

DOI: 10.32517/2221-1993-2023-22-2-25-33

В. О. Дженжер, Л. В. Денисова*Оренбургский государственный педагогический университет, Оренбург, Россия*

ЧИСЛОВЫЕ СПИРАЛИ: ИССЛЕДОВАНИЕ И ПРОГРАММИРОВАНИЕ

Аннотация

В статье описывается пример обучения школьников компьютерному моделированию с использованием программных и графических средств среды Scratch и системы программирования PascalABC.NET. Отмечено, что техника мастер-классов в работе кружка программирования не может считаться полноценной, если не содержит элементов доказательности и научности, а также не включает самостоятельное мышление школьника. Предлагается пример комбинированных занятий по занимательной математике и программированию, проводимых по схеме: 1) исследование математического объекта — некоторых числовых последовательностей; 2) визуализация последовательности путем построения на бумаге по определенному алгоритму; 3) математическое доказательство замкнутости или незамкнутости полученных числовых змеек; 4) построение компьютерной модели с использованием языка программирования. В статье приводятся различные примеры программ для школьников разного возраста и разного уровня подготовки. Приведенный в статье код краток, понятен и легко модифицируется, что позволяет использовать его как базу для проведения дальнейшего исследования. В заключение делаются выводы о том, что решение математических задач при помощи средств программирования позволяет соединить игровую деятельность с учебной и привнести элемент научности в работу детского кружка.

Ключевые слова: занимательная математика, программирование, компьютерное моделирование, Scratch, исполнитель Черепашка, PascalABC.NET, числовые спирали.

1. Введение

Постоянно возрастающая в последние десятилетия нехватка специалистов в области информационных технологий находит отражение в сфере дополнительного образования в виде большого количества различных курсов программирования, в том числе в кружках и печатных изданиях. Перед детьми и их родителями рисуются

заманчивые перспективы, вроде «пройди 30 уроков по созданию игр и стань программистом». Как правило, подобные курсы построены с применением блочных сред типа Scratch, Blockly, Codu Game Lab и т. п. для снижения возрастного порога. Оценить практическую ценность подобных курсов родители, как правило, не в состоянии. При опоре только на игровую мотивацию ребенок зачастую становится пассивным участником

Контактная информация

Дженжер Вадим Олегович, канд. физ.-мат. наук, доцент, доцент кафедры информатики, физики и методики преподавания информатики и физики, Оренбургский государственный педагогический университет, Оренбург, Россия; *адрес:* 460014, Россия, г. Оренбург, ул. Советская, д. 19; *e-mail:* vdjenjer@yandex.ru

Денисова Людмила Викторовна, канд. пед. наук, доцент кафедры информатики, физики и методики преподавания информатики и физики, Оренбургский государственный педагогический университет, Оренбург, Россия; *адрес:* 460014, Россия, г. Оренбург, ул. Советская, д. 19; *e-mail:* lv-denisova@yandex.ru

V. O. Dzhenzher, L. V. Denisova

Orenburg State Pedagogical University, Orenburg, Russia

NUMERICAL SPIRALS: RESEARCH AND PROGRAMMING

Abstract

The article describes an example of teaching schoolchildren computer modeling using software and graphical tools of the Scratch environment and the PascalABC.NET programming system. It is noted that the technique of master classes in the work of a programming circle cannot be considered complete if it does not contain an element of evidence and scientific character, and also does not include the student's independent thinking. An example of combined classes in entertaining mathematics and programming is offered, conducted according to the scheme: 1) the study of a mathematical object — some numerical sequences; 2) visualization of the sequence by building on paper according to a certain algorithm; 3) mathematical proof of the closedness or openness of the obtained numerical snakes; 4) developing a computer model using a programming language. The article provides various examples of programs for schoolchildren of different ages and different levels of training. The code given in the article is short, understandable and easily modified, which allows it to be used as a basis for further research. The conclusions are drawn that solving mathematical problems with the help of programming tools makes it possible to combine gaming activities with educational ones and introduce an element of science into the work of a kids club.

Keywords: entertaining mathematics, programming, computer modeling, Scratch, Turtle performer, PascalABC.NET, numerical spirals.

отношений, в которых нужно воспроизвести то, что показал учитель (или автор книги) в режиме мастер-класса. Если в книге и присутствуют какие-либо объяснения, то, скорее всего, они будут пропущены учеником: во-первых, в стремлении как можно быстрее сделать игру и похвастаться родителям или друзьям, а во-вторых, из-за отсутствия навыка осмысленного чтения. В этом наши заключения совпадают с результатами других наблюдений: неспособность читать художественную и научную литературу из-за непонимания содержания прочитанного подтверждается исследованиями психологов [8].

