DOI: 10.32517/2221-1993-2026-25-1-54-67

Н. И. Заводчикова
Ярославский государственный педагогический университет им. К. Д. Ушинского, г. Ярославль, Россия;
Образовательный комплекс № 20, г. Ярославль, Россия
В. В. Курылёва
Ярославский государственный педагогический университет им. К. Д. Ушинского, г. Ярославль, Россия

ФОРМИРОВАНИЕ У ШКОЛЬНИКОВ ЗНАКОВО-СИМВОЛИЧЕСКИХ СТРУКТУР, СВЯЗАННЫХ С ОПИСАНИЕМ РЕКУРСИВНЫХ ОБЪЕКТОВ*

Аннотация

Федеральные рабочие программы основного и среднего общего образования по информатике на углубленном уровне предполагают знакомство школьников с рекурсивными алгоритмами: учащиеся должны уметь анализировать готовые рекурсивные подпрограммы и разрабатывать собственные. Условием осознанности действий учащихся при решении задачи по программированию является организация их знаково-символической деятельности на этапах перехода от решения задачи для конкретных входных данных к визуальному и словесному описанию алгоритма и от описания алгоритма к командам языка программирования. При разработке рекурсивных алгоритмов посредником между задачей и программой для ее решения служат малознакомые школьникам знаково-символические структуры: рекуррентные формулы и схемы древовидной структуры. Присвоению учащимися этих структур способствуют выполнение ими заданий на словесное описание рекуррентных последовательностей, определение для них рекуррентной формулы и базовых случаев, нахождение некоторого члена рекуррентной последовательности по заданной формуле и начальным значениям, построение дерева рекурсивных вызовов для готовой рекурсивной подпрограммы. В статье приведены примеры заданий указанных типов, а также описана разработанная авторами учебная программная среда, которая позволяет учащимся анализировать особенности работы рекурсивных подпрограмм.

Ключевые слова: рекурсия, знаково-символическая деятельность, семиотика, визуализатор дерева рекурсивных вызовов, учебная программная среда.

Информация об авторах

Заводчикова Надежда Ивановна, канд. пед. наук, доцент, доцент кафедры теории и методики обучения информатике, физико-математический факультет, Ярославский государственный педагогический университет им. К. Д. Ушинского, г. Ярославль, Россия; учитель информатики, средняя школа № 76, образовательный комплекс № 20, г. Ярославль, Россия; *e-mail:* zaw-nadejda@yandex.ru

Курылёва Виктория Владимировна, студентка 5-го курса кафедры теории и методики обучения информатике, физико-математический факультет, Ярославский государственный педагогический университет им. К. Д. Ушинского, г. Ярославль, Россия; *e-mail:* viktoria.kurylyova@yandex.ru

* Материалы к статье можно скачать на сайте ИНФО: http://infojournal.ru/journals/school/school_01-2026/

N. I. Zavodchikova
Yaroslavl State Pedagogical University named after K. D. Ushinsky, Yaroslavl, Russia;
Educational Complex 20, Yaroslavl, Russia
V. V. Kurylyova
Yaroslavl State Pedagogical University named after K. D. Ushinsky, Yaroslavl, Russia

FORMATION OF SIGN-SYMBOLIC STRUCTURES IN SCHOOLCHILDREN RELATED TO THE DESCRIPTION OF RECURSIVE OBJECTS

Abstract

The Federal working program of basic and general secondary education in computer science at an advanced level involves introducing students to recursive algorithms: students should be able to analyze ready-made recursive subroutines and develop their own. A condition for the meaningfulness of students' actions when solving a programming problem is the organization of their sign-symbolic activity at the stages of transition from solving a problem for specific input data to a visual and verbal description of the algorithm, and from the description of the algorithm to the commands of a programming language. When developing recursive algorithms, the intermediaries between the problem and the program for its solution are sign-symbolic structures that are unfamiliar to students: recurrence relations and tree-structured diagrams. The appropriation of these structures by students is facilitated by their performance of tasks on the verbal description of recurrent sequences, the determination of recurrence formulas and base cases for them, finding a specific term of a recurrent sequence given a formula and initial values, and constructing a recursion tree for a ready-made recursive

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

subroutine. The article provides examples of tasks of these types and also describes an educational software environment developed by the authors that allows students to analyze the features of the operation of recursive subroutines.

Keywords: recursion, sign-symbolic activity, semiotics, recursion tree visualizer, educational software environment.

Information about the authors

Nadejda I. Zavodchikova, Candidate of Sciences (Education), Docent, Associate Professor at the Department of Theory and Methods of Teaching Informatics, Faculty of Physics and Mathematics, Yaroslavl State Pedagogical University named after K. D. Ushinsky, Yaroslavl, Russia; an informatics teacher, School 76, Educational Complex 20, Yaroslavl, Russia; *e-mail*: zaw-nadejda@yandex.ru

Viktoria V. Kurylyova, 5th-year student at the Department of Theory and Methods of Teaching Informatics, Faculty of Physics and Mathematics, Yaroslavl State Pedagogical University named after K. D. Ushinsky, Yaroslavl, Russia; *e-mail:* viktoria.kurylyova@yandex.ru

1. Введение

Рекурсия — это мощный инструмент, который позволяет решать задачи, разбивая их на более мелкие подзадачи. Рекурсивный подход используется, если решение задачи имеет древовидную структуру и предусматривает поиск с возвратом, но не предполагает слишком большую глубину рекурсии, способную спровоцировать переполнение стека [16].

Федеральная рабочая программа основного общего образования по информатике на углубленном уровне определяет содержание темы следующим образом: «Рекурсия. Рекурсивные подпрограммы (процедуры, функции). Условие окончания рекурсии (базовые случаи). Применение рекурсии для перебора вариантов» [18]. В рамках углубленного курса информатики X—XI классов рассматриваются следующие вопросы: рекурсия, рекурсивные объекты (фракталы), рекурсивные процедуры и функции, использование стека для организации рекурсивных вызовов [19].

При изучении рекурсии учащиеся, как правило, знакомятся с некоторым количеством известных рекурсивных алгоритмов, таких как вычисление факториала, алгоритм Евклида, нахождение чисел Фибоначчи или решение задачи о ханойских башнях [1, 2, 4, 9, 11, 14]. Обсуждение рекурсивных алгоритмов сопровождается построением дерева или изображением стека рекурсивных вызовов. Затем школьникам предлагается ряд задач на анализ готовых алгоритмов и написание собственных.

Умение анализировать и реализовывать рекурсивные алгоритмы проверяется и в рамках единого государственного экзамена по информатике [3]. Задание 16 ЕГЭ, согласно спецификации, проверяет выполнение предметного требования к результатам освоения основной образовательной программы «вычисление рекуррентных выражений». Программное решение заданий 19–21 и 23 предполагает реализацию рекурсивного алгоритма. В последние годы формат указанных заданий почти не менялся и, как неоднократно отмечалось, большое количество школьников заучивают готовые шаблоны программ без должного их понимания [15].

Многие исследователи относят тему «Рекурсия» к наиболее сложным вопросам раздела «Алгоритмы и программирование» курса информатики, отмечая необходимость снижения высокого уровня абстракции с помощью использования графических исполнителей [12], проведения аналогии между циклическим и рекурсивным подходами [13], визуализации дерева вызовов рекурсивной функции [15].

Как было отмечено нами ранее в работах [5–7], важным условием осознанности действий учащихся при решении задачи по программированию является организация их знаково-символической деятельности на этапах перехода от решения задачи для конкретных входных данных к визуальному и словесному описанию алгоритма и от описания алгоритма — к командам языка программирования.

При решении задач, предполагающих реализацию циклического алгоритма, на этапе моделирования действий с частными случаями входных данных используются трассировочные таблицы, на этапе схематизации — блок-схемы [5–7]. При разработке рекурсивных алгоритмов посредником между задачей и программой для ее решения служат малознакомые школьникам знаково-символические средства: рекуррентные формулы и схемы древовидной структуры.

Например, при решении задачи 23 ЕГЭ по информатике переход от работы с конкретными входными данными к схеме алгоритма может реализовываться с помощью построения дерева перечисления всех программ, для которых при исходном числе N результатом является число K. Словесное описание алгоритмов построения дерева и нахождения количества программ по дереву позволяет записать рекуррентную формулу и перевести шаги алгоритма в команды языка программирования.

До изучения рекурсии школьники мало сталкивались с необходимостью решения задач с помощью построения деревьев и совсем не сталкивались с программами, процесс выполнения которых нельзя изобразить с помощью трассировочной таблицы. Также школьники почти не встречались с самоподобными объектами, а рекуррентные формулы видели только на уроках математики при изучении темы «Прогрессии». Очевидно, что для самостоятельного решения задач, предполагающих реализацию рекурсивного алгоритма, указанные знаково-символические средства должны быть присвоены учащимися.

