DOI: 10.32517/2221-1993-2022-21-5-5-15

**Л. Л. Босова**
***Московский педагогический государственный университет, г. Москва, Россия***

**В. И. Филиппов**
***Академия социального управления, г. Москва, Россия***

**А. Ю. Босова**
***Ивановская средняя общеобразовательная школа, г. о. Истра, Московская область, Россия***

# ПРОГРАММИРОВАНИЕ ГРАФИКИ И АНИМАЦИИ В КУРСЕ ИНФОРМАТИКИ ОСНОВНОЙ ШКОЛЫ

***Аннотация***
В статье анализируются новые возможности в изучении тематического раздела «Алгоритмы и программирование» курса информатики основной школы, появившиеся при переходе на обновленные федеральные государственные образовательные стандарты. Приведено поурочное планирование темы «Компьютерная графика и анимация», инвариантное относительно модулей graph и turtle языка программирования Python («Знакомство с модулем», «Прямоугольник, окружность. Построение изображений из графических примитивов», «Использование циклов для построения изображений», «Штриховка замкнутой области простой формы», «Процедуры», «Вложенные циклы», «Анимация», «Управляемая анимация»). Дано описание основных возможностей модуля turtle. Представлены подробное описание каждого из восьми уроков темы, включающее пример решения базовой задачи и его разбор, а также задания для модификации предложенного решения и разработки на его основе решений других задач. Приведено решение средствами черепашьей графики задачи Единого государственного экзамена по информатике.
***Ключевые слова:*** информатика, обучение программированию, язык программирования Python, графика.

## 1. Введение

В настоящее время важность подготовки в области информатики и информационных технологий обучающихся всех уровней образования не вызывает сомнений. В декабре 2020 года Президент России В. В. Путин обратил внимание на необходимость совершенствования преподавания учебного предмета «Информатика» в общеобразовательных организациях с установлением его приоритета в учебном плане и корректировкой содержания примерных основных образовательных программ общего образования [6]. В 2021—2022 годах были при-

***Контактная информация***

**Босова Людмила Леонидовна**, доктор пед. наук, доцент, член-корреспондент РАО, зав. кафедрой теории и методики обучения математике и информатике, Институт математики и информатики, Московский педагогический государственный университет, г. Москва, Россия; *адрес:* 119991, Россия, г. Москва, ул. Малая Пироговская, д. 1, стр. 1; *e-mail:* akulll@mail.ru

**Филиппов Владимир Ильич**, канд. пед. наук, ст. преподаватель кафедры общеобразовательных дисциплин, Академия социального управления, г. Москва, Россия; *адрес:* 129344, Россия, г. Москва, ул. Енисейская, д. 3, корп. 5; *e-mail:* vf95@rambler.ru

**Босова Анна Юрьевна**, учитель информатики, Ивановская средняя общеобразовательная школа, г.о. Истра, Московская область, Россия; *адрес:* 143511, Россия, Московская область, Истринский район, п. Агрогородок; *e-mail:* abosova@gmail.com

**L. L. Bosova**
Moscow Pedagogical State University, Moscow, Russia
**V. I. Filippov**
Academy of Social Management, Moscow, Russia
**A. Yu. Bosova**
Ivanovo Secondary School, Istra, Moscow Region, Russia

## PROGRAMMING GRAPHICS AND ANIMATION IN THE INFORMATICS COURSE OF BASIC SCHOOL

***Abstract***
The article analyzes new opportunities in the study of the thematic section "Algorithms and Programming" of the informatics course of basic school, which appeared during the transition to updated Federal State Educational Standards. The planning of lessons of the theme "Computer graphics and animation" is given, invariant with respect to the graph and turtle modules of the Python programming language ("Introduction to the module", "Rectangle, circle. Building images from graphic primitives", "Using loops to create images", "Simple shape closed area hatching", "Procedures", "Nested loops", "Animation", "Controlled animation"). The main features of the turtle module are described. A detailed description of each of the eight lessons of the theme is presented, including an example of solving a basic problem and its analysis, as well as tasks for modifying the proposed solution and developing solutions to other problems based on it. The solution by means of turtle graphics of the problem of the Unified State Exam in informatics is presented.
***Keywords***: informatics, training in programming, Python programming language, graphics.

няты примерные рабочие программы основного общего образования по информатике для V—VI и VII—IX классов базового и углубленного уровней (https://edsoo.ru/Primernie_rabochie_progra.htm), в которых приоритетное внимание уделяется тематике, связанной с алгоритмами и программированием [9—11].

Линия «Алгоритмизация и программирование» — традиционная содержательная линия школьного курса информатики, освоение которой, по сути, способствует знакомству обучающихся с методологией решения широкого спектра жизненных задач и имеет в связи с этим ярко выраженный метапредметный характер. Тематический раздел «Алгоритмы и программирование», входящий во все примерные рабочие программы по информатике для основной школы, непосредственно направлен на «развитие алгоритмического мышления как необходимого условия профессиональной деятельности в современном информационном обществе, предполагающего способность обучающегося разбивать сложные задачи на более простые подзадачи; сравнивать новые задачи с задачами, решенными ранее; определять шаги для достижения результата и т. д.» [8], а также на освоение базовых навыков программирования как важной составляющей цифровых навыков современного человека.

Основными трудностями, связанными с изучением алгоритмизации и программирования, являются [2, 3]:

- позднее начало изучения и небольшое время, отводимое на освоение соответствующего содержания;
- преобладание задач с математическим контекстом: проверка делимости одного целого числа на другое; решение квадратного уравнения, имеющего вещественные корни; нахождение наибольшего общего делителя двух натуральных чисел; разложение натурального числа на простые сомножители; разбиение записи натурального числа в позиционной системе с основанием, меньшим или равным 10, на отдельные цифры и др.

Примерные рабочие программы основного общего образования по информатике дают шанс преодолеть указанные трудности.