Разработка любой сколь-нибудь содержательной компьютерной игры предполагает использование переменных, циклов и условных операторов. Понимание концепции переменной (variable) невозможно без развитого абстрактного мышления, когда происходит оперирование не конкретными предметами и их количествами, а отвлеченными буквенными величинами и их отношениями. В свою очередь, использование в программе условных и циклических операторов требует развитого логического компонента понятийного мышления. Но ни абстрактное, ни понятийное мышление у большинства современных школьников даже к концу школьного обучения так и не развиваются [9]. Что же ожидать от младшего школьника, являющегося целевой аудиторией блочных сред и вышеупомянутых курсов программирования? В итоге все обучение сводится к репродукции кода учителя или автора книги, что лишь усугубляет проблему, создавая при этом у детей и их родителей ложное ощущение причастности к миру программирования.

Таким образом, дополнительное образование в виде курсов программирования для младших школьников должно не эксплуатировать естественную потребность детей играть и делать игры, а оказывать некоторую поддержку традиционному систематическому обучению в плане развития необходимых мыслительных операций. По нашему мнению, это возможно реализовать через построение компьютерных моделей математических объектов, в том числе с привлечением графики. Примеры подобного подхода можно встретить в [2–4], этому же посвящена и настоящая статья. В ней мы хотим привести примеры занятий для школьников младшего и среднего звеньев, изучающих программирование в курсах дополнительного образования. Занятия объединены общей темой «Построение числовых спиралей».

2. Числовые спирали на бумаге

В математике нет понятия числовых спиралей. Это условное название, смысл которого станет понятен после построения хотя бы одной из них.

Идея занятия появилась у нас благодаря книге американской учительницы Анны Вельтман «Математика — это красиво! Графическая тетрадь № 2» [1] после того, как на математическом кружке с учениками IV–V классов мы изучали следующий занимательный объект.

Пусть дана последовательность всех четных натуральных чисел от 2 до какого-то заданного числа. Будем вычислять сумму цифр каждого числа этой последова-

тельности (пока хватает терпения). Если полученная сумма окажется двузначным числом, то снова вычислим сумму его цифр.

Например, для числа 38 мы получим:

$$38 \rightarrow 3 + 8 = 11 \rightarrow 1 + 1 = 2.$$

Число	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38
Сумма цифр	2	4	6	8	1	3	5	7	9	2	4	6	8	1	3	5	7	9	2

Получили другую последовательность, в которой сначала идут все однозначные четные числа, затем все однозначные нечетные числа и затем все повторяется. Длина повторяющейся части называется **периодом**. В нашей последовательности период равен 9. То, что новая последовательность получилась **периодической**, — неочевидно и удивило всех. Поэтому далее можно (и нужно) рассмотреть последовательности, получающиеся из цепочки чисел, кратных 3, 4, 5 и т. д., и постараться уловить в них какие-либо закономерности. Здесь мы опустим эту часть занятия, поскольку она не имеет прямого отношения к данной статье.

Вернемся к полученной последовательности и построим ее на бумаге. Строить будем по клеткам, двигаясь последовательно вверх, вправо, вниз, влево. В каждом из этих направлений будем смещаться на количество клеточек, равное очередному элементу последовательности (рис. 1). Построение нужно проводить до тех пор, пока не произойдет возврат в исходную точку (рис. 2).

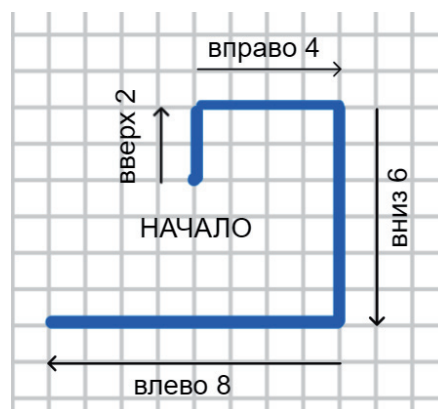


Рис. 1. Схема построения числовой последовательности

Всегда ли полученная траектория замкнется? Чтобы ответить на этот вопрос, предложим детям построить цепочку из чисел 2, 4, 6, 8. Уже после нескольких шагов становится понятно, что в начало никак не вернуться (рис. 3).

Из полученных рисунков становится видно, что построенные линии закручиваются в замкнутые или незамкнутые спирали или змейки. Отсюда и название.

Возникает вопрос: почему одни спирали получаются замкнутые, а другие — нет?

Необходимо выделить время для самостоятельного изучения учащимися этой проблемы. Ребята должны сами построить различные спирали и попытаться сформулировать гипотезу.