По мнению А. С. Турчина, процесс присвоения сознанием новых знаковых систем должен строиться в соответствии с представлениями о предназначении «зоны ближайшего развития» [17]. С. В. Маланов отмечает, что знаково-символические схемы, формирование которых происходит управляемо, могут стать рефлексивными, осознаваться и выступать в качестве эталона для верного выполнения умственного действия [8].

Для управления процессом освоения учащимися перечисленных выше знаково-символических средств целесообразно предложить им задания, в которых рекуррентные формулы выступают в качестве заместителей для рекуррентных последовательностей, а деревья — для последовательности вызовов рекурсивной функции.

ISSN 2221-1993 • **Informatics in School** • 2026 • Volume 25 No 1

Во разделе 2 данной статьи приведен набор упражнений, позволяющий сформировать у школьников необходимые знаково-символические структуры:

- задания, направленные на определение рекуррентной формулы и базовых случаев для заданных числовых последовательностей и обратные им, предполагающие нахождение n-ого члена последовательности, заданной своими первыми членами и рекуррентными формулами;
- упражнения на анализ готовых рекурсивных подпрограмм и построение дерева рекурсивных вызовов.

2. Последовательность заданий для формирования у учащихся знаково-символических структур, связанных с описанием рекурсивных объектов

Задание 1.

Угадайте следующий член последовательности, сформулируйте правила, по которому он строится:

а) 1, 2, 3, 4, 5, 6, 7, ...
б) 1, 2, 4, 8, 16, 32, ...
в) 1, 1, 2, 3, 5, 8, 13, ...
г) 1, 2, 6, 24, 120, ...

Решение.

а) 8. Первый член последовательности равен единице, каждый следующий получается из предыдущего прибавлением единицы.
б) 64. Первый член последовательности равен единице, каждый следующий получается из предыдущего умножением на два.
в) 21. Первый и второй члены последовательности равны единице, каждый следующий равен сумме двух предыдущих.
г) 720. Первый член последовательности равен единице, каждый следующий член последовательности равен произведению своего номера и предыдущего члена последовательности.

Задание 2.

Формула, которая позволяет выразить следующий член числовой последовательности через предыдущие, называется ***рекуррентной*** формулой. Если за $f(n)$ обозначить n-й член последовательности, то в задании (а) n-й и $(n-1)$-й члены последовательности будут связаны формулой $f(n) = f(n-1) + 1$, при этом $f(1) = 1$.

Запишите рекуррентные формулы для правил, описанных в предыдущем задании.

Решение.

а) $f(1) = 1$; $f(n) = f(n-1) + 1$
б) $f(1) = 1$; $f(n) = f(n-1) \cdot 2$
в) $f(1) = 1$, $f(2) = 1$; $f(n) = f(n-1) + f(n-2)$
г) $f(1) = 1$, $f(n) = f(n-1) \cdot n$

Задание 3.

Последовательность $f(n)$ задана следующими соотношениями:

$f(1) = 1$; $f(n) = 2 \cdot f(n-1) + n + 3$, если $n > 1$.

Найдите первые шесть членов последовательности.

Решение.

$f(1) = 1$;
$f(2) = 2 \cdot f(1) + 2 + 3 = 7$;
$f(3) = 2 \cdot f(2) + 3 + 3 = 20$;
$f(4) = 2 \cdot f(3) + 4 + 3 = 47$;
$f(5) = 2 \cdot f(4) + 5 + 3 = 102$;
$f(6) = 2 \cdot f(5) + 6 + 3 = 213$.

Ответ: 1, 7, 20, 47, 102, 213.

Задание 4.

Заполните пропуски в таблице (табл. 1), где каждая строка должна содержать начало рекуррентной последовательности, правило, описывающее ее построение, и формулу, обозначающую это правило.

Примерный вариант ответа представлен в таблице 2.

Таблица 1

Таблица к заданию 4

Последовательность	Правило	Формула
3, 5, 7, 9, ...	Первый член последовательности равен 3, каждый следующий получается из предыдущего прибавлением 2	$f(1) = 3$, $f(n) = f(n-1) + 2$
	Первый член последовательности равен 2, каждый следующий получается из предыдущего умножением на 3	
		$f(1) = 1$, $f(n) = 5 \cdot f(n-1) + n$
1, 1, 1, 3, 5, 9, 17, 31, ...		
		$f(0) = 0$, $f(1) = 1$ $f(n) = f(n-1) + 2 \cdot f(n-2)$
10, 21, 32, 43, 54, ...		

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

Таблица 2

Примерный вариант ответа к заданию 4

Последовательность	Правило	Формула
3, 5, 7, 9, ...	Первый член последовательности равен 3, каждый следующий получается из предыдущего прибавлением 2	$f(1) = 3$, $f(n) = f(n-1) + 2$
2, 6, 18, 54, 162, ...	Первый член последовательности равен 2, каждый следующий получается из предыдущего умножением на 3	$f(1) = 2$, $f(n) = 3 \cdot f(n-1)$
1, 7, 38, 194, 975, ...	Первый член последовательности равен 1, каждый следующий получается из предыдущего умножением на 5 и прибавлением к результату номера члена последовательности	$f(1) = 1$, $f(n) = 5 \cdot f(n-1) + n$
1, 1, 1, 3, 5, 9, 17, 31, ...	Первый, второй и третий члены последовательности равны единице, каждый следующий равен сумме трех предыдущих	$f(1) = 1$, $f(2) = 1$, $f(3) = 1$, $f(n) = f(n-1) + f(n-2) + f(n-3)$
0, 1, 1, 3, 5, 11, ...	Нулевой член последовательности равен нулю, первый — единице, каждый следующий равен сумме предыдущего и удвоенного члена, стоящего перед предыдущим	$f(0) = 0$, $f(1) = 1$ $f(n) = f(n-1) + 2 \cdot f(n-2)$
10, 21, 32, 43, 54, ...	Первый член последовательности равен 10, каждый следующий получается из предыдущего прибавлением 11	$f(1) = 10$ $f(n) = f(n-1) + 11$

Задание 5.

Последовательность $f(n)$ задана следующими соотношениями:

$f(n) = 2$ при $n \le 1$;

$f(n) = f(n-1) + f(n-2) + 2n + 4$, если $n > 1$.

Найдите значение пятого члена последовательности.

Решение.

Рекурсивный спуск:

$f(5) = f(4) + f(3) + 2 \cdot 5 + 4 =$

$f(4) = f(3) + f(2) + 2 \cdot 4 + 4 =$

$f(3) = f(2) + f(1) + 2 \cdot 3 + 4 =$

$f(2) = f(1) + f(0) + 2 \cdot 2 + 4 =$

$f(1) = 2$

$f(0) = 2$

Рекурсивный подъем:

$f(0) = 2$

$f(1) = 2$

$f(2) = f(1) + f(0) + 2 \cdot 2 + 4 = 12$

$f(3) = f(2) + f(1) + 2 \cdot 3 + 4 = 24$

$f(4) = f(3) + f(2) + 2 \cdot 4 + 4 = 48$

$f(5) = f(4) + f(3) + 2 \cdot 5 + 4 = 86$

Ответ: 86.

Задание 6.

Последовательность $f(n)$ задана следующими соотношениями:

$f(n) = n$, если $n < 10$;

$f(n) = 5 \cdot n + f(n-1)$, если $n \ge 10$.

Найдите значение выражения $f(2026) - f(2024)$.

Решение.

$f(2026) = 5 \cdot 2026 + f(2025) =$

$= 5 \cdot 2026 + (5 \cdot 2025 + f(2024))$;

$f(2026) - f(2024) = 5 \cdot 2026 + (5 \cdot 2025 + f(2024)) -$

$- f(2024) = 5 \cdot 2026 + 5 \cdot 2025 = 5 \cdot (2026 + 2025) =$

$= 5 \cdot 4051 = 20\,255$.

Ответ: 20 255.

Задание 7.

Заполните пропуски в программе, предназначенной для решения задачи.

Задача:

Алгоритм вычисления функции $f(n)$ задан следующими соотношениями:

$f(n) = 1$ при $n \le 1$;

$f(n) = 2 \cdot f(n-1) + f(n-2) + 3$, если $n > 1$.

Чему равно значение функции $f(100)$?

Программа:

```
def f(n):
    if n<=1:
        return __
    else:
        return 2*f(__)+_______+3
print(f(__))
```

Задание 8.

