

# Методика решения рекурсивных алгоритмов (проблемы решения задач данного типа а ЕГЭ)

---

*Миляева О.И., учитель информатики и  
ИКТ МБОУ СОШ № 9 г. Холмска*

**Рекурсия** – это свойство объекта подражать самому себе. Объект является рекурсивным если его части выглядят также как весь объект. Рекурсия очень широко применяется в математике и программировании:

$$F(n) = n * (n-1) * \dots * 1, 0! = 1$$

Рекурсивное определение этой функции имеет вид:

$$F(n) = \begin{cases} 1, n = 0 \\ n * F(n-1), n > 0 \end{cases}$$

где  $F(n-1) = (n-1)!$

## Что нужно знать:

рекурсия – это приём, позволяющий свести исходную задачу к одной или нескольким более простым задачам того же типа

чтобы определить рекурсию, нужно задать  
условие остановки рекурсии (базовый случай или несколько базовых случаев)  
рекуррентную формулу

любую рекурсивную процедуру можно запрограммировать с помощью цикла

рекурсия позволяет заменить цикл и в некоторых сложных задачах делает решение более понятным, хотя часто менее эффективным

существуют языки программирования, в которых рекурсия используется как один из основных приемов обработки данных (Lisp, Haskell)

Ниже на пяти языках программирования записана рекурсивная функция (процедура) F.

<b>Бейсик</b>	<b>Python</b>
<pre>SUB F(n)   print n,   IF n &gt;= 7 THEN     F(n - 3)     F(n - 1)   END IF END SUB</pre>	<pre>def F(n):     print(n, end='')     if n &gt;= 7:         F(n - 3)         F(n - 1)</pre>
<b>Алгоритмический язык</b>	<b>Паскаль</b>
<pre><u>алг</u> F(<u>цел</u> n) <u>нач</u>   <u>вывод</u> n   <u>если</u> n &gt;= 7 <u>то</u>     F(n - 3)     F(n - 1)   <u>все</u> <u>кон</u></pre>	<pre>procedure F(n: integer); begin   write(n);   if n &gt;= 7 then   begin     F(n - 3);     F(n - 1)   end end;</pre>
<b>Си</b>	
<pre>void F(int n) {   printf("%d", n);   if (n &gt;= 7) {     F(n - 3);     F(n - 1);   } }</pre>	

Что выведет программа при вызове F(9)? В ответе запишите последовательность введенных цифр слитно (без пробелов).

Исполним алгоритм для указанного аргумента.

Команда алгоритма	Результат исполнения	Вывод	Примечание
<u>вывод</u> n	<b>ВЫВОД «9»</b>	9	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>9 <math>\geq</math> 7, ИСТИНА</b>		
F (n - 3)	<b>ВЫЗОВ F(6)</b>		1 уровень рекурсии
<u>вывод</u> n	<b>ВЫВОД «6»</b>	6	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>6 <math>\geq</math> 7, ЛОЖЬ</b>		возврат в F(9)
F (n - 1)	<b>ВЫЗОВ F(8)</b>		1 уровень рекурсии
<u>вывод</u> n	<b>ВЫВОД «8»</b>	8	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>8 <math>\geq</math> 7, ИСТИНА</b>		
F (n - 3)	<b>ВЫЗОВ F(5)</b>		2 уровень рекурсии
<u>вывод</u> n	<b>ВЫВОД «5»</b>	5	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>5 <math>\geq</math> 7, ЛОЖЬ</b>		возврат в F(8)
F (n - 1)	<b>ВЫЗОВ F(7)</b>		2 уровень рекурсии
<u>вывод</u> n	<b>ВЫВОД «7»</b>	7	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>7 <math>\geq</math> 7, ИСТИНА</b>		
F (n - 3)	<b>ВЫЗОВ F(4)</b>		3 уровень рекурсии
<u>вывод</u> n	<b>ВЫВОД «4»</b>	4	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>4 <math>\geq</math> 7, ЛОЖЬ</b>		возврат в F(7)
F (n - 1)	<b>ВЫЗОВ F(6)</b>		3 уровень рекурсии
<u>вывод</u> n	<b>ВЫВОД «6»</b>	6	
<u>если</u> n $\geq$ 7 <u>то</u>	<b>6 <math>\geq</math> 7, ЛОЖЬ</b>		возврат в F(7)
<u>кон</u>			возврат в F(8)
<u>кон</u>			возврат в F(9)
<u>кон</u>			завершение алгоритма

Ответ: 9685746

## Задание 11.2

Ниже на пяти языках программирования записаны две рекурсивные функции (процедуры): F и G.