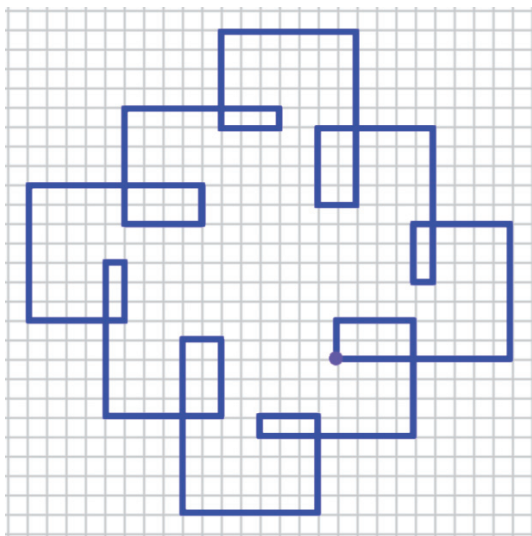


Рис. 2. Результат построения спирали по числовой последовательности 2, 4, 6, 8, 1, 3, 5, 7, 9

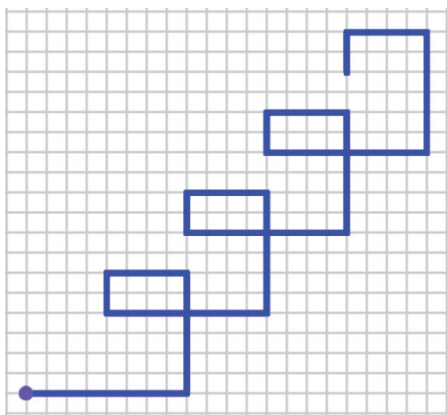


Рис. 3. Результат построения спирали по числовой последовательности 2, 4, 6, 8

3. Как это работает (неформальное доказательство)

Разберемся, почему при эволюции траекторий некоторые из них замыкаются, а некоторые нет.

Рассмотрим произвольную цепочку чисел. Пусть, например, это будут числа 3, 6 и 9. Мы помним, что ходы совершаются в строго определенном порядке: вверх, вправо, вниз, влево. Для удобства закодируем их буквами В (вверх), П (право), Н (низ) и Л (лево). Предложим детям построить таблицу, в которой столбцы будут подписаны соответствующими направлениями, и будем последовательно заполнять ее шагами из выбранной цепочки, начиная с шага длины 3.

В	П	Н	Л
3	6	9	3
6	9	3	6
9	3	6	9
3	Остановка		

Как только числа в первом столбике начнут повторяться, прекратим заполнение и проанализируем результат.

Заметим, что в каждом столбце имеются все числа из цепочки, а значит, каждый шаг на заданную длину выполнен во всех четырех направлениях. Рассмотрим движение на шаг длины a изолированно от других шагов. На каждое движение вправо на длину a имеется противоположное движение влево на ту же длину. Получается возврат в ту же точку. Аналогично на каждое движение вверх на длину a имеется противоположное движение вниз на ту же длину. И мы вновь имеем возврат в ту же точку. Поэтому, «отложив» каждое расстояние во всех четырех направлениях, мы в итоге возвращаемся туда, откуда начали.

Аналогично можно продемонстрировать возврат в начало и на других цепочках чисел. При этом совершенно не важно, имеется в этих цепочках закономерность или нет.

После этого становится очевидным, почему некоторые цепочки при их рисовании давали незамкнутые спирали, убегающие в бесконечность. Это свойство проявлялось на цепочках чисел, длина которых (период) кратна четырем. Младшим школьникам и в этом случае желательно построить таблицу ходов по направлениям. Например, для цепочки чисел 3, 5, 7, 9 последовательность ходов выглядит так:

В	П	Н	Л
3	5	7	9
3	5	7	9
...			

Отсюда видно, что строки таблицы просто повторяются и каждому числу цепочки соответствует единственное направление, возврата назад нет.

Несмотря на то что объяснение этого удивительного явления довольно простое, не все ученики могут увидеть его самостоятельно. Может быть, это связано с тем, что для построения спиралей на бумаге мы предлагали довольно длинные цепочки чисел. А здесь в качестве примера приводим цепочку всего из трех элементов. Так или иначе, на процесс рисования и на попытку объяснения получаемых результатов уходит целое занятие. И, конечно, только самые упорные и увлеченные ребята не оставляют попыток в рисовании длинных спиралей, поскольку малейшая ошибка способна погубить все усилия. А чтобы радость от знакомства с этим явлением не была омрачена неудачами, мы предлагаем поручить рутинную процедуру рисования компьютеру.