Нарисуйте дерево рекурсивных вызовов функции из предыдущего задания для значения аргумента, равного 5. Корень дерева соответствует первому вызову рекурсивной функции, точки ветвления — рекурсивным случаям, а листья — базовым случаям, в которых рекурсивные вызовы перестают выполняться. Как меняется стек рекурсивных вызовов?

Решение.

На рисунке 1 представлено дерево рекурсивных вызовов функции $f(5)$, на рисунке 2 — процесс изменения стека рекурсивных вызовов.

ISSN 2221-1993 • **Informatics in School** • 2026 • Volume 25 No 1

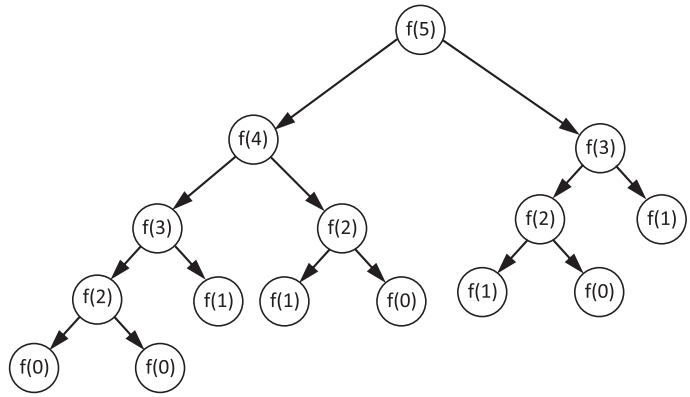

Рис. 1. Решение задания 8. Дерево рекурсивных вызовов

Рис. 2. Решение задания 8. Стек рекурсивных вызовов

Задание 9.

Для следующих программ, вызывающих рекурсивные функции, нарисуйте дерево рекурсивных вызовов и определите, что выведет программа.

а)
```
def f(n):
  if n > 2:
   return f(n-1)+f(n-2)
  else:
   return n
print(f(5))
```

б)
```
def f(n):
  print(n)
  if n >= 3:
    f(n - 1)
    f(n - 3)
f(5)
```

в)
```
def f(n):
  print(n)
  if n < 5:
    print(n)
    f(n + 3)
f(1)
```

г)
```
def f(n):
  print(n)
  if n < 5:
    f(n+3)
    print(n)
    f(n*3)
f(1)
```

д)
```
def f(n):
  print(n)
  if n < 5:
    f(n+3)
    f(n*3)
    f(n*2)
    print(n)
f(1)
```

е)
```
def f(n):
  if n > 0:
    f(n - 3)
    print(n)
    f(n // 3)
f(9)
```

ж)
```
def f(n):
  if n<3:
    return 1
  return f(n+1)+f(n+2)
print(f(5))
```

Рассмотрим назначение каждого из приведенных выше заданий.

В результате выполнения *задания 1* у учащихся формируется представление о рекурсивных объектах. После его решения можно ввести определение, данное в учебнике [14]: «Рекурсия — это способ определения множества объектов через само это множество на основе простых базовых случаев». При обсуждении необходимо обратить внимание на два аспекта рекурсивного определения объектов: наличие базового случая и правила построения объекта через другие объекты множества.

В *задании 2* школьникам предлагается для более формального описания правил построения членов последовательности использовать обозначения, вводится понятие рекуррентной формулы.

Выполняя задания 1 и 2, учащиеся формулируют правила построения рекурсивных объектов и вводят для этих правил обозначения, определяют знаково-символические средства, которые замещают реальные объекты. *Задание 3* направлено на применение новых знаково-символических средств: по заданным формулами правилам учащиеся восстанавливают рекурсивные объекты.

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

Задание 4 направлено на осуществление учениками тех же действий сопоставления объекта, его словесного описания и знакового обозначения, что и при выполнении предыдущих заданий, но уже в свернутом виде.

При выполнении *задания 5* целесообразно продемонстрировать учащимся метод отложенных вычислений, т. е. сначала выразить пятый член последовательности через четвертый, четвертый — через третий и т. д., пока не достигнем базового случая. После этого закончить выполнение вычислений сначала для второго члена последовательности, затем для третьего и т. д. до искомого.

Задание 6 имеет более высокий уровень абстракции — учащиеся вынуждены оперировать только знаковыми обозначениями для членов рекуррентной последовательности, не опираясь на их конкретные числовые значения.

При выполнении *задания 7* необходимо обсудить со школьниками, что рекуррентные формулы позволяют создавать рекурсивные функции, т. е. функции, которые вызывают сами себя напрямую или через другие функции [2, 14]. Для того чтобы написать рекурсивную функцию для решения некоторой задачи, обычно нужно определить рекуррентную формулу и условие останова (базовые случаи), т. е. значения параметров функции, для которых значение функции известно или вычисляется без рекурсивных вызовов.

Задание 8 позволяет обсудить с учащимися особенности работы рекурсивной подпрограммы и ввести знаково-символическую структуру для изображения последовательности рекурсивных вызовов. При выполнении этого задания необходимо обратить внимание учащихся на то, что рекурсивные подпрограммы работают на основе стека вызовов, где каждый новый вызов помещается в стек до тех пор, пока не будет достигнут базовый случай — условие, при котором рекурсивные вызовы прекращаются. Стек вызовов подпрограммы, помимо прочего, содержит информацию о строке кода, с которой должно продолжиться выполнение программы после завершения функции. После возврата из функции соответствующий объект извлекается из стека. После завершения работы рекурсивной функции стек вызовов опустеет. Построение дерева и изображение состояния стека рекурсивных вызовов (см. рис. 1, 2) необходимо производить одновременно. Также нужно обратить внимание учащихся на тот факт, что новый вызов подпрограммы, например f(3), запускает ее выполнение заново, даже если она уже вызывалась ранее.

Набор упражнений в *задании 9* помогает учащимся понять, как работают рекурсивные подпрограммы, а учителю — оценить уровень усвоения изучаемого материала школьниками. В упражнениях этого задания представлены как функции, возвращающие результат, так и подпрограммы, которые выводят значения аргумента на экран без возвращения результата функции. Различное положение оператора print в рекурсивных подпрограммах позволяет обратить внимание школьников на место (адрес, точку) возврата, с которого должно продолжиться выполнение программы после возврата из подпрограммы. Выполнение последнего упражнения в этом задании дает возможность обсудить со школьниками причины возникновения бесконечного количества рекурсивных вызовов.

3. Программа для визуализации процесса выполнения рекурсивных подпрограмм

Количество упражнений, подобных представленным в задании 9, достаточное для усвоения изучаемого материала, может быть различным и зависит от уровня подготовки и развития учащихся. Большой набор подобных заданий можно найти в материалах для подготовки к решению задания 11 докомпьютерной версии ЕГЭ по информатике [10].

При фронтальном решении на доске упражнений этого вида многие школьники получают результат решения в готовом виде и воспринимают процесс вызова рекурсивных подпрограмм симультанно, в результате чего происходит неверное установление последовательно-временных связей. Чтобы организовать индивидуальную работу учащихся с подобными заданиями, была разработана программа, в которой построение дерева последовательности вызовов рекурсивной под-

Рис. 3. Главное меню программы для визуализации выполнения рекурсивных подпрограмм

программы происходит последовательно и каждый шаг сопровождается комментариями.

Рассмотрим интерфейс программы. На рисунке 3 представлено главное меню, которое позволяет выбрать один из двух видов упражнений: задания на анализ процесса выполнения функции, реализующей некоторую рекуррентную последовательность, или задания на анализ подпрограммы, которая не возвращает результат, а выводит на экран значения переменных.

Выбрав тип задания, пользователь переходит к новому окну (рис. 4, 5), которое содержит поле для редактирования шаблона программы, область для изображения дерева рекурсивных вызовов, управляющие кнопки, область комментариев работы программы, область отображения изменений в стеке рекурсивных вызовов для первого шаблона и область вывода результатов выполнения оператора print() для второго шаблона.

Ученик имеет возможность частично изменять выбранный шаблон: заполнять условия продолжения и окончания рекурсивных вызовов, значение аргумента в вызовах подпрограммы; комментировать строки, содержащие оператор print(), нажав на соответствующую строку.

Программа позволяет осуществлять пошаговое выполнение кода, при этом подсвечиваются выполняемая строка кода и вершина дерева, соответствующая параметру функции, с которым работает программа на данном шаге. Каждый шаг сопровождается комментариями, также можно увидеть изменения стека рекурсивных вызовов (рис. 6–19).

Пользователь имеет возможность посмотреть полный список действий, которые были выполнены в ходе работы программы (рис. 20, 21).