Во-первых, выделено время на изучение темы «Алгоритмы и программирование в V—VI классах»: 10 часов в V классе и 12 часов в VI классе. При этом в V классе рекомендуется составление программ для управления исполнителем в среде блочного или текстового программирования, а в VI классе — переход на использование среды текстового программирования. Что касается V класса, то здесь мы считаем целесообразным использование среды программирования Scratch [4]. Шестиклассников можно познакомить с «черепашьей» графикой на Python (модуль *turtle*); методические рекомендации по работе с такой возрастной аудиторией представлены в работе [12] и на канале [7].

Во-вторых, при выборе для VII—IX классов углубленного уровня освоения информатики появляется возможность выделить на освоение алгоритмизации и программирования 86 часов учебного времени (табл. 1), начав этот процесс с VII класса (напомним, что на базовом уровне освоение алгоритмизации и программирования начинается в VIII классе и отводится на эту тему 27 часов).

*Таблица 1*

**Планирование раздела «Алгоритмы и программирование» в VII—IX классах на углубленном уровне**

| Класс | Темы | Количество часов |
|---|---|---|
| VII | Алгоритмы и исполнители. Алгоритмические конструкции | 16 |
| | Компьютерная графика и анимация | 8 |
| VIII | Язык программирования | 34 |
| IX | Разработка алгоритмов и программ | 24 |
| | Управление | 4 |
| | Итого: | 86 |

В-третьих, на углубленном уровне появляется возможность разнообразить предлагаем ые ученикам задачи, в частности, выделено время на тему «Компьютерная графика и анимация», в рамках которой предусмотрено программирование статических и движущихся изображений.

## 2. Поурочное планирование темы «Компьютерная графика и анимация»

Рассмотрим возможные подходы к реализации темы «Компьютерная графика и анимация» в VII классе.

Содержание этой темы в Примерной рабочей программе по информатике представлено следующим образом: «Система координат в компьютерной графике. Изменение цвета пикселя. Графические примитивы: отрезок, прямоугольник, окружность (круг). Свойства контура (цвет, толщина линии) и заливки. Построение изображений из графических примитивов. Использование циклов для построения изображений. Штриховка замкнутой области простой формы (прямоугольник, треугольник с основанием, параллельным оси координат). Принципы анимации. Использование анимации для имитации движения объекта. Управление анимацией с помощью клавиатуры» [11].

Принимая во внимание то, что на углубленном уровне возможно использование таких языков программирования, как КуМир (только в VII классе), Python, C++, C# и Java [8], остановим свой выбор на Python, а именно на использовании модулей *graph* и *turtle* языка программирования Python.

Возможное поурочное планирование темы «Компьютерная графика и анимация», инвариантное относительно модулей *graph* и *turtle*, представлено в таблице 2.

## 3. Общие сведения о модуле *turtle*

Рекомендации по реализации представленного выше поурочного планирования с использованием модуля graph даны в работе [1].

Приведем рекомендации по освоению темы «Компьютерная графика и анимация» с использованием модуля *turtle*, основные возможности которого представлены в таблице 3.

*Таблица 2*

**Поурочное планирование темы «Компьютерная графика и анимация»**

| № п/п | Тема урока | Теоретический материал |
|---|---|---|
| 1 | Знакомство с модулем *graph* (*turtl*e) | Система координат в компьютерной графике. Графические примитивы: точка, отрезок. Изменение цвета пикселя. Свойства контура (цвет, толщина линии) |
| 2 | Прямоугольник, окружность. Построение изображений из графических примитивов | Графические примитивы: прямоугольник, окружность (круг), дуга, сектор. Заливка |
| 3 | Использование циклов для построения изображений | Цикл *for*, функция *range* |
| 4 | Штриховка замкнутой области простой формы | Штриховка замкнутой области простой формы (прямоугольник, треугольник с основанием, параллельным оси координат) |
| 5 | Процедуры | Описание процедуры. Примеры |
| 6 | Вложенные циклы | Центрические и линейные орнаменты |
| 7 | Анимация | Принципы анимации. Использование анимации для имитации движения объекта |
| 8 | Управляемая анимация | Управление анимацией с помощью клавиатуры |

*Таблица 3*

**Графические возможности модуля *turtle* (Python 3)**

| № п/п | Команда | Описание | Пример |
|---|---|---|---|
| 1 | `from turtle import *` | Подключить модуль *turtle* | `from turtle import *` |
| 2 | `title(t)` | Заголовок окна рисования; рекомендуется в качестве заголовка давать название рисунка | `title("Picture")` |
| 3 | `setup(x, y)` | Установка размера окна рисования в пикселях | `setup(600, 600)` |
| 4 | `reset()` | Очищает экран, убирает все настройки, возвращает Черепаху домой — в центр экрана | `reset()` |
| 5 | `clear()` | Очищает экран. Удаляются рисунки Черепахи с экрана. При этом Черепаха не перемещается, ее состояние и положение не изменяются | `clear()` |
| 6 | `shape(f)` | Устанавливает форму Черепахи: arrow, turtle, circle, square, triangle, classic | `shape("arrow")` |
| 7 | `shapesize(m, n, k)` | Устанавливает размер Черепахи. Параметры команды — целые числа, определяющие ширину, высоту и толщину контура изображения; по умолчанию равны 1 | `shapesize(2)`<br>`shapesize(2, 4)`<br>`shapesize(2, 4, 2)` |
| 8 | `bgcolor(s)` | Устанавливает цвет фона; по умолчанию цвет фона белый | `bgcolor("blue")` |
| 9 | `color(s)` | Устанавливает цвет следа Черепахи и ее цвет; по умолчанию цвет черный | `color("brown")` |
| 10 | `color(s1, s2)` | Устанавливает цвет следа и контура Черепахи (s1) и ее цвет (s2); по умолчанию цвет черный | `color("brown", "red")` |
| 11 | `width n)` | Устанавливает ширину следа Черепахи в *n* пикселей | `width(5)` |
| 12 | `speed(n)` | Устанавливает скорость Черепахи — от 1 (медленно) до 10 (быстро) или 0 (мгновенно) | `speed(10)` |
| 13 | `down()` | Опускает перо, чтобы оставлять след при перемещении | `down()` |
| 14 | `up()` | Поднимает перо | `up()` |
| 15 | `forward(n)` | Перемещение Черепахи вперед на *n* пикселей | `forward(50)` |