Алгоритмический язык	Паскаль
<pre><u>алг</u> F(<u>цел</u> n) <u>нач</u>   <u>если</u> n &gt; 0 <u>то</u>     G(n - 1)   <u>все</u> <u>кон</u></pre> <pre><u>алг</u> G(<u>цел</u> n) <u>нач</u>   <u>вывод</u> "*"   <u>если</u> n &gt; 1 <u>то</u>     F(n - 2)   <u>все</u> <u>кон</u></pre>	<pre>procedure F(n: integer); forward; procedure G(n: integer); forward;  procedure F(n: integer); begin   if n &gt; 0 then     G(n - 1); end;  procedure G(n: integer); begin   writeln('*');   if n &gt; 1 then     F(n - 2); end;</pre>

Сколько символов «звёздочка» будет напечатано на экране при выполнении вызова F(11)?

Команда алгоритма	Результат исполнения	Вывод	Примечание
<u>если</u> $n > 0$ <u>то</u>	$11 > 0$ , истина		выполняется F(11)
G(n - 1)	вызов G(10)		
<u>вывод</u> "*"	ВЫВОД «*»	*	выполняется G(10)
<u>если</u> $n > 1$ <u>то</u>	$10 > 1$ , истина		
F(n - 2)	вызов F(8)		
<u>если</u> $n > 0$ <u>то</u>	$8 > 0$ , истина		выполняется F(8)
G(n - 1)	вызов G(7)		
<u>вывод</u> "*"	ВЫВОД «*»	*	выполняется G(7)
<u>если</u> $n > 1$ <u>то</u>	$7 > 1$ , истина		
F(n - 2)	вызов F(5)		
<u>если</u> $n > 0$ <u>то</u>	$5 > 0$ , истина		выполняется F(5)
G(n - 1)	вызов G(4)		
<u>вывод</u> "*"	ВЫВОД «*»	*	выполняется G(4)
<u>если</u> $n > 1$ <u>то</u>	$4 > 1$ , истина		
F(n - 2)	вызов F(2)		
<u>если</u> $n > 0$ <u>то</u>	$2 > 0$ , истина		выполняется F(2)
G(n - 1)	вызов G(1)		
<u>вывод</u> "*"	ВЫВОД «*»	*	выполняется G(1)
<u>если</u> $n > 1$ <u>то</u>	$1 > 1$ , ложь		
<u>кон</u>			завершение G(1)
<u>кон</u>			завершение F(2)
<u>кон</u>			завершение G(4)
<u>кон</u>			завершение F(5)
<u>кон</u>			завершение G(7)
<u>кон</u>			завершение F(8)
<u>кон</u>			завершение G(10)
<u>кон</u>			завершение F(11)

Таким образом, символ «звёздочка» будет напечатан 4 раза.

Ответ: 4

## Решение:

- 1) заметим, что каждая функция вызывает другую (это называется косвенная рекурсия), причём только один раз
- 2) вот цепочка вызовов:  
 $F(11) \rightarrow G(10) \rightarrow F(8) \rightarrow G(7) \rightarrow F(5) \rightarrow G(4) \rightarrow F(2) \rightarrow G(1)$
- 3) за один вызов функции G выводится одна звёздочка, внутри функции F звёздочки не выводятся, поэтому за 4 вызова G будет выведено 4 звёздочки
- 4) Ответ: 4.

