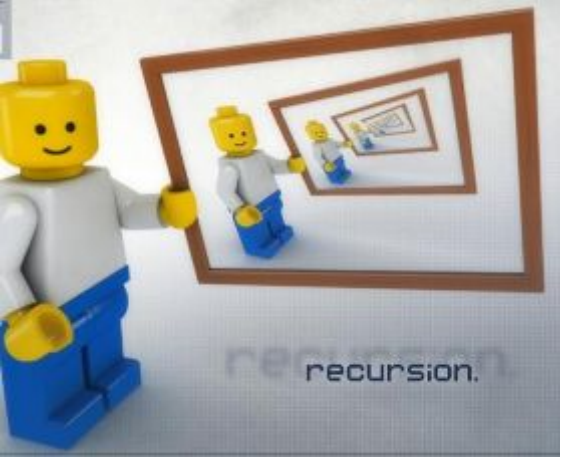


Методика решения рекурсивных алгоритмов (проблемы решения задач данного типа а ЕГЭ)

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:

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

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.

Бейсик

```
SUB F(n)
  print n,
  IF n >= 7 THEN
    F(n - 3)
    F(n - 1)
  END IF
END SUB
```

Python

```
def F(n):
    print(n, end='')
    if n >= 7:
        F(n - 3)
        F(n - 1)
```

Алгоритмический язык

```
алг F(цел n)
нач
  вывод n
  если n >= 7 то
    F(n - 3)
    F(n - 1)
  все
кон
```

Паскаль

```
procedure F(n: integer);
begin
  write(n);
  if n >= 7 then
  begin
    F(n – 3);
    F(n - 1)
  end
end;
```

Си

```
void F(int n) {
  printf("%d", n);
  if (n >= 7) {
    F(n - 3);
    F(n - 1);
  }
}
```

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

Исполним алгоритм для указанного аргумента.

Команда алгоритма	Результат исполнения	Вывод	Примечание
вывод n	вывод «9»	9	
если n >= 7 то	9 ≥7, истина		
F(n - 3)	вызов F(6)		1 уровень рекурсии
вывод n	вывод «6»	6	
если n >= 7 то	6 ≥7, ложь		возврат в F(9)
F(n - 1)	вызов F(8)		1 уровень рекурсии
вывод n	вывод «8»	8	
если n >= 7 то	8 ≥7, истина		
F(n - 3)	вызов F(5)		2 уровень рекурсии
вывод n	вывод «5»	5	
если n >= 7 то	5 ≥7, ложь		возврат в F(8)
F(n - 1)	вызов F(7)		2 уровень рекурсии
вывод n	вывод «7»	7	
если n >= 7 то	7 ≥7, истина		
F(n - 3)	вызов F(4)		3 уровень рекурсии
вывод n	вывод «4»	4	
если n >= 7 то	4 ≥7, ложь		возврат в F(7)
F(n - 1)	вызов F(6)		3 уровень рекурсии
вывод n	вывод «6»	6	
если n >= 7 то	6 ≥7, ложь		возврат в F(7)
кон			возврат в F(8)
кон			возврат в F(9)
кон			завершение алгоритма

Ответ: 9685746

Задание 11.2

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

Алгоритмический язык	**Паскаль**

алг F(цел n)
нач
  если n > 0 то
  G(n - 1)
  все
кон
 	
алг G(цел n)
нач
  вывод "*"
  если n > 1 то
  F(n - 2)
  все
кон
	procedure F(n: integer); forward;
	procedure G(n: integer); forward;
	
	procedure F(n: integer);
	begin
	  if n > 0 then
	  G(n - 1);
	end;
	
	procedure G(n: integer);
	begin
	  writeln('*');
	  if n > 1 then
	  F(n - 2);
	end;

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

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

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

Ответ: 4

Решение:

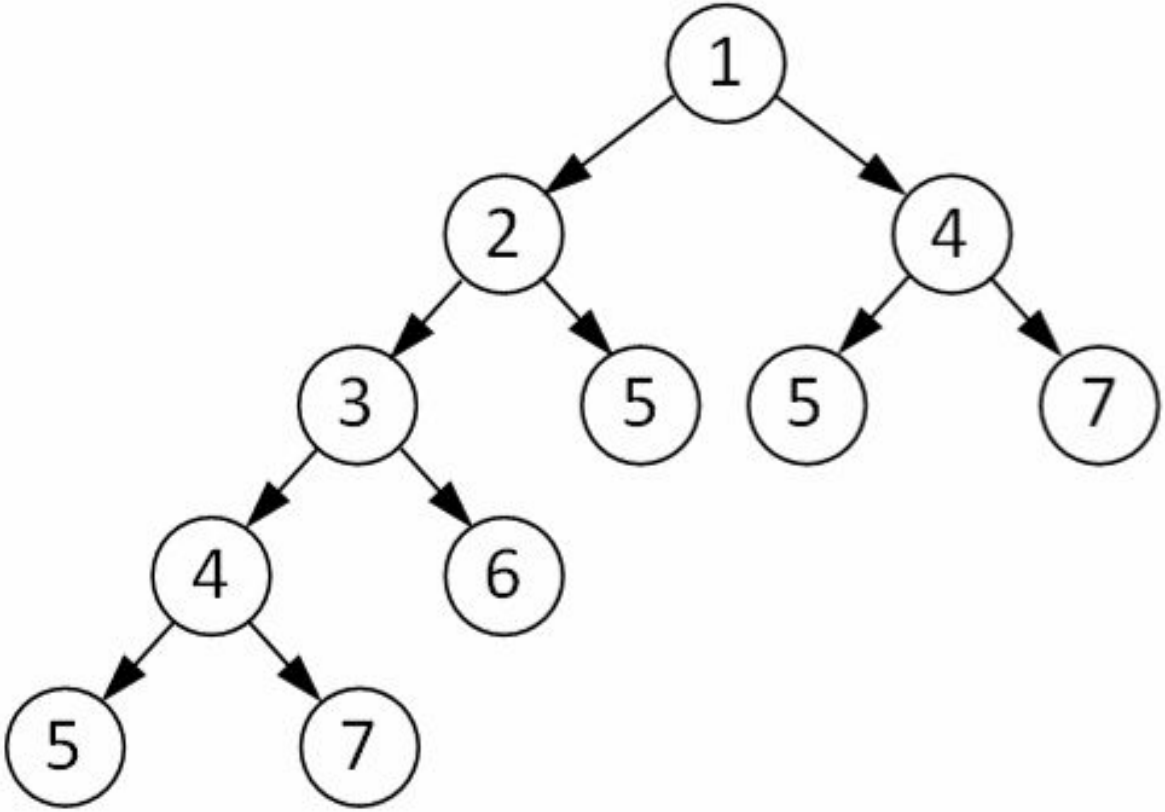
1) заметим, что каждая функция вызывает другую (это называется косвенная рекурсия), причём только один раз
2) вот цепочка вызовов:
   $F(11) \to G(10) \to F(8) \to G(7) \to F(5) \to G(4) \to F(2) \to G(1)$
3) за один вызов функции G выводится одна звёздочка, внутри функции F звездочки не выводятся, поэтому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

Р-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 >= 6 процедура выводит число n и заканчивает работу без рекурсивных вызовов:
   G(n) = n при n >= 6
3) при n < 6 процедура выводит число n и дважды вызывает сама себя:
   G(n) = n + G(n+2) + G(3n) 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:

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

```
F(1) = 1
F(n) = F(n-1) * n, 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:

Р-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 при всех n <= 0
   G(n) = 1 + G(n-2) + G(n div 2) при n > 0
3) вспомним, что `n 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.