*Окончание табл. 3*

| № п/п | Команда | Описание | Пример |
|---|---|---|---|
| 16 | `backward(n)` | Перемещение Черепахи назад на *n* пикселей | `backward(30)` |
| 17 | `left(m)` | Поворот Черепахи налево на *m* градусов | `left(90)` |
| 18 | `right(m)` | Поворот Черепахи направо на *m* градусов | `right(45)` |
| 19 | `circle(r)` | Рисование окружности радиуса *r*; с центром слева от Черепахи — если $r > 0$, справа — если $r < 0$ | `circle(20)` |
| 20 | `circle(r, m)` | Рисование дуги радиуса *r*, градусной мерой *m*; против часовой стрелки — если $r > 0$, по часовой стрелке — если $r < 0$ | `circle(50, 180)` |
| 21 | `dot(d, s)` | Рисование точки диаметра *d* цвета *s*; параметр *s* необязателен | `dot(10, "red")` |
| 22 | `goto(x, y)` | Перемещение Черепахи в точку с координатами (*x*, *y*) | `goto(10, 100)` |
| 23 | `setx(x)` | Установка координаты *x* Черепахи | `setx(300)` |
| 24 | `sety(y)` | Установка координаты *y* Черепахи | `sety(300)` |
| 25 | `home()` | Возвращение Черепахи домой — в точку с координатами (0, 0) | `home()` |
| 26 | `begin_fill()`<br>…<br>`end_fill()` | Заливка цветом Черепахи области, нарисованной с помощью команд, расположенных в программе между begin_fill() и end_fill() | `begin_fill()`<br>…<br>`end_fill()` |
| 27 | `mainloop()` | Прекращение выполнения программы | `mainloop()` |
| 28 | `exitonclick()` | Закрытие окна для графики при нажатии на левую кнопку мыши | `exitonclick()` |

## 4. Урок 1 «Знакомство с модулем *turtle*»

На первом уроке рассматриваются:

- система координат, в которой работает Черепаха;
- основные возможности Черепахи по изображению графических примитивов;
- возможности изменения свойств контура (цвета, толщины линии) и заливки.

Размер окна для вывода графических изображений рекомендуется подбирать в зависимости от диагонали устройства, имеющегося в распоряжении обучающегося. Начало координат в графическом окне для модуля *turtle* находится в центре окна. Положительное направление оси *X* определяется слева направо, положительное направление оси *Y* — снизу вверх: чем больше *X*, тем *правее* Черепаха, чем больше *Y*, тем *выше* Черепаха.

Пример системы координат в графическом окне размером 600×600 пикселей приведен на рисунке 1.

Знакомство обучающихся с Черепахой можно построить через разбор специально подготовленной про-

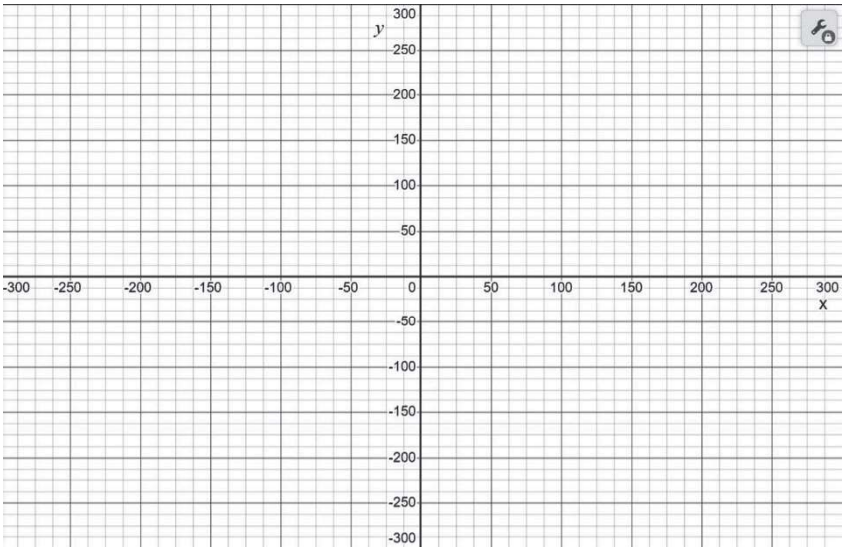



*Рис. 1. Графическое окно для Черепахи размером 600×600 пикселей*

граммы с основными операторами, пример которой с построчными комментариями приведен ниже:

| Текст программы | Комментарий к программе |
|---|---|
| `from turtle import*` | Подключение модуля *turtle* |
| `title("Picture")` | Заголовок окна с исполнителем. Рекомендуется в качестве заголовка давать название рисунка |
| `setup(600,600)` | Установка размеров графического окна в пикселях |
| `reset()` | Очистка экрана, черепашка переходит в центр экрана |
| `bgcolor("green")` | Установка цвета фона (графического окна) |
| `shape("turtle")` | Выбор внешнего вида Черепахи |
| `shapesize(2)` | Установка размера Черепахи |
| `color("brown")` | Установка цвета следа Черепахи и ее цвета |
| `width(5)` | Установка ширина следа Черепахи в пикселях |
| `down()` | Перо опускается. Начинается процесс рисования |
| | Команды рисования объектов |
| | |
| `up()` | Перо поднимается. Завершается процесс рисования |
| `exitonclick()` | Закрытие графического окна при нажатии на левую кнопку мыши |

При заданных в примере установках покажите ученикам, как можно изображать графические примитивы — точку и отрезок; дайте время для ввода и отладки текста программы в среде программирования.