Работа с визуализатором позволяет школьникам установить соответствие между реальным процессом выполнения рекурсивной подпрограммы, речевым описанием этого процесса и знаково-символическим заместителем для последовательности рекурсивных вызовов. Отсутствие такого соответствия ведет к возникновению затруднений в понимании сущности рекурсивных алгоритмов и их реализации.

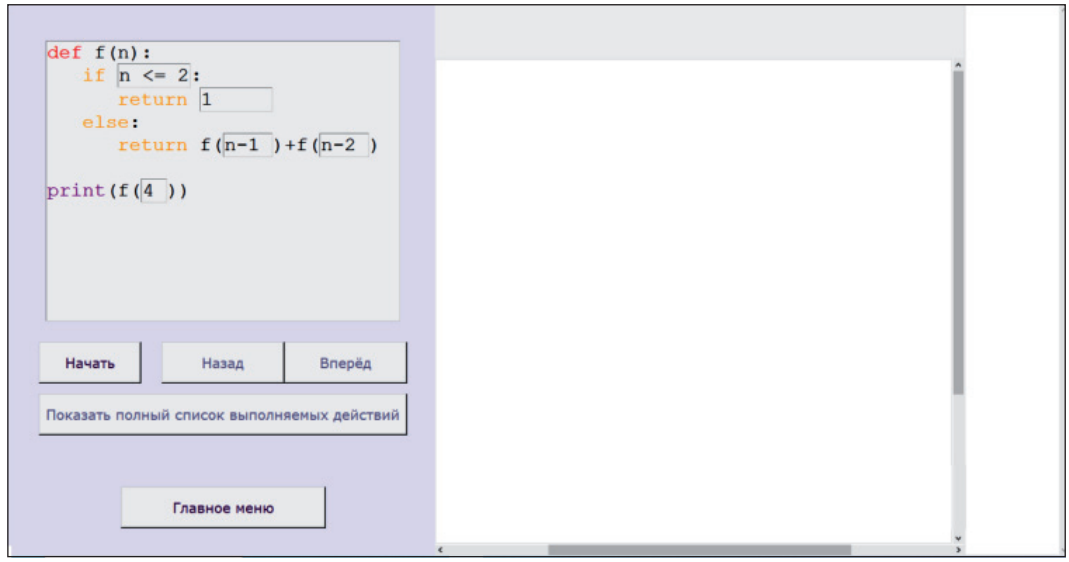

Рис. 4. Интерфейс программы для визуализации выполнения рекурсивных подпрограмм, шаблон 1

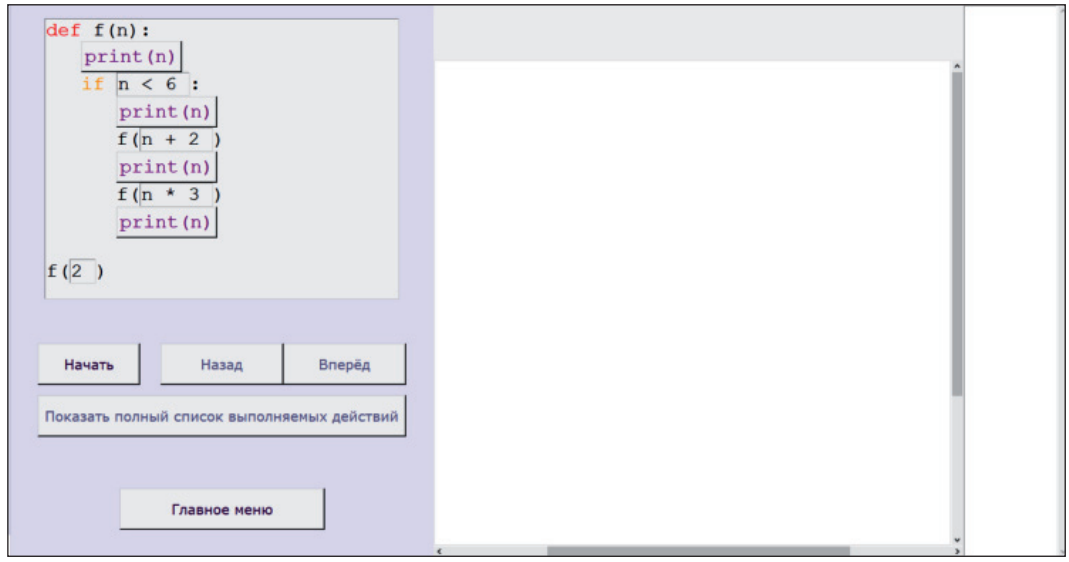

Рис. 5. Интерфейс программы для визуализации выполнения рекурсивных подпрограмм, шаблон 2