4. Компьютерное моделирование числовых спиралей

Рисованию числовых спиралей на компьютере посвящено отдельное занятие из серии уроков кружка программирования. Мы понимаем, что не все дети, посещающие кружок занимательной математики, изучают еще и программирование (и наоборот). Тем не менее

область пересечения этих двух групп детей в нашей практике довольно велика. В любом случае в начале урока мы еще раз проводим бумажный этап, но на этот раз без детального обсуждения. Затем переходим к созданию процедуры рисования числовых спиралей на компьютере.

4.1. Компьютерная реализация в среде Scratch [10]

Предлагаемое занятие возможно проводить в группе детей второго-третьего года обучения.

Школьник должен уверенно владеть следующими понятиями и командами:

- идти $\langle N \rangle$ шагов;
- повернуть на 90 градусов;
- переменная;
- цикл (повторить $\langle N \rangle$ раз);
- работа с пером;
- процедуры с параметром;
- работа со строками.

Желательно, чтобы этому занятию предшествовало построение квадратов и других прямоугольных узоров, например, лесенок и меандров.

4.1.1. Первый вариант программы

Идея самой простой программы состоит в том, чтобы воспользоваться уже имеющейся процедурой для рисования квадрата (см. рисунок 4 — рисование квадрата в Scratch) и немного исправить ее.

При рисовании змейки, так же как и при рисовании квадрата, мы должны смещаться на некоторую величину и поворачивать на 90 градусов после каждого смещения. В отличие от квадрата, величина смещения каждый раз

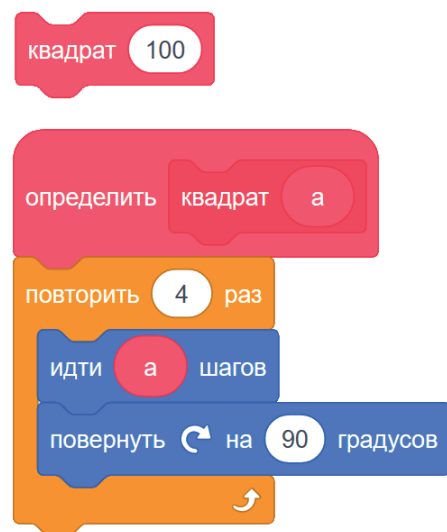


Рис. 4. Процедура рисования квадрата со стороной a и вызов ее со значением параметра, равным 100

меняется. При этом каждое смещение происходит 4 раза. Поэтому имеем первый вариант программы, представленный на рисунке 5.

Дадим пояснения по коду. Блок «Змейка 5» содержит шесть параметров. Первые пять — это последовательность, которую мы строим, а последний параметр — это размер шага исполнителя Перо. Конечно, сначала стоит создать с детьми процедуру с пятью параметрами, без шага. Но первый же запуск покажет, что получаемое изображение будет очень мелким либо нужно использовать в качестве элементов последовательности двузначные числа. Однако мы с ребятами сразу догово-