Цвета в программах для Черепахи можно задать следующими способами:

1) явным образом (*white*, *black*, *gray*, *navy*, *blue*, *cyan*, *green*, *yellow*, *red*, *orange*, *brown*, *maroon*, *violet*, *purple* и ряд других цветов). На сайте: https://trinket.io/docs/colors можно узнать о всех доступных названиях цветов;
2) через шестнадцатеричный код цвета (например, красный цвет: #ff0000). Для определения кода цвета можно использовать сайт: https://trinket.io/docs/colors;
3) через десятичный код цвета путем предварительной установки десятичного режима шкалы RGB с помощью функции *colormode*(255).

Предложите ученикам поэкспериментировать со свойствами контура — цветом и толщиной.

Обсудите с учениками разные способы перемещения Черепахи:

1) на заданное число шагов в заданном направлении;
2) в точку с заданными координатами.

В заключительной части урока можно предложить ученикам самостоятельно запрограммировать следующее изображение (рис. 2).

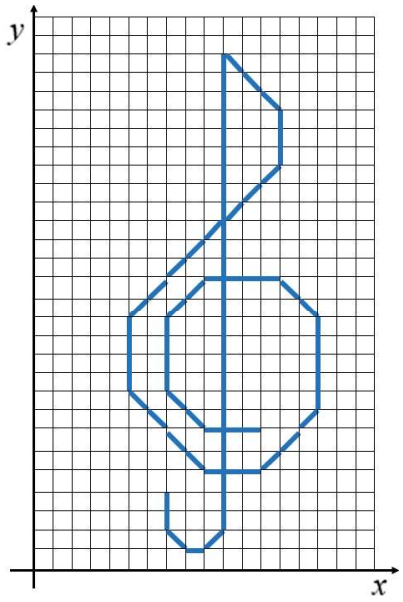


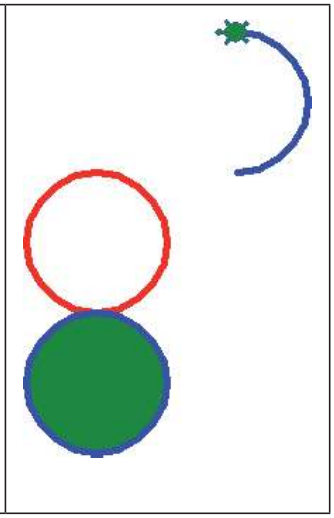

*Рис. 2. Рисунок для Черепахи*

## 5. Урок 2 «Прямоугольник, окружность. Построение изображений из графических примитивов»

На втором уроке рекомендуется заняться работой с другими графическими примитивами: окружностью, дугой, треугольником и прямоугольником.

Сначала можно обсудить со школьниками программу изображения окружности, круга (закрашенной окружности) и дуги:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("red")
width(5)
circle(50)
color("blue","green")
begin_fill()
circle(-50)
end_fill()
up()
goto(100,100)
down()
circle(50, 180)
```



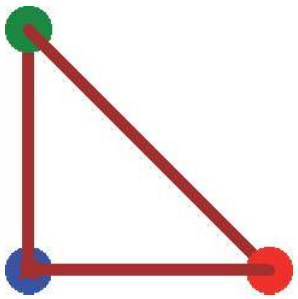
При отработке умений изображения окружностей и дуг можно предложить школьникам нарисовать один или несколько смайликов (рис. 3).

*Рис. 3. Смайлики*

На этапе формирования умений изображения треугольников и прямоугольников можно предложить ученикам по имеющимся текстам программ установить, что будет нарисовано Черепахой на экране при их выполнении. При этом следует обратить внимание обучающихся на возможность использования команд перемещения как по координатам, так и по точкам.

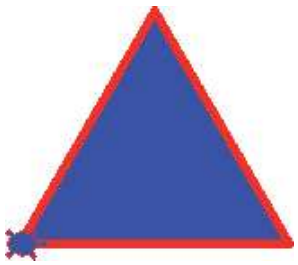
Пример программы рисования треугольника по координатам:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("brown")
speed(10)
dot(40,"blue")
width(10)
goto(0,200)
dot(40,"green")
goto(200,0)
dot(40,"red")
goto(0,0)
```



Пример программы рисования закрашенного равностороннего треугольника:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("red","blue")
width(5)
begin_fill()
forward(150)
left(120)
forward(150)
left(120)
forward(150)
left(120)
end_fill()
```



Последний из рассмотренных примеров позволяет обратить внимание школьников на то, что одна и та же группа из трех команд выполнялась в программе трижды. Аналогичная ситуация возникает при изображении квадрата и любого другого правильного многоугольника. О том, как записывать многократно повторяющиеся действия, пойдет разговор на третьем уроке темы.

## 6. Урок 3 «Использование циклов для построения изображений»

Необходимо познакомить обучающихся с циклом *for* и его реализацией в Python.

Общий вид цикла *for*:

```
for<параметр> in range(k, n, m):
  <операторы>
```

Здесь:

- <параметр> — переменная целого типа;
- *range*() — функция, описывающая необходимое количество повторов тела цикла; в скобках может быть указано от одного до трех чисел:
  - одно число ($n$) указывает на то, что нужно последовательно перебрать все целые числа от 0 до $n$, причем само $n$ не рассматривается;
  - два числа ($k$, $n$) говорят о том, что нужно последовательно перебрать все целые числа, находящиеся в диапазоне от $k$ (начальное значение) до $n - 1$ (конечное значение);
  - три числа ($k$, $n$, $m$) указывают на то, что параметр должен изменяться от $k$ до $n - 1$ с шагом, равным $m$;
- <операторы> — один или несколько операторов, образующих тело цикла.

При выполнении этого оператора после каждого выполнения тела цикла происходит увеличение на шаг параметра цикла. Если $k = n$ или $k > n$, цикл не выполнится ни разу.

Программа рисования закрашенного равностороннего треугольника с циклом *for* может быть записана так:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("red","blue")
width(5)
begin_fill()
for i in range(3):
  forward(150)
  left(120)
end_fill()
```

Обсудив эту программу со школьниками, предложите им:

1) воспроизвести программу в среде программирования;
2) модифицировать ее в программу рисования квадрата.