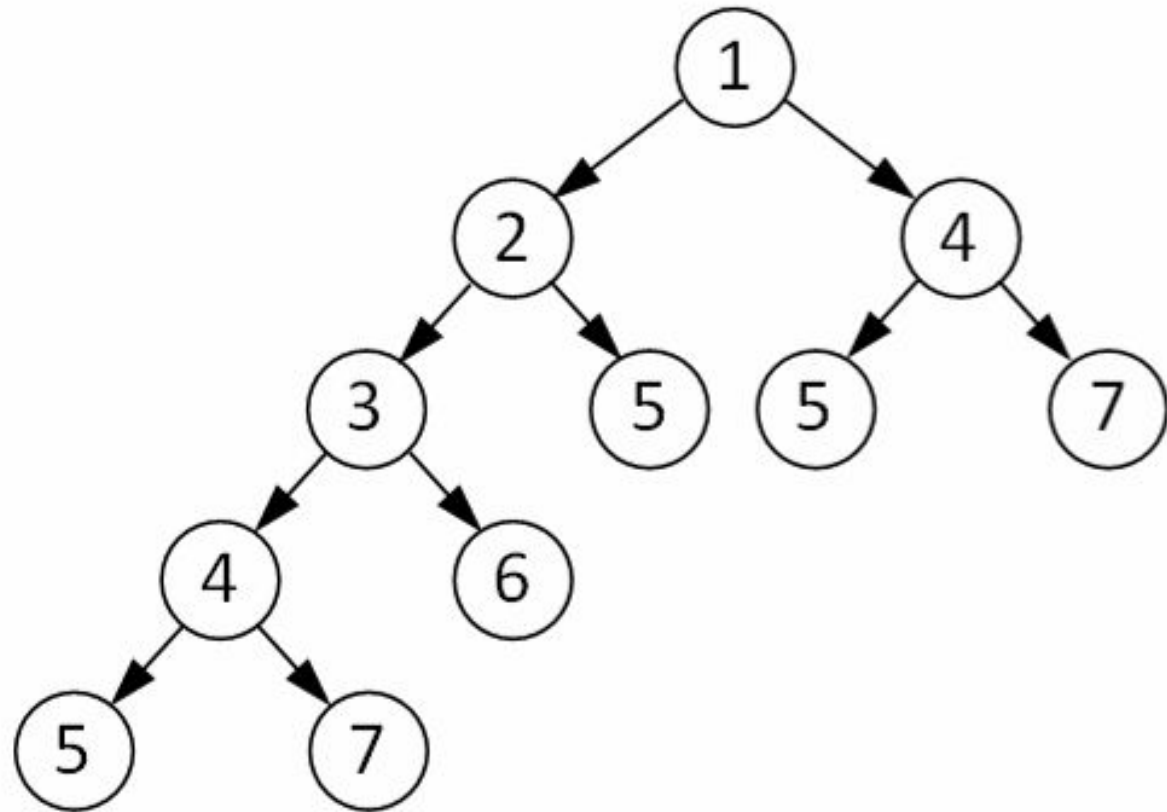


Ещё пример задания:

P-05. Дан рекурсивный алгоритм:

```
procedure F(n: integer);  
begin  
    writeln(n);  
    if n < 5 then begin  
        F(n + 1);  
        F(n + 3)  
    end  
end;
```

*Найдите сумму чисел, которые будут выведены при вызове F(1).*



складывая все эти числа, получаем 49|

ответ: 49.

Решение (вариант 2, подстановка):

- 1) можно обойтись и без дерева, учитывая, что при каждом вызове с  $n < 5$  происходит два рекурсивных вызова; сумму чисел, полученных при вызове  $F(n)$ , обозначим через  $S(n)$ :

$$S(n) = \begin{cases} n + S(n+1) + S(n+3), & n < 5 \\ n, & n \geq 5 \end{cases}$$

- 2) выполняем вычисления:

$$S(1) = 1 + S(2) + S(4)$$

$$S(2) = 2 + S(3) + S(5) = 7 + S(3)$$

$$S(3) = 3 + S(4) + S(6) = 9 + S(4)$$

$$S(4) = 4 + S(5) + S(7) = 16$$

- 3) теперь остаётся вычислить ответ «обратным ходом»:

$$S(3) = 9 + 16 = 25$$

- 4)  $S(2) = 7 + 25 = 32$

$$S(1) = 1 + 32 + 16 = 49$$

- 5) Ответ: **49**.