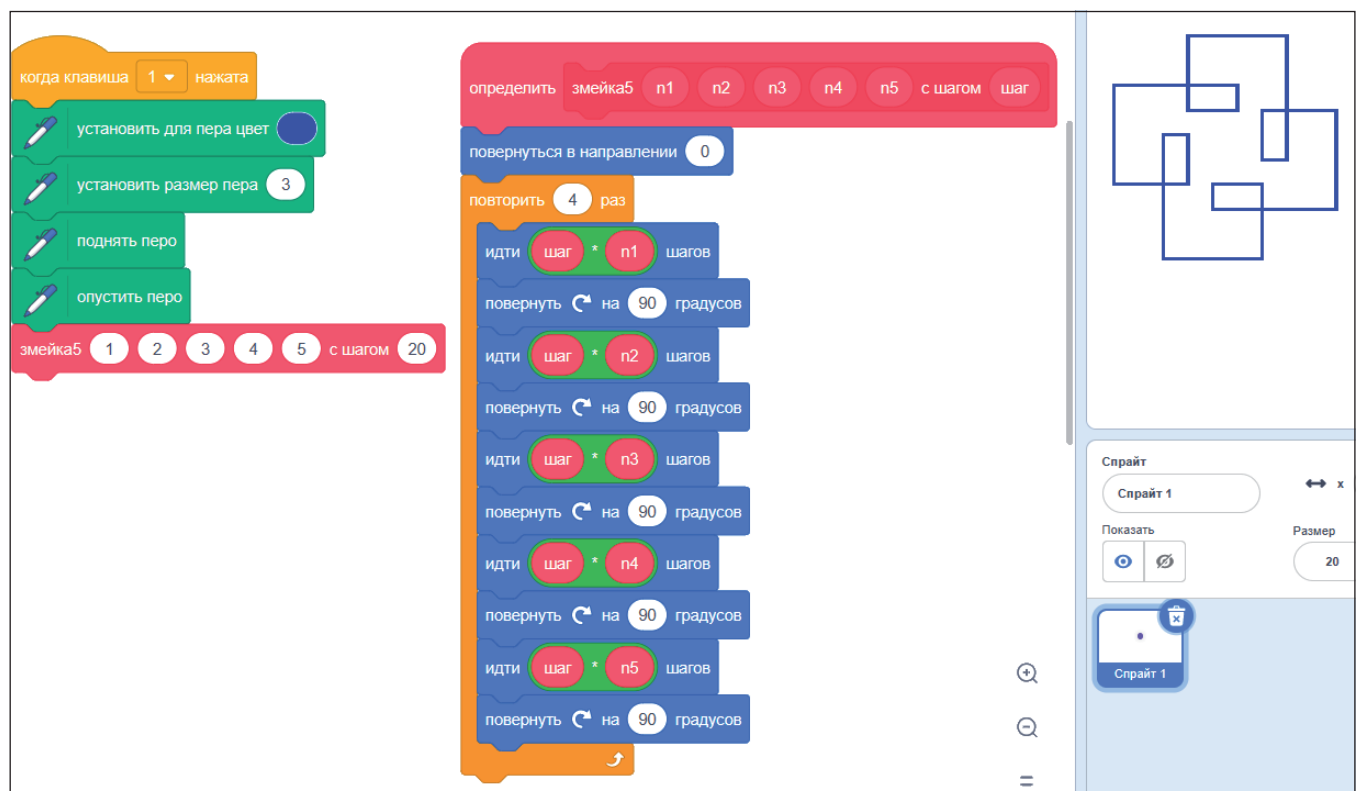


Рис. 5. Процедура «Змейка 5» и результат ее работы на последовательности 1, 2, 3, 4, 5

риваемся, что используем только однозначные числа*. Этот вариант кода мы здесь не приводим, потому что с ребятами мы сразу же переходим к обсуждению того, как можно увеличить рисунок, не увеличивая числа последовательности. Очевидно, что нужно применить идею масштаба — умножать каждый сдвиг на какое-то число. Удобнее всего это сделать при помощи еще одного — дополнительного — параметра, который мы назвали «шаг».

После экспериментов с разными последовательностями из пяти элементов задаем детям вопрос: а как быть, если мы захотим нарисовать более короткую или более длинную числовую спираль? Как правило, большинство детей сразу понимают, что можно создать еще один блок с нужным количеством параметров. Число пар «идти — повернуть» внутри цикла «Повторить 4 раза» будет равно длине змейки. Далее ученики очень ловко создают блок из трех элементов, из четырех элементов (заодно убеждаются, что такая змейка «убегает» и не закикливается). Но при попытке создать процедуру с семью элементами последовательности, а тем более с девятью ребята начинают допускать ошибки, связанные с большим количеством одинаковых команд и многократных перетаскиваний параметров из заголовка процедуры в тело процедуры. Предложение нарисовать змейку из 11 чисел энтузиазма у детей не вызывает.

И мы переходим ко второму варианту программы.

4.1.2. Работа со строками

Это новый проект, которому посвящена отдельная серия занятий.

Итак, мы хотим составить универсальный алгоритм, который будет действовать при любом (разумном) количестве чисел. Но это означает, что вместо отдельных параметров — элементов числовой последовательности мы должны работать с одним параметром — всей числовой последовательностью и обращаться к ее элементам от первого до последнего. В Scratch существуют два средства для работы с такими объектами — списки и строки. Для решения поставленной задачи наиболее удобным средством нам показались строки. Работа со строками ведется посредством четырех блоков палитры «Операторы» (рис. 6).

Если до этого ученики еще ни разу не работали со строками, то полезно вначале изучить работу этих четырех операторов. Для начала нужно перенести в область редактора каждый оператор и выполнить на нем щелчок кнопкой мыши. Получаемые результаты можно увидеть на рисунке 7. Следует обсудить с детьми каждый результат и обязательно изучить значения, получаемые на других входных данных.

Поставим теперь перед ребятами такую задачу: высветить на экране все буквы слова «яблоко» одна за другой. Линейный вариант дети могут предложить сами (рис. 8).

Будет ли эта программа работать на других словах? Ставим эксперименты и убеждаемся, что на всех шести-



Рис. 6. Операторы для работы со строками в Scratch



Рис. 7. Результаты работы операторов на значениях по умолчанию



Рис. 8. Линейный код для последовательного отображения всех букв слова «яблоко»

* Это наш методический прием, смысл которого станет ясен позднее. А детям можно напомнить, что мы строим цепочки чисел, являющихся суммами цифр. Однако, если в конце занятий у учеников все же возникнет необходимость в двузначных элементах, то следует переписать эту программу с использованием списков.