После этого рекомендуется совместными усилиями сформулировать правило рисования Черепахой правильных многоугольников: чтобы нарисовать правильный n-угольник со стороной $a$, Черепаха должна повторить $n$ раз группу команд:

```
forward(a)
left(360/n)
```

Далее следует пояснить ученикам, как можно ввести с клавиатуры значения целочисленных величин $a$ и $n$, и предложить им разработать универсальную программу рисования правильного многоугольника:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("red")
width(5)
down()
```

```
n=int(input('Введите количество углов'))
a=int(input('Введите длину стороны'))
for i in range(n):
  forward(a)
  right(360/n)
```

Можно предложить ученикам усовершенствовать эту программу, введя переменную толщину контура и предусмотрев заливку внутренней области многоугольника.

Продолжить освоение цикла *for* можно в процессе анализа следующей программы, результатом которой является ряд из 10 точек в центральной части экрана:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("red","blue")
n=10
x = -200
up()
for i in range(10):
  goto(x, 0)
  dot(20)
  x = x + 50
```

Предложите ученикам усовершенствовать программу, вводя с клавиатуры количество точек, диаметр точки и величину, на которую изменяется значение переменной *x*.

Закрепить умение работать с циклом *for* можно в процессе создания программ, выводящих на экран следующие изображения:

1) горизонтальный ряд из *n* точек диаметра *d*;
2) один из орнаментов, представленных на рисунке 4.

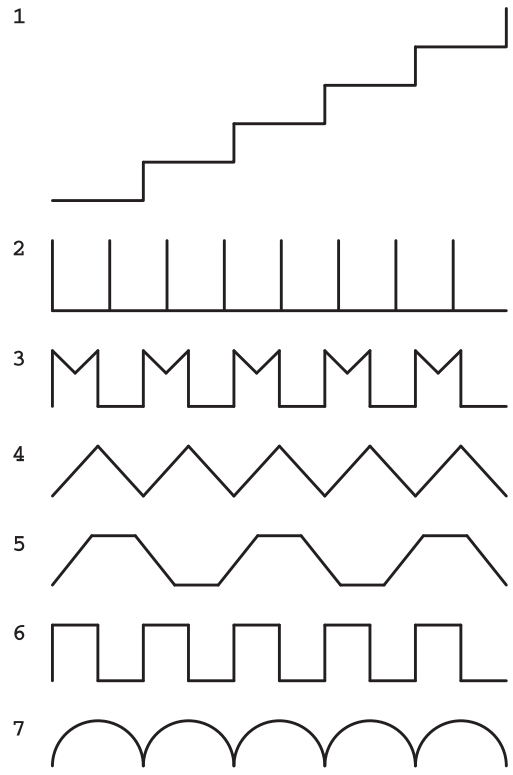


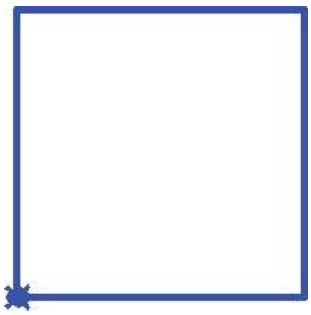

*Рис. 4. Варианты геометрических орнаментов*

## 7. Урок 4 «Штриховка замкнутой области простой формы»

На четвертом уроке запланировано продолжение отработки навыков работы с циклическими алгоритмами в процессе штриховки замкнутой области простой формы, например, прямоугольника. Штриховка строится из параллельных линий, которые удобно рисовать в цикле.

Изобразим в центре экрана квадрат со стороной 200 пикселей:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
width(5)
color("blue")
up()
goto(-100, -100)
down()
for i in range(4):
  fd(200)
  left(90)
```



Выполним вертикальную штриховку квадрата отрезками, находящимися от сторон квадрата и друг от друга на расстоянии 10 пикселей. Количество полос *n* рассчитаем по формуле:

$$n = (x2 - x1) / h,$$

где:

$x1 = -100$ и $x2 = 100$ — абсциссы левой и правой сторон нашего квадрата;

*h* — расстояние между линиями штриховки.

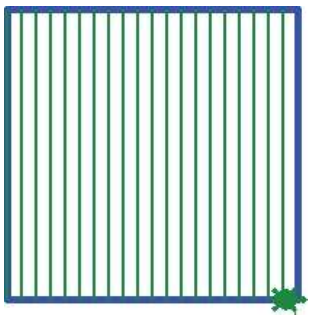
В нашем случае $n = 20$.

В любом случае количество линий — целое значение. Поэтому в общем случае величину *n* следует округлять до ближайшего целого значения:

$$n = round((x2 - x1) / h).$$

Программа штриховки будет иметь вид:

```
width(2)
color("green")
x = -100
for i in range(20):
  up()
  goto(x,100)
  down()
  goto (x,-100)
  x = x + 10
```



Предложите ученикам:

1) воспроизвести программу в среде программирования;
2) модифицировать программу так, чтобы расстояние между линиями вводилось с клавиатуры;
3) изменить программу так, чтобы выполнялась горизонтальная штриховка квадрата;

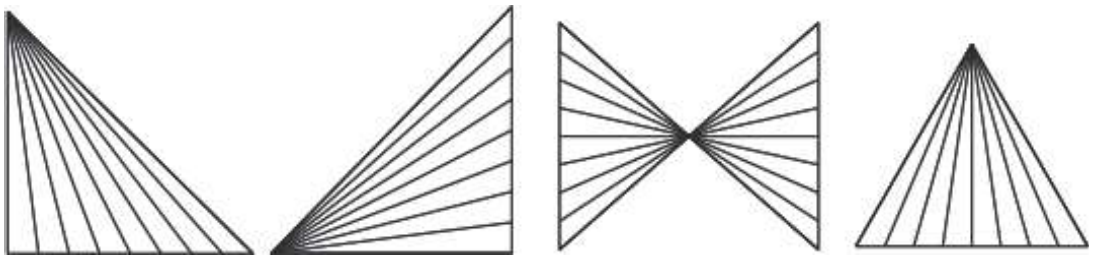


*Рис. 5. Варианты штриховки фигур*

4) изменить программу так, чтобы выполнялась диагональная штриховка квадрата;
5) написать программу, изображающую одну из фигур, представленных на рисунке 5.

## 8. Урок 5 «Процедуры»

На пятом уроке предполагается знакомство школьников с понятием процедуры и с тем, как описывается процедура на языке программирования Python.

***Процедура*** — это фрагмент программного кода, который решает какую-либо задачу. Его можно вызывать в любом месте основной программы. Процедуры помогают избегать дублирования кода при многократном его использовании.

Описание процедуры имеет вид:

```
def <имя процедуры>():
  <операторы>
```

Процедура начинается со служебного слова *def* (от *англ.* define — определить). После этого записываются имя процедуры, скобки и двоеточие. Операторы, которые входят в тело процедуры, записываются с отступом. Так мы показываем, какие команды входят в процедуру. Для того чтобы процедура заработала, ее необходимо вызвать по имени; причем таких вызовов может быть сколько угодно.

Процедура должна быть определена к моменту ее вызова, т. е. должна быть выполнена команда *def*, создающая объект-процедуру в памяти. Если процедура вызывается из основной программы, то нужно поместить определение процедуры раньше точки вызова.

Оформим фрагмент кода, строящего изображение квадрата, в виде процедуры:

```
def square():
  for i in range(4):
    forward (20)
    left(90)
```

Следующий за процедурой код основной программы, использующий ее для изображения ряда из 12 квадратов, будет иметь вид:

```
from turtle import*
setup(600,600)
reset()
up()
goto(-280,0)
shape("turtle")
shapesize(1)
color("red")
width(5)
down()
for i in range(12):
  square()
  up()
  forward(40)
  down()
```

Предложите ученикам:
1) выполнить программу в среде программирования;
2) модифицировать программу, изменив длину стороны квадрата и расстояние между квадратами;
3) изменить программу так, чтобы получался вертикальный ряд квадратов;
4) изменить программу так, чтобы добиться диагонального расположения квадратов;
5) создать процедуру *cross* (рис. 6) и с ее помощью изобразить пять крестиков в разных частях экрана: один — в начале координат и по одному в каждой координатной четверти.

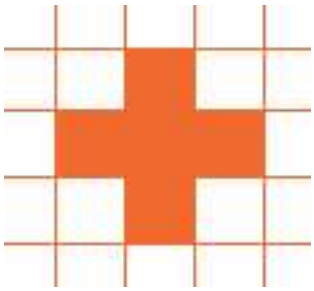


*Рис. 6. Изображение фигуры «крестик»*

Запишем в виде процедуры с параметрами фрагмент программного кода, изображающего правильный многоугольник:

```
def regular_polygon(a,n):
  for i in range(n):
    fd(a)
    left(360/n)
```

В этой процедуре два параметра: $a$ — длина стороны и $n$ — количество углов правильного многоугольника. При вызове процедуры из основной программы надо указывать конкретные числовые значения этих параметров, например: *regular_polygon*(30, 6).

Предложите ученикам:
1) написать программу, выводящую в разных частях экрана равносторонний треугольник, квадрат, правильный пятиугольник и правильный шестиугольник с использованием процедуры *regular_polygon*($a$, $n$);

2) превратить процедуру *cross*() в процедуру с параметром *cross*(*a*), где *a* — размер клетки, составляющей 1/3 высоты крестика;
3) написать процедуру с параметрами *rectangle*(*a*, *b*) и на ее основе написать основную программу, выводящую на экран одно из изображений, представленных на рисунке 7.

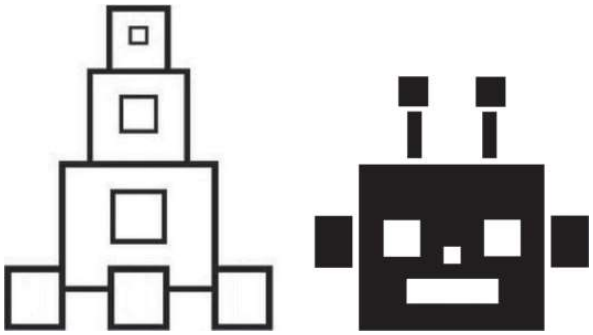


*Рис. 7. Изображения из прямоугольников*

## 9. Урок 6 «Вложенные циклы»

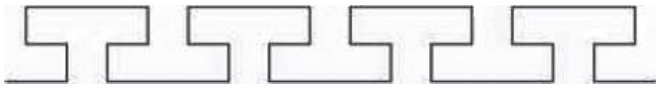
На шестом уроке рекомендуется ввести понятие вложенного цикла.

***Цикл*** называется ***вложенным***, если он содержится внутри (в теле) другого цикла. Цикл, содержащий в себе другой цикл, называют ***внешним***, а цикл, содержащийся в теле другого цикла, — ***внутренним***.

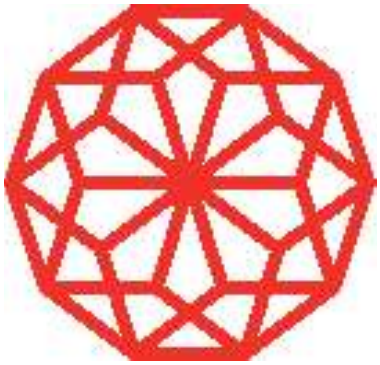
Обсудите с учениками следующую программу, выясните, что получится в результате ее выполнения:

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(1)
color("red","blue")
n=5
y = 200
up()
for i in range(n):
  x=-200
  for i in range(10):
    goto(x, y)
    dot(20)
    x = x + 50
  y = y - 50
```