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

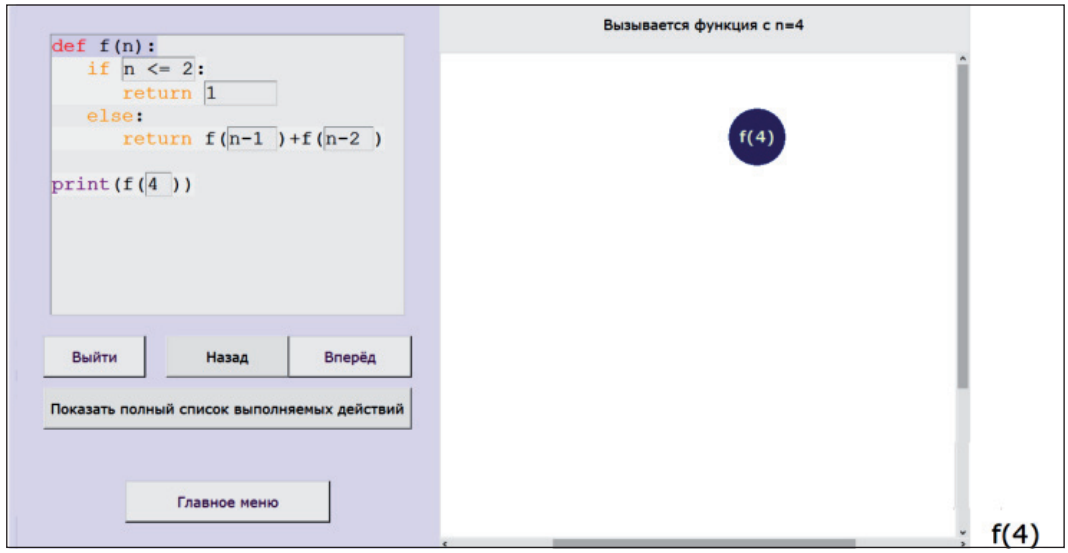

Рис. 6. Пошаговое выполнение рекурсивной подпрограммы, шаг 1

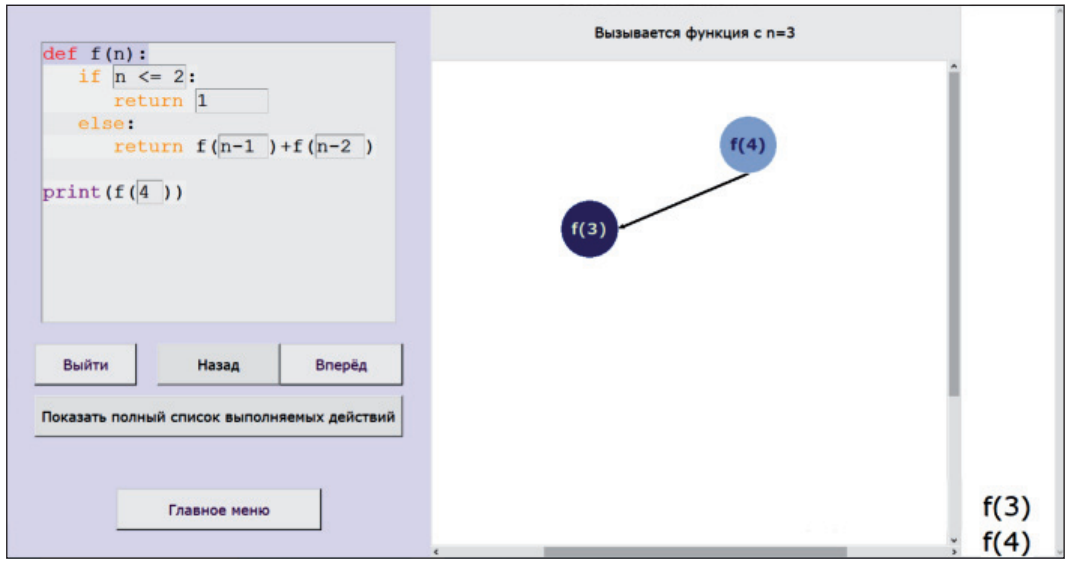

Рис. 7. Пошаговое выполнение рекурсивной подпрограммы, шаг 2

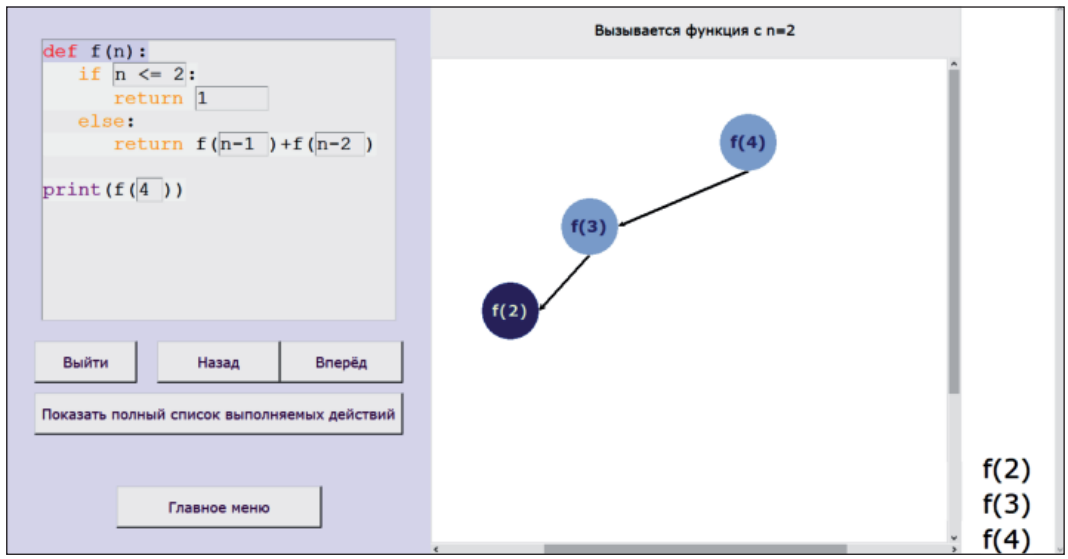

Рис. 8. Пошаговое выполнение рекурсивной подпрограммы, шаг 3

ISSN 2221-1993 • **Informatics in School** • 2026 • Volume 25 No 1

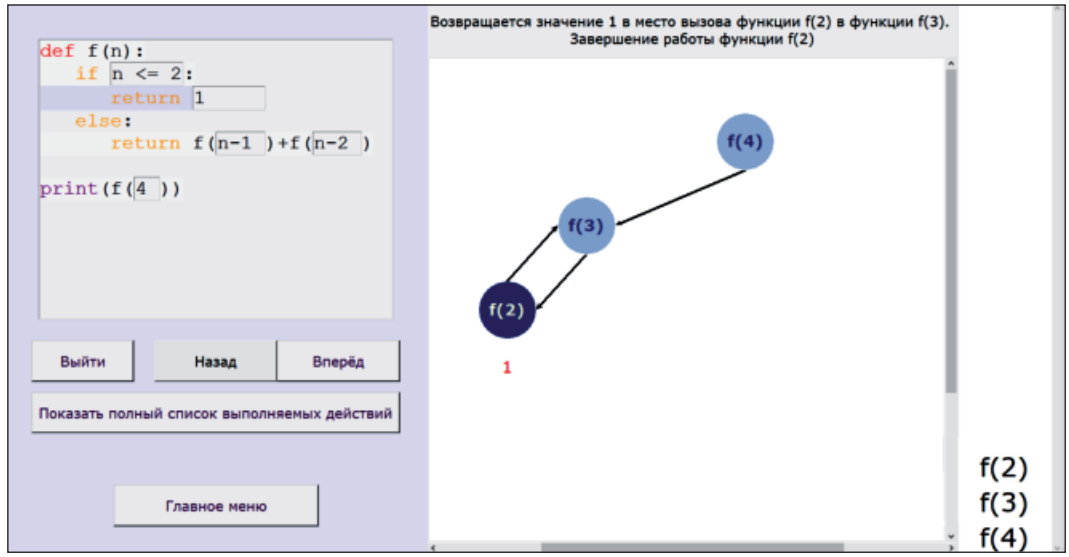

Рис. 9. Пошаговое выполнение рекурсивной подпрограммы, шаг 4

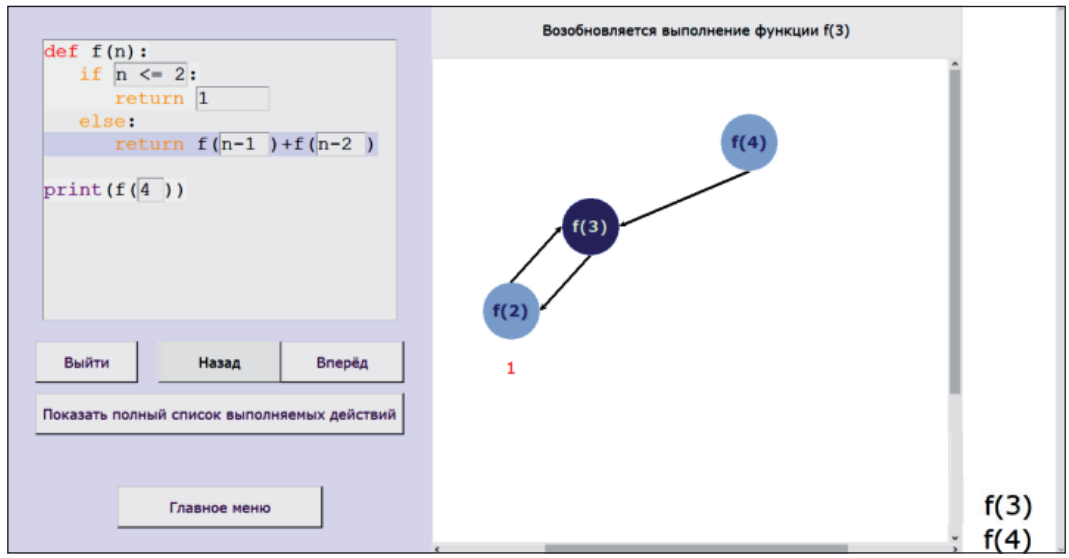

Рис. 10. Пошаговое выполнение рекурсивной подпрограммы, шаг 5