буквенных словах программа работает верно. На более длинных словах буквы с номерами, большими шести, не отображаются. Отдельного рассмотрения заслуживают короткие слова. Например, если каждой команде этой же программы подать на вход слово «слон», то на экране последовательно будут отображены все бук-

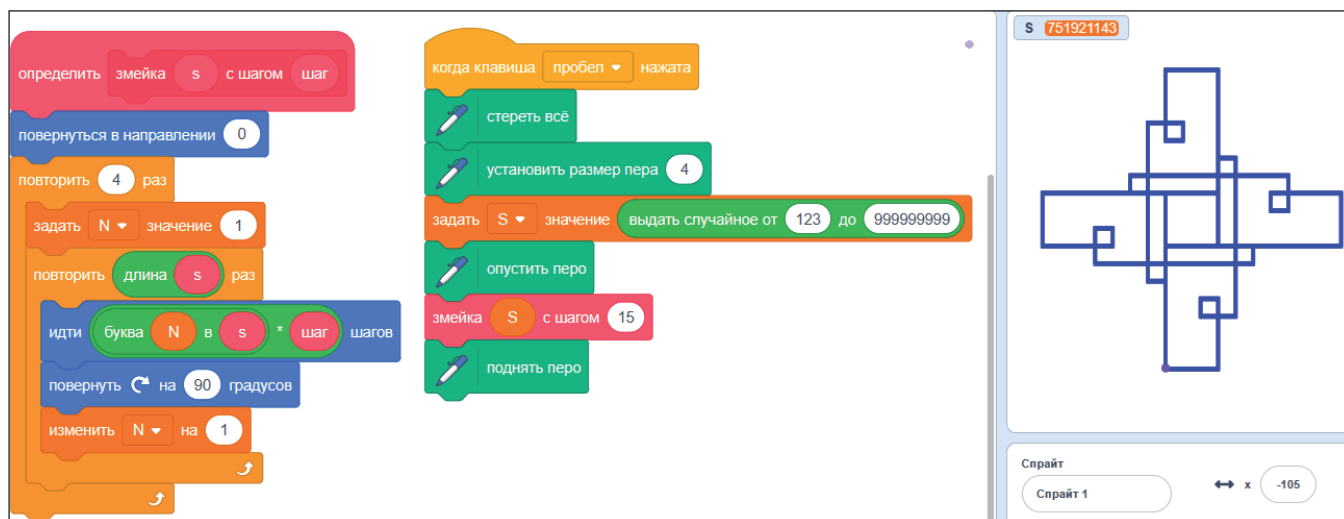


Рис. 12. Универсальная процедура «Змейка» и результат ее работы

4.1.3. Универсальная числовая змейка

Теперь мы подошли к тому, чтобы изменить процедуру чтения слова на рисование числовых змеек. Для этого будем хранить всю цепочку чисел в одной строке S . А затем *четырежды* пройдемся по ней, заглядывая в каждый ее элемент и смещаясь на длину, равную этому элементу (в масштабе), и выполняя поворот на 90 градусов после каждого смещения. Заметим, что теперь становится понятным, почему мы в качестве элементов последовательности подставляем только *однозначные* числа. И, наконец, если в строку подавать случайную последовательность, то результаты могут быть очень впечатляющими. Этот вариант программы представлен на рисунке 12.

4.2. Компьютерная реализация средствами PascalABC.NET

Как показывает практика, рассмотренные задачи могут быть интересны и для более старших школьников. Они, как правило, уже знакомы с понятиями вектора, коллинеарных векторов и суммы векторов. Поэтому для доказательства того, что змейка будет (или не бу-

дет) замкнутой, можно просто предложить вычислить сумму всех построенных векторов. Очевидно, что для замкнутой змейки мы имеем множество пар противоположно направленных векторов равной длины. Поэтому итоговая сумма окажется равной нулю, что и означает возврат в исходную точку.

Для рисования змеек со старшими школьниками мы используем PascalABC.NET [7], который является современным языком программирования, с успехом используемым как в школьном и вузовском преподавании, так и в научных исследованиях. В нем реализованы все базовые примитивы и парадигмы программирования [5, 6]. Поэтому PascalABC.NET позволяет писать лаконичный и понятный код. Являясь одним из языков платформы Microsoft.NET, язык использует все ее достоинства.

В PascalABC.NET можно смоделировать змейку различными способами.