P-04. Дан рекурсивный алгоритм:

```
procedure F(n: integer);  
begin  
  writeln(n);  
  if n < 6 then begin  
    F(n+2);  
    F(n*3)  
  end  
end;
```

Найдите сумму чисел, которые будут выведены при вызове F(1).

## Решение (вариант 1, метод подстановки):

- 1) сначала определим рекуррентную формулу; обозначим через  $G(n)$  сумму чисел, которая выводится при вызове  $F(n)$
- 2) при  $n \geq 6$  процедура выводит число  $n$  и заканчивает работу без рекурсивных вызовов:

$$G(n) = n \text{ при } n \geq 6$$

- 3) при  $n < 6$  процедура выводит число  $n$  и дважды вызывает сама себя:

$$G(n) = n + G(n+2) + G(3n) \text{ при } n < 6$$

- 4) в результате вызова  $F(1)$  получаем

$$G(1) = 1 + G(3) + G(3)$$

$$G(3) = 3 + G(5) + G(9) = 3 + G(5) + 9$$

$$G(5) = 5 + G(7) + G(15) = 5 + 7 + 15 = 27$$

- 5) используем обратную подстановку:

$$G(3) = 3 + G(5) + 9 = 3 + 27 + 9 = 39$$

$$G(1) = 1 + 2 * G(3) = 79$$

- 6) Ответ: 79.

### Ещё пример задания:

P-01. Алгоритм вычисления значения функции  $F(n)$ , где  $n$  – натуральное число, задан следующими соотношениями:

$$F(1) = 1$$

$$F(n) = F(n-1) * n, \text{ при } n > 1$$

Чему равно значение функции  $F(5)$ ?

В ответе запишите только целое число.

### Решение:

1) используя заданную рекуррентную формулу, находим, что

$$F(5) = F(4) * 5$$

2) применив формулу еще несколько раз, получаем

$$F(5) = F(3) * 4 * 5 = F(2) * 3 * 4 * 5 = F(1) * 2 * 3 * 4 * 5$$

3) мы дошли до базового случая, который останавливает рекурсию, так как определяет значение  $F(1) = 1$

4) окончательно  $F(5) = 1 * 2 * 3 * 4 * 5 = 120$

5) ответ: 120.

## Пример задания:

P-03. Дан рекурсивный алгоритм:

```
procedure F(n: integer);
```

```
begin
```

```
  writeln('*');
```

```
  if n > 0 then begin
```

```
    F(n-2);
```

```
    F(n div 2)
```

```
  end
```

```
end;
```

*Сколько символов "звездочка" будет напечатано на экране при выполнении вызова F(7)?*

Решение (вариант 1, составление полной таблицы):

1) сначала определим рекуррентную формулу; обозначим через  $G(n)$  количество звездочек, которые выводит программа при вызове  $F(n)$

2) из программы видим, что

$$G(n) = 1 \text{ при всех } n \leq 0$$

$$G(n) = 1 + G(n-2) + G(n \text{ div } 2) \text{ при } n > 0$$

3) вспомним, что  $n \text{ div } 2$  – это частное от деления  $n$  на 2

4) по этим формулам заполняем таблицу, начиная с нуля:

$$G(0) = 1$$

$$G(1) = 1 + G(-1) + G(0) = 1 + 1 + 1 = 3$$

$$G(2) = 1 + G(0) + G(1) = 1 + 1 + 3 = 5$$

$$G(3) = 1 + G(1) + G(1) = 1 + 3 + 3 = 7$$

$$G(4) = 1 + G(2) + G(2) = 1 + 5 + 5 = 11$$

$$G(5) = 1 + G(3) + G(2) = 1 + 7 + 5 = 13$$

$$G(6) = 1 + G(4) + G(3) = 1 + 11 + 7 = 19$$

$$G(7) = 1 + G(5) + G(3) = 1 + 13 + 7 = 21$$

n	0	1	2	3	4	5	6	7
G(n)	1	3	5	7	11	13	19	21

5) Ответ: 21.