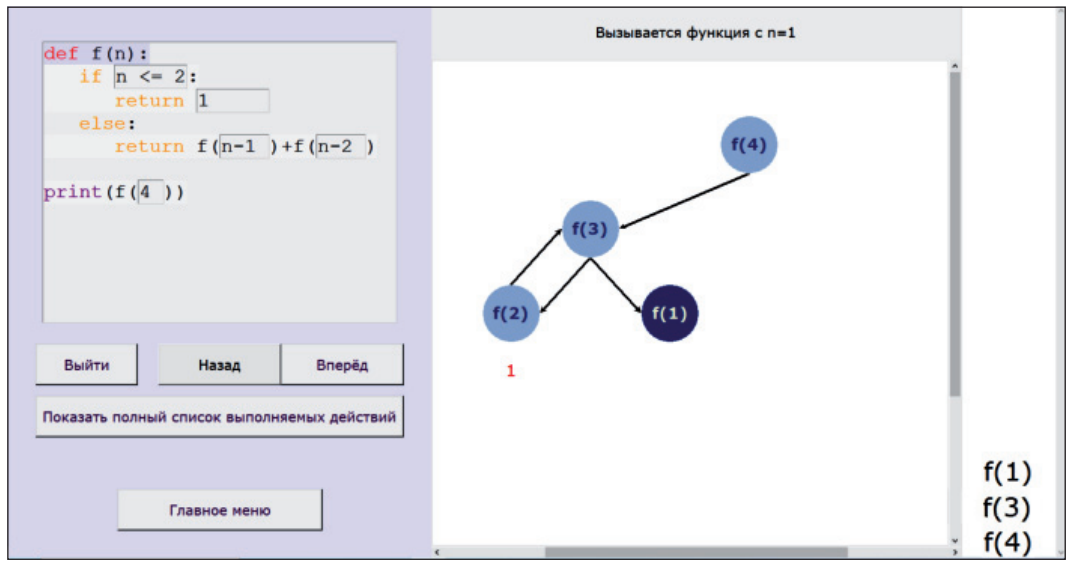

Рис. 11. Пошаговое выполнение рекурсивной подпрограммы, шаг 6

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

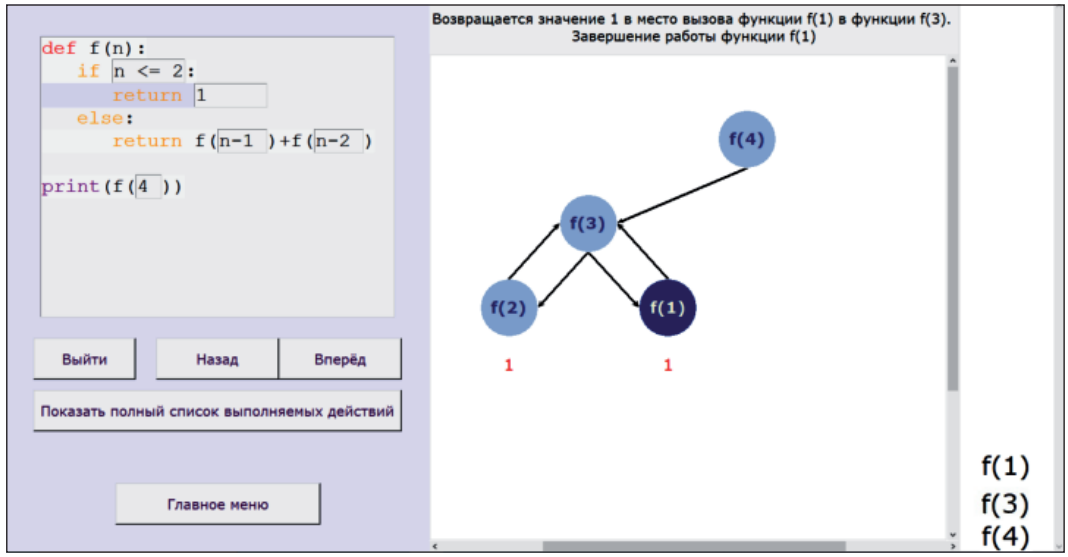

Рис. 12. Пошаговое выполнение рекурсивной подпрограммы, шаг 7

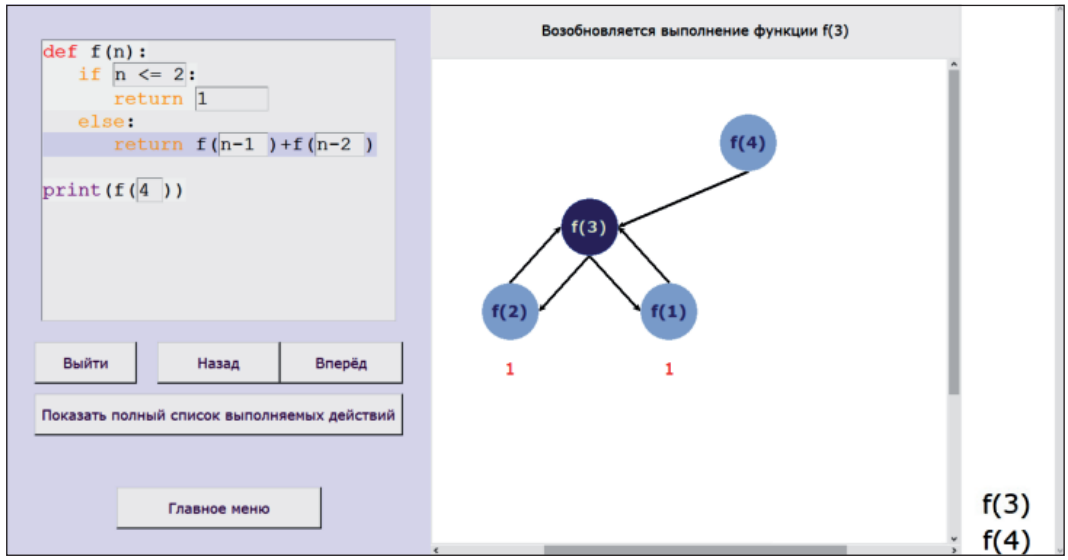

Рис. 13. Пошаговое выполнение рекурсивной подпрограммы, шаг 8

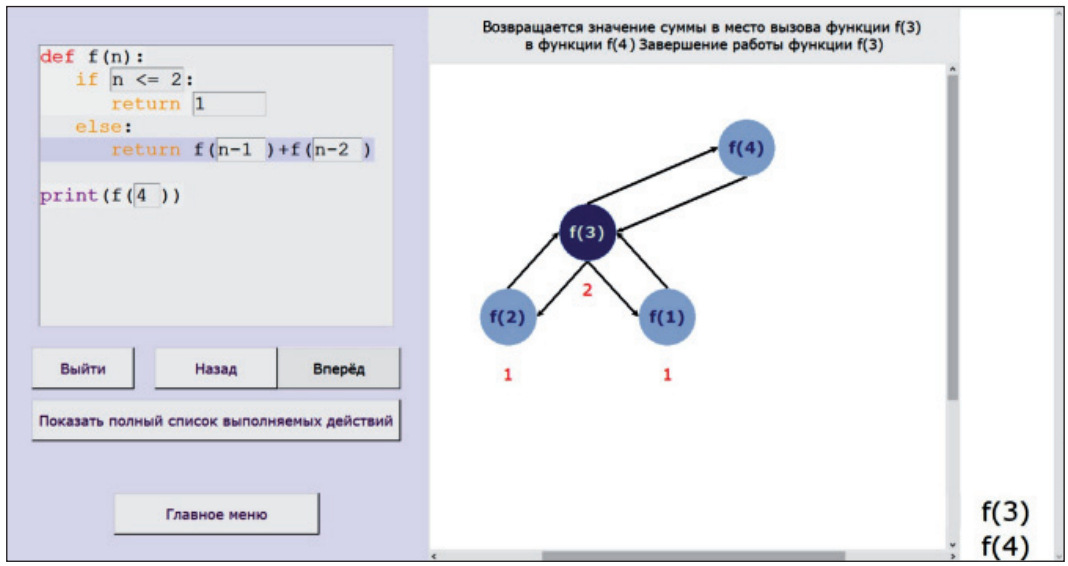

Рис. 14. Пошаговое выполнение рекурсивной подпрограммы, шаг 9

ISSN 2221-1993 • **Informatics in School** • 2026 • Volume 25 No 1

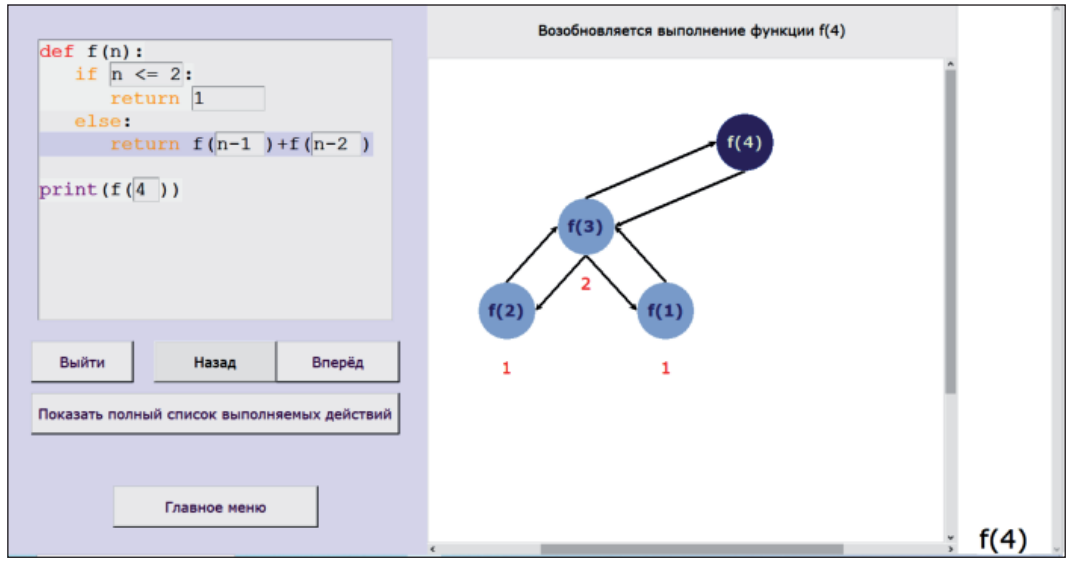

Рис. 15. Пошаговое выполнение рекурсивной подпрограммы, шаг 10