Прежде всего, для этого подойдет исполнитель Черепашка, который, как правило, знаком школьникам. Его команды отлично подходят для рисования числовых змеек. Приведем пример такой программы и ее результат (рис. 13).

```
uses TurtleWPF;
begin
  SetSpeed(10); // Скорость Черепашки
  var a := |2, 8, 6, 7, 5, 1, 3|; // Массив смещений
  var step := 30; // Пикселей на единицу смещения
  SetWidth(4); // Толщина пера Черепашки
  Down; // Опустить хвост
  loop 4 do // 4 направления движения
    foreach var x in a do // Перебираем все смещения
      begin
        SetColor(RandomColor); // Случайный цвет пера
        Forw(x * step); // Делаем шаг
        Turn(90); // Поворот на 90 градусов
      end;
    end;
end.
```

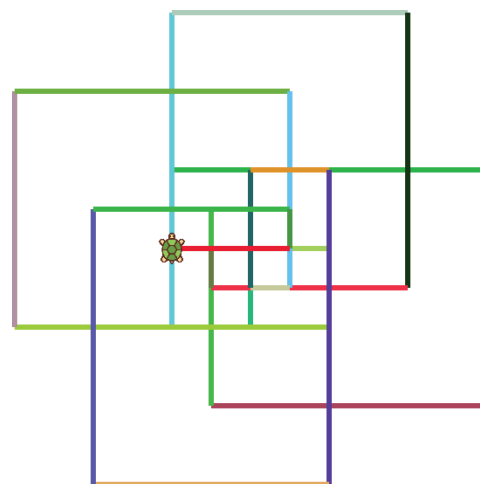


Рис. 13. Программа для рисования числовой змейки при помощи исполнителя Черепашка и ее результат

```

uses GraphWPF;
begin
  Window.Maximize; // Окно - на весь экран
  SetMathematicCoords(-30, 30); // Привычные координаты
  // var a := |3, 7, 5, 1, 2, 3, 6|;
  var a := |3, 7, 5, 1, 2, 4, 6|; // Массив смещений
  // Массив единичных направляющих векторов:
  var d := |(0, 1), (1, 0), (0, -1), (-1, 0)|;
  Pen.Width := 0.2; // Толщина пера
  Pen.RoundCap := True; // Скруглять стыки линий
  MoveTo(0, 0); // Рисуем из центра
  var k := 0; // Индекс в массиве d
  loop 4 do
    foreach var p in a do
      begin
        // Линия из текущей точки в заданную:
        LineBy(d[k].Item1 * p, d[k].Item2 * p);
        k := (k + 1) mod 4; // 0, 1, 2, 3, 0, 1, ...
        Pen.Color := RandomColor; // Случайный цвет
        Sleep(100); // Задержка в 100 миллисекунд
      end;
    end.
end.

```

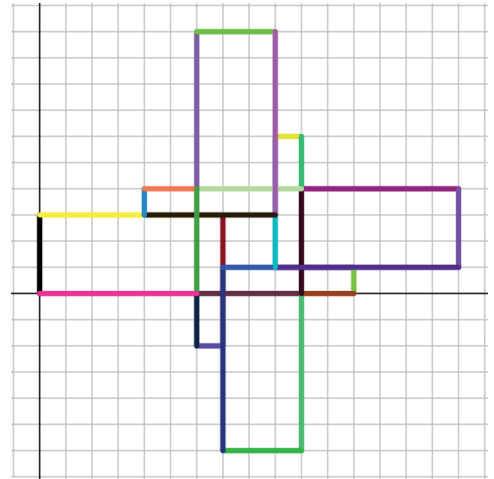


Рис. 14. Программа для рисования числовой змейки стандартными средствами в естественной системе координат и ее результат

Второй способ предполагает использование стандартных графических примитивов рисования, а именно линий. Этот метод принципиально не отличается от только что рассмотренного. Он выглядит немного сложнее, потому что предполагает самостоятельное моделирование поворотов и смещений, которые так легко совершает Черепашка. Но и ценность такой программы несколько выше: она требует использования большего количества инструментов программиста. Текст программы и результат ее работы приведены на рисунке 14.

В программе используется современный модуль GraphWPF, в котором можно работать с естественной (не графической) системой координат на экране*. Такая возможность появляется после вызова процедуры:

```
SetMathematicCoords(x1, x2);
```

которая подготавливает систему координат с отображением оси абсцисс от x_1 до x_2 . Размер по оси ординат зависит от размера окна. Масштаб по осям одинаковый. У процедуры есть дополнительные настройки, ознакомиться с которыми можно по встроенной подсказке. Заметим только, что теперь единицей является не пиксель, а размер клетки. Это поначалу может привести к путанице (обратите внимание на толщину пера: 0,2 — это не в пикселях).