Предложите ученикам:
1) воспроизвести программу в среде программирования;
2) модифицировать программу так, чтобы выводилось 10 рядов по пять точек в каждом ряду;
3) изменить программу так, чтобы на экран выводилось пять рядов следующих изображений:



4) написать программу с подпрограммой, в результате выполнения которой на экране должны появиться три ряда:
   - ряд из пяти равносторонних треугольников;
   - ряд из пяти квадратов;
   - ряд из пяти правильных восьмиугольников;
5) написать программу с подпрограммой, выводящую на экран центрический орнамент из 10 правильных пятиугольников:



## 10. Урок 7 «Анимация»

На седьмом уроке обучающиеся научатся создавать простейшие анимации.

***Анимация*** на компьютере — это, как правило, быстрая смена рисунков, создающая иллюзию движения.

Анимация создается следующим образом:
1) объект рисуется в точке (*x*, *y*);
2) делается пауза (задержка) на несколько миллисекунд;
3) объект стирается;
4) изменяются координаты (*x*, *y*);
5) осуществляется переход к шагу 1.

Пользуясь свойствами Черепахи, мы пропускаем пункт 1 и можем пропустить пункт 3.

Предложите ученикам проанализировать следующую программу: что получится в результате ее выполнения?

```
from turtle import*
setup(600,600)
reset()
shape("turtle")
shapesize(2)
bgcolor("green")
color("red","brown")
x = 0
up()
for i in range(58):
  goto(x, 0)
  x = x + 5
  clear()
```

После того как ученики проверят свои предположения в среде программирования, предложите им дополнить программу так, чтобы Черепаха обходила графическое окно по периметру.

Реализация простой анимации возможна с использованием случайных чисел.

Генерация случайных значений в программах на языке программирования Python осуществляется путем импорта модуля *random*. Оператор *randint*(*A*, *B*) возвращает случайное целое число *n* такое, что $A \le n \le B$.

Обсудите с учениками следующую программу:

```
from turtle import *
from random import randint
setup (600,600)
reset()
shape ("turtle")
speed(0)
colormode(255)
for i in range (100):
  dot(randint (20,50),(randint(0,255),0,0))
  penup()
  goto(randint(-300, 300), randint(-300, 300))
  down()
```

Предложите ученикам:

1) воспроизвести программу в среде программирования;
2) модифицировать программу так, чтобы:
   - выводилось большее число точек;
   - размеры точек изменялись в другом диапазоне;
   - изменялись все составляющие цвета точки;
   - Черепаха оставляла след при перемещении от точки к точке.

## 11. Урок 8 «Управляемая анимация»

Восьмой урок — завершающий урок по теме; он посвящен управлению анимацией с клавиатуры. Ученикам надо пояснить, что Python — объектно-ориентированный язык программирования.

***Объект*** — это набор данных (переменных) и методов (функций), которые с этими данными взаимодействуют. Каждый объект является экземпляром некоторого класса. В свою очередь, ***класс*** — это программный шаблон (заготовка, трафарет) для создания объектов. ***Событие*** — это некоторый сигнал, запускающий программный код, связанный с этим сигналом.

*Turtle* — класс, его имя пишется с большой буквы.

С помощью оператора присваивания создадим экземпляр класса (объект):

```
circ = turtle.Turtle()
```

Свойства этого объекта опишем так:

```
circ.shape("circle")
circ.color("orange")
circ.up()
```

Функция *turtle.listen*() обеспечивает сбор событий, например, нажатий шклавиш.

Функция *turtle.onkeypress*(*fun*, *key*) обеспечивает обработку нажатий клавиш. Для этого существуют две функции: вызывается функция *fun* при нажатии клавиши *key*.

Клавиша задается строкой с ее названием. Например:

- `"Up"` — стрелка вверх;
- `"Down"` — стрелка вниз;
- `"Left"` — стрелка влево;
- `"Right"` — стрелка вправо.

По нажатию клавиши мы будем перемещать фигуру. Для этого понадобятся функции, которые сообщают и изменяют координаты:

- *xcor*() и *ycor*() выдают координаты по *x* и *y*;
- *setx*(*x*) и *sety*(*y*) устанавливают координаты *x* и *y*.

Напишем функцию *up*(), которая будет запускаться при нажатии стрелки вверх и перемещать объект *circ* на 10 пикселей вверх:

```
def up():
  y = circ.ycor() + 10
  circ.sety(y)
Полный текст программы:
def up():
  y = circ.ycor() + 10
  circ.sety(y)
import turtle
circ = turtle.Turtle()
circ.shape("circle")
circ.color("orange")
circ.up()
turtle.listen()
turtle.onkeypress(up, "Right")
turtle.exitonclick()
```

Предложите ученикам:

1) воспроизвести программу в среде программирования;
2) модифицировать программу так, чтобы скорость движения объекта была больше (меньше);
3) дополнить программу, организовав возможность управления движением объекта вниз, вправо и влево;
4) добавить второй объект и настроить управление им.

Дальнейшее развитие этой тематики возможно в рамках внеурочной деятельности, которая может быть построена на основе материалов [5].

## 12. Заключение

После освоения содержания представленных выше восьми уроков темы «Компьютерная графика и анимация» семиклассники:

- будут уметь писать простые программы на языке программирования Python;
- познакомятся со средой разработки;
- будут понимать, что такое синтаксис языка;
- получат первоначальный опыт отладки программ.