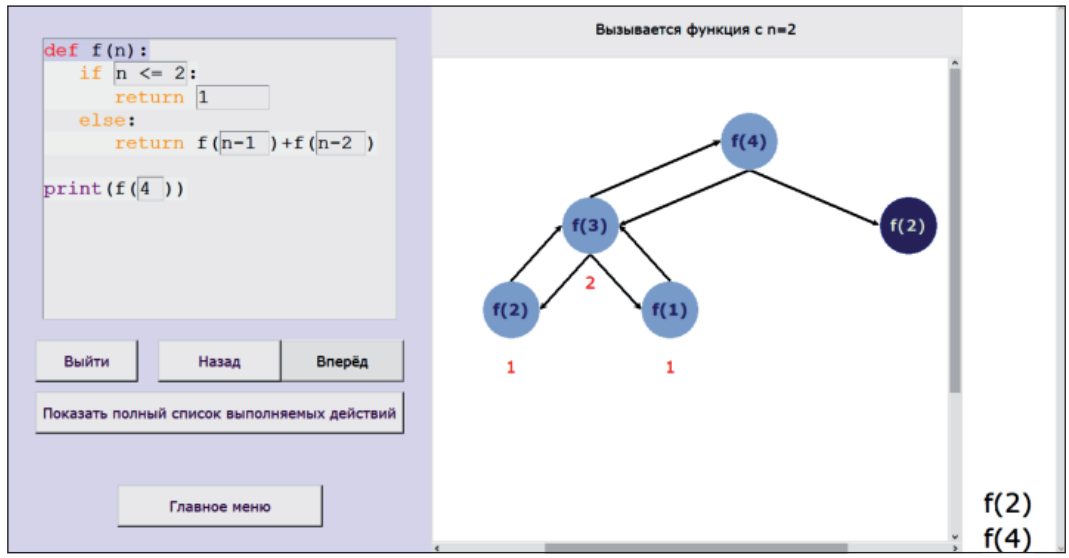

Рис. 16. Пошаговое выполнение рекурсивной подпрограммы, шаг 11

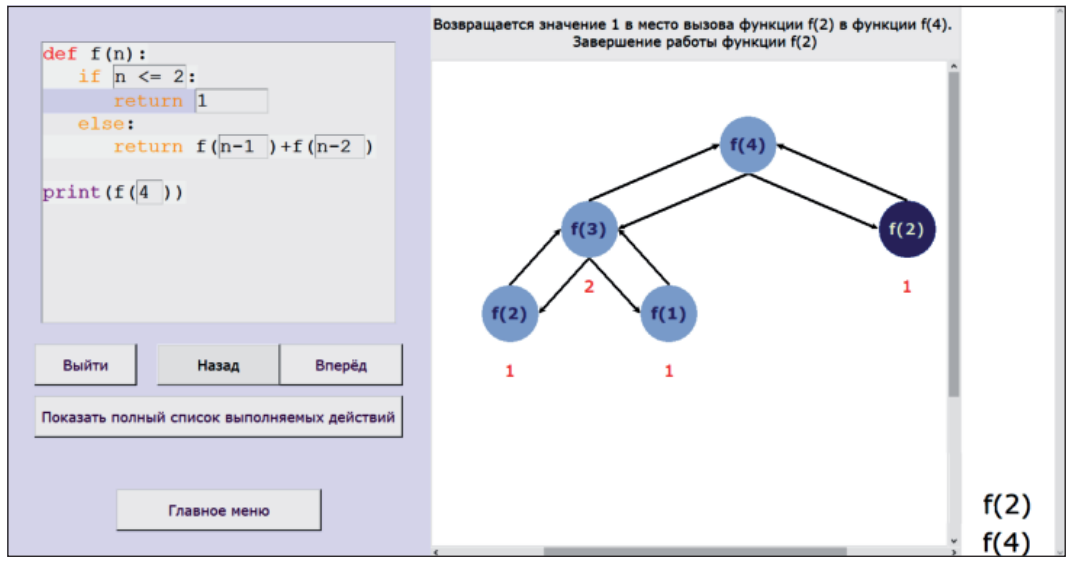

Рис. 17. Пошаговое выполнение рекурсивной подпрограммы, шаг 12

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

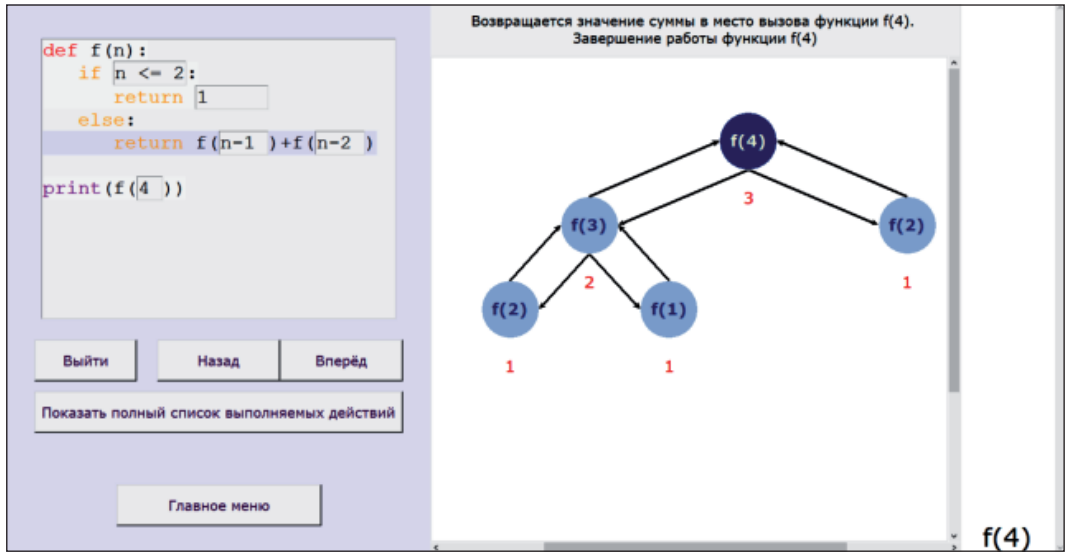

Рис. 18. Пошаговое выполнение рекурсивной подпрограммы, шаг 13

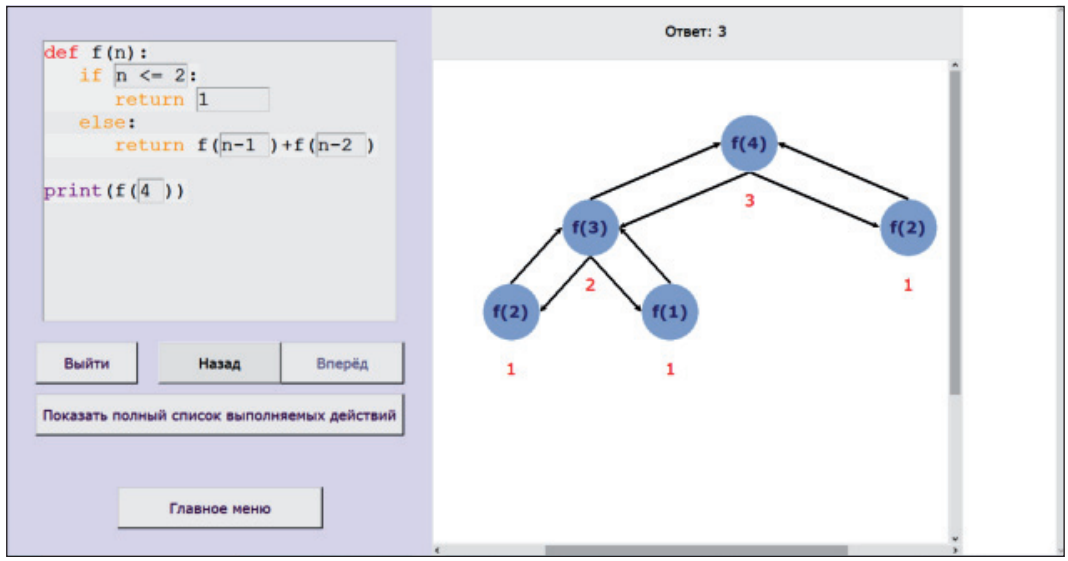

Рис. 19. Пошаговое выполнение рекурсивной подпрограммы, шаг 14

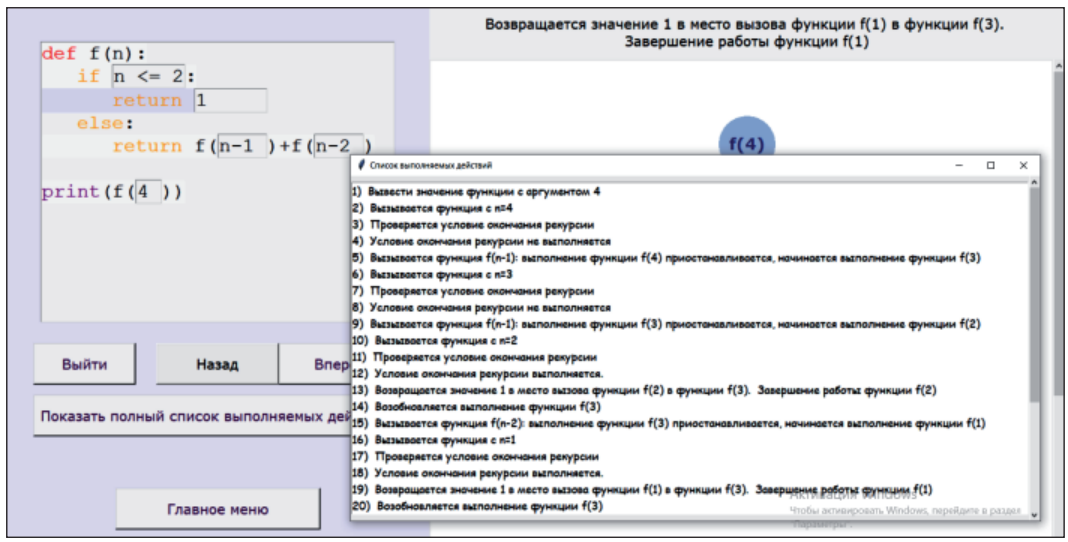

Рис. 20. Список действий, которые были выполнены в ходе работы программы, шаблон 1

