	<a href="#">TatyanaOvsyannikova2011</a>	

(с) Материалы раздела "Язык Си" публикуются под лицензией [GNU Free Documentation License](#).