Использование модуля *turtle* языка программирования Python будет способствовать развитию у обучающихся алгоритмического, логического и системного мышления, формированию широкого спектра цифровых навыков. Такой задел позволит школьникам легче перейти к программированию алгоритмов обработки числовых и строковых данных с использованием языка программирования Python.

В заключение приведем еще один неожиданно возникший довод в пользу изучения черепашьей графики, а именно, представленное ниже условие задачи 6 из проекта демонстрационного варианта контрольных измерительных материалов Единого государственного экзамена 2023 года по информатике (https://fipi.ru/ege/demoversii-specifikacii-kodifikatory#!/tab/151883967-5).

**Задача.**

Исполнитель Черепаха действует на плоскости с декартовой системой координат. В начальный момент Че-

репаха находится в начале координат, ее голова направлена вдоль положительного направления оси ординат, хвост опущен. При опущенном хвосте Черепаха оставляет на поле след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существует две команды: `Вперед n` (где *n* — целое число), вызывающая передвижение Черепахи на *n* единиц в том направлении, куда указывает ее голова, и `Направо m` (где *m* — целое число), вызывающая изменение направления движения на *m* градусов по часовой стрелке.

Запись `Повтори k [Команда1 Команда2 … КомандаS]` означает, что последовательность из *S* команд повторится *k* раз.

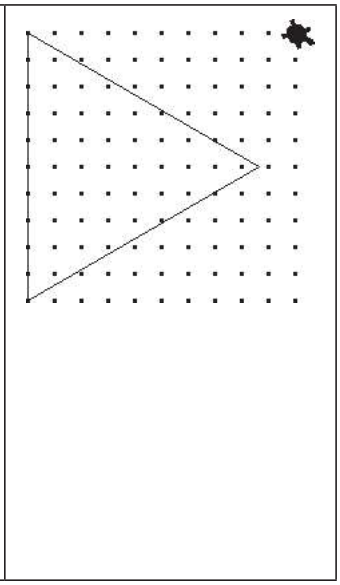
Черепахе был дан для исполнения следующий алгоритм:

```
Повтори 7 [Вперед 10 Направо 120]
```

Определите, сколько точек с целочисленными координатами будут находиться внутри области, ограниченной линией, заданной данным алгоритмом. Точки на линии учитывать не следует.

Надеемся, читатели смогут оценить красоту решения этой задачи с помощью черепашьей графики:

```
from turtle import*
setup (600,600)
reset()
shape ("turtle")
up()
left(90)
down()
for i in range(7):
    forward(10*20)
    right(120)
y = 0
up()
for i in range(11):
  x=0
  for i in range(11):
    goto(x, y)
    dot(4)
    x = x + 20
  y = y +20
```



## Список источников


1. *Босова Л. Л.* Информатика (углубленный уровень). Реализация ФГОС основного общего образования: методическое пособие для учителя. М.: Институт стратегии развития образования РАО, 2022. 211 с.

2. *Босова Л. Л.* Как учат программированию в XXI веке: отечественный и зарубежный опыт обучения программированию в школе // Информатика в школе. 2018. № 6. С. 3–11. EDN XZOOJV.

3. *Босова Л. Л.* Программирование в школе: возможности, проблемы, решения // Информатизация образования — 2018. Труды Международной научно-практической конференции (Москва, 11–12 сентября 2018 года) / Академия информатизации образования; Академия компьютерных наук, Институт управления образованием РАО. М.: Изд-во Современного гуманитарного университета, 2018. С. 172–179. EDN VUJTHS.

4. *Босова Л. Л., Босова А. Ю., Филиппов В. И.* «Программируем, учимся и играем!». Программа курса внеурочной деятельности для учащихся III–VI классов // Информатика в школе. 2021. № 6. С. 3–15. DOI 10.32517/2221-1993-2021-20-6-3-15. EDN PPFNPU.

5. *Бриггс Д.* Python для детей. Самоучитель по программированию / пер. с англ. С. Ломакина. 2-е изд. М.: Манн, Иванов и Фербер, 2018. 320 с.

6. Перечень поручений Президента России В. В. Путина по итогам конференции по искусственному интеллекту. http://www.kremlin.ru/acts/assignments/orders/64859

7. Плейлист Python. Turtle graphics. https://www.youtube.com/playlist?list=PLMInhDclNR1FjCQvSdT1vvrlAFRx__ehT

8. Приказ Министерства просвещения Российской Федерации от 31.05.2021 № 287 «Об утверждении федерального государственного образовательного стандарта основного общего образования». http://publication.pravo.gov.ru/File/GetFile/0001202107050027?type=pdf

9. Примерная рабочая программа основного общего образования. Информатика. Базовый уровень (для 5–6 классов образовательных организаций). Одобрена решением федерального учебно-методического объединения по общему образованию, протокол 2/22 от 29.04.2022 г. https://fgosreestr.ru/oop/314

10. Примерная рабочая программа основного общего образования. Информатика. Базовый уровень (для 7–9 классов образовательных организаций). Одобрена решением федерального учебно-методического объединения по общему образованию, протокол 3/21 от 27.09.2021 г. https://fgosreestr.ru/oop/237

11. Примерная рабочая программа основного общего образования. Информатика. Углубленный уровень (для 7–9 классов образовательных организаций). Одобрена решением федерального учебно-методического объединения по общему образованию, протокол 2/22 от 29.04.2022 г. https://fgosreestr.ru/oop/313

12. *Сорокина Т. Е.* Использование метода аналогий при раннем обучении программированию // Современные информационные технологии и ИТ-образование. Сборник научных трудов II Международной научной конференции и XII Международной научно-практической конференции (Москва, 24–26 ноября 2017 года) / под ред. В. А. Сухомлина. М.: Лаборатория открытых информационных технологий факультета ВМК МГУ им. М.В. Ломоносова, 2017. С. 304–309. EDN YTGYTO.