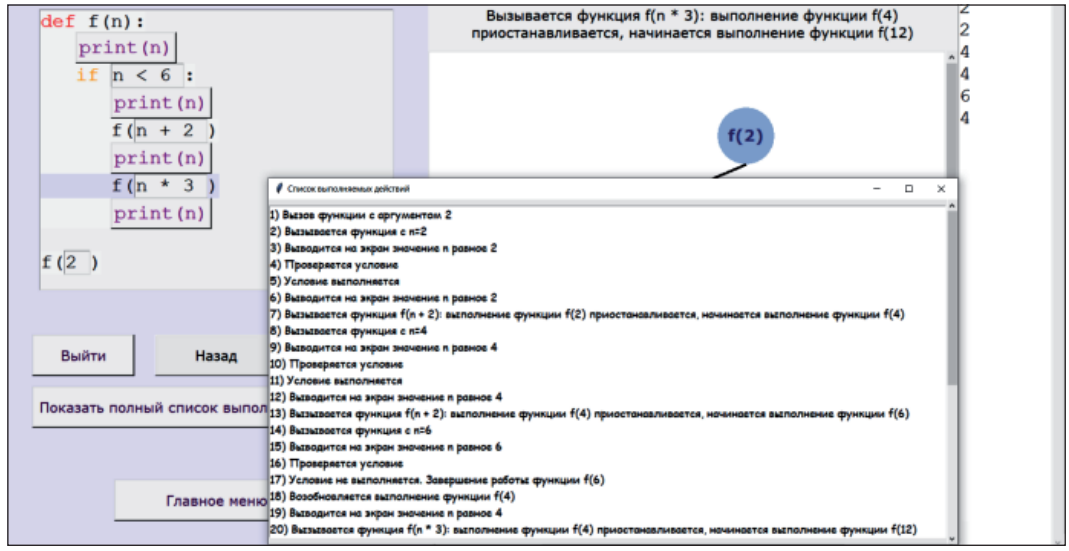

Рис. 21. Список действий, которые были выполнены в ходе работы программы, шаблон 2

4. Заключение

Наш опыт показывает, что использование описанного в статье набора заданий и программы-визуализатора *в условиях ограниченности учебного времени* помогает учащимся присвоить такие знаково-символические структуры, как рекуррентные формулы и деревья, и сформировать готовность осуществлять переходы от условия задачи к дереву и рекуррентной формуле; от дерева и рекуррентной формулы — к программе. Усвоение учащимися знаково-символических структур, необходимых для описания и визуализации рекурсивных алгоритмов, до процесса написания ими собственных рекурсивных подпрограмм позволит в дальнейшем сделать этот процесс управляемым, а деятельность школьников более осознанной.

Список источников

1. *Босова Л. Л.* Информатика: 9 класс: базовый уровень: учебник. М.: Просвещение, 2022. 208 с. EDN: OOJGYD.

2. *Босова Л. Л., Аквилянов Н. А., Кочерыгин И. О., Штепа Ю. Л., Бурцева Т. А.* Информатика. 8–9 классы. Начала программирования на языке Python. Дополнительные главы к учебникам. М.: БИНОМ. Лаборатория знаний, 2020. 96 с. EDN: LPMXCB.

3. Демоверсии, спецификации, кодификаторы ЕГЭ. https://fipi.ru/ege/demoversii-specifikacii-kodifikatory#!/tab/151883967-5

4. *Есаян А. Р.* Теория и методика обучения алгоритмизации на основе рекурсии в курсе информатики педагогического вуза: дис. ... д-ра пед. наук: 13.00.02 — теория и методика обучения и воспитания (по областям и уровням образования). Тула, 2001. 363 с. EDN: NLVEZZ.

5. *Заводчикова Н. И., Быкова И. А.* Организация различных видов знаково-символической деятельности при обучении программированию // Информатика в школе. 2023. № 5 (184). С. 33–38. DOI: 10.32517/2221-1993-2023-22-5-33-38. EDN: UZXXOD.

6. *Заводчикова Н. И., Быкова И. А.* Особенности методики работы с задачей по программированию при реализации семиотического подхода к обучению // Информатика в школе. 2025. Т. 24. № 5. С. 44–51. DOI: 10.32517/2221-1993-2025-24-5-44-51. EDN: YEJAEA.

7. *Заводчикова Н. И., Быкова И. А.* Особенности разработки укрупненных упражнений при реализации семиотического подхода в обучении программированию // Фундаментальные проблемы обучения математике, информатике и информатизации образования. Сборник тезисов докладов X международной научной конференции. Елец: Елецкий государственный университет им. И. А. Бунина, 2024. С. 51–54. EDN: XUOXHW.

8. *Маланов С. В.* Психолого-педагогические условия развития рефлексивных форм теоретического мышления в учебно-познавательной деятельности (в контексте деятельностного и культурно-исторического подходов к анализу и объяснению психических явлений): автореф. дис. ... д-ра психол. наук: 19.00.07 — теория и методика обучения и воспитания (по областям и уровням образования). Тула, 2010. 43 с. EDN: ZOAZTV.

9. *Мартынюк Ю. М., Ванькова В. С., Даниленко С. В.* Изучение и использование рекурсивных алгоритмов в подготовке учителя информатики // Чебышевский сборник. 2022. Т. 23. Вып. 5. С. 258–268. DOI: 10.22405/2226-8383-2022-23-5-258-268. EDN: VMPDYR.

10. Материалы для подготовки к ЕГЭ-2020 по информатике. https://kpolyakov.spb.ru/download/ege2020kp.7z

11. *Окулов С. М.* Основы программирования. 10-е изд., электрон. М.: Лаборатория знаний, 2020. 339 с.

12. *Орещенков И. С.* Изучение основ темы «Рекурсия» в среде графического исполнителя «Букашка» // Информатика в школе. 2024. Т. 23. № 2. С. 44–52. DOI: 10.32517/2221-1993-2024-23-2-44-52. EDN: CDHQUF.

13. *Пирогов В. Ю., Баландина И. В.* Основы методики обучения рекурсивному программированию // Интернет-журнал «Мир науки». 2018. Т. 6. № 4. С. 1–14. EDN: YLKSTB.

14. *Поляков К. Ю., Еремин Е. А.* Информатика. Углубленный уровень: учебник для 10 класса: в 2 ч. Ч. 2. М.: БИНОМ. Лаборатория знаний, 2013. 304 с.

15. *Попов В. С., Алефиренко Е. А., Черницына Л. Ю.* Рекурсивная функция для решения задания на динамическое программирование: от беззнаниевой формы компетенции к знанию и пониманию // Информатика в школе. 2024. Т. 23. № 5. С. 85–94. DOI: 10.32517/2221-1993-2024-23-5-85-94. EDN: MDGTEQ.

16. *Свейгарт Э.* Рекурсивная книга о рекурсии. СПб.: Питер, 2023. 336 с.

17. *Турчин А. С.* Семиотическая функция и процесс обучения. СПб.: Изд-во НУ «Центр стратегических исследований», 2019. 132 с. EDN: KVITUF.

ISSN 2221-1993 • **Информатика в школе** • 2026 • Том 25 № 1

18. Федеральная рабочая программа основного общего образования. Информатика (углубленный уровень) (для 7–9 классов образовательных организаций). https://edsoo.ru/wp-content/uploads/2023/08/16_ ФРП_Информатика-7-9-классы_угл.pdf

19. Федеральная рабочая программа среднего общего образования. Информатика (углубленный уровень) (для 10–11 классов образовательных организаций). https://edsoo.ru/wp-content/uploads/2023/08/22_ФРП_Информатика-10-11-классы_угл.pdf