Для смены направления движения мы используем массив d , в котором разместили единичные векторы вдоль нужных нам направлений: вверх (0; 1), вправо (1; 0), вниз (0; -1), влево (-1; 0). При умножении такого вектора на число p мы получим вектор длины p в том же направлении. Это соответствует нужному перемещению.

Элементы массива d являются кортежами из двух чисел. Добраться до них можно либо при помощи индекса (индексация от нуля), либо при помощи свойств, как это сделано у нас. Таким образом, $d[k]$ — это кортеж на k -й позиции, $d[k].Item1$ — первый элемент кортежа, $d[k].Item2$ — второй элемент кортежа. В цикле мы перебираем все возможные смещения p и получаем очередной вектор смещения как $(d[k].Item1 * p, d[k].Item2 * p)$. Затем передаем его в процедуру *LineBy*, которая проводит линию из текущей позиции на заданный вектор. В строке

```
k := (k + 1) mod 4;
```

переменная k принимает значения всех возможных остатков при делении на 4: 0, 1, 2, 3, 0, 1, 2, и т. д. Таким образом мы перебираем по очереди все направляющие вектора из массива d .

Цикл *ForEach* позволяет без индексов пройти по всем значениям массива a .

Приведенные выше программы являются только началом исследования интересного математического объекта. После занятия полезно предложить детям изучить поведение змеек при изменении угла поворота, например, с 90 градусов на 60 или 75 градусов либо другое число. Ребята могут попытаться самостоятельно изменить процедуру так, чтобы змейка замыкалась при других углах поворота. Можно подумать о том, когда змейка получается наиболее красивой. Повлияет ли на змейку четность и нечетность чисел? Их простота? Ответы на все эти и многие другие вопросы можно получить с использованием приведенных программ.

5. Заключение

Занятия на построение числовых змеек можно проводить в разных возрастных группах детей в дополнительном образовании и во внеурочной деятельности.

* Напомним, что в графической системе координат центр находится в левом верхнем углу экрана и ось Y направлена вниз. А в школьной тетрадке мы привыкли рисовать систему координат так, чтобы оси пересекались в центре листа и ось Y была направлена вверх.

Для компьютерной реализации изученной модели в данной статье предложены разные программные инструменты в зависимости от возраста детей и уровня их подготовки.

Важной частью занятий является обоснование замкнутости или незамкнутости линий. Это позволяет установить межпредметные связи и показывает важность математики для изучения программирования.

Выдвижение собственных гипотез, их проверку и дальнейшее развитие модели мы считаем важным шагом в развитии школьника. Именно такая работа позволяет отойти от обычных игровых техник и превратить занятие в небольшое исследование.

Список источников

1. Вельтман А. Математика — это красиво! Графическая тетрадь № 2. М.: Манн, Иванов и Фербер, 2018. 96 с.
2. Денисова Л. В., Дженджер В. О. Вокруг треугольника Серпинского. Игра в хаос и программирование в среде Scratch // Информатика в школе. 2020. № 4. С. 20–26. EDN: WOHVML. DOI: 10.32517/2221-1993-2020-19-4-20-26
3. Денисова Л. В., Дженджер В. О. Любителям среды программирования Scratch: математическое вышивание астроиды // Информатика в школе. 2019. № 3. С. 26–31. EDN: JXBLGP. DOI: 10.32517/2221-1993-2019-18-3-26-31
4. Златопольский Д. М. Красивые узоры таблицы Пифагора // Мир информатики. 2016. № 6. https://infojournal.ru/wp-content/uploads/2020/03/mir_info_6-2016.pdf
5. Михалкович С. С., Дженджер В. О. Новые примитивы в школьном программировании // Преподавание информационных технологий в Российской Федерации: сборник научных трудов. Материалы Восемнадцатой открытой всероссийской конференции (Москва, онлайн, 14–15 мая 2020 г.) / отв. ред. А. В. Альминдеров. М., 2020. С. 333–334. EDN: RBUAFI. https://it-education.ru/conf2020/thesis/IT-education_in_Russia_Thesis_2020_preview.pdf
6. Осипов А. PascalABC.NET. Введение в современное программирование. Ростов-на-Дону, 2019. 572 с. <http://pascalabc.net/downloads/OsipovBook/ModernProgr.pdf>
7. Система программирования PascalABC.NET. <http://pascalabc.net/>
8. Ясюкова Л. А. Как остановить дальнейшее падение качества образования? // Школьные технологии. 2015. № 3. С. 166–173. EDN: UADFPV.
9. Ясюкова Л. А. Как развиваются и как не развиваются математические способности: анализ методик обучения математике в современной школе // Школьные технологии. 2022. № 3. С. 102–110. EDN: ZFPONB.
10. Scratch — Imagine, Program, Share. <http://scratch.mit.edu>