

Федеральное агентство по образованию

Ю.П. Немчанинова

**Алгоритмизация и основы программирования
на базе KТurtle
(ПО для обучения программированию KТurtle)**

Учебное пособие

Москва 2008

- 508 Н **Немчанинова Ю.П.** Алгоритмизация и основы программирования на базе Kturtle (ПО для обучения программированию Kturtle): Учебное пособие. - Москва: 2008. - 50 с.

В пособии рассмотрены вопросы обучения программированию младших школьников на базе языка Лого. Рассматривается настройка программы Kturtle, основные команды языка Лого, реализация на языке Лого основных алгоритмических конструкций.

Рекомендуется учителям информатики, учителям начальной школы.

Оглавление

Предисловие.....	4
Введение.....	5
<i>Алгоритмы и исполнители.....</i>	<i>5</i>
<i>Свойства алгоритмов.....</i>	<i>6</i>
<i>Способы записи алгоритмов.....</i>	<i>6</i>
Словесный.....	6
Графический.....	7
Табличный	7
В виде блок-схемы.....	8
<i>Виды алгоритмов.....</i>	<i>9</i>
Линейный	9
Разветвляющийся.....	9
Циклический	11
Циклы с условием.....	13
<i>Задания для самоконтроля.....</i>	<i>14</i>
Введение в Kturtle.....	16
<i>Запуск программы.....</i>	<i>16</i>
<i>Интерфейс программы.....</i>	<i>17</i>
<i>Подготовка к работе.....</i>	<i>18</i>
<i>Настройка программы.....</i>	<i>18</i>
<i>Последовательность работы.....</i>	<i>19</i>
Программирование на языке Лого.....	21
<i>Первая программа.....</i>	<i>21</i>
Команды перемещения.....	21
Команды очистки.....	21
Управление спрайтом.....	22
Программа 1.....	22
<i>Сохранение проектов.....</i>	<i>23</i>
Задания для самоконтроля.....	24
<i>Управление пером черепашки.....</i>	<i>24</i>
Программа 2.....	25
Задания для самоконтроля.....	27
<i>Работа с холстом.....</i>	<i>27</i>
<i>Переменные в Лого. Контейнеры.....</i>	<i>29</i>
Контейнеры.....	29
Текстовые контейнеры.....	30
<i>Получение случайных чисел.....</i>	<i>30</i>
<i>Вывод данных на экран.....</i>	<i>31</i>
Организация диалога.....	31
Задания для самостоятельного выполнения.....	33
<i>Условный оператор.....</i>	<i>33</i>
Сложные условия.....	35
<i>Повторение команд.....</i>	<i>36</i>
Цикл со счетчиком.....	36
Цикл с условием.....	38
Задания для самоконтроля.....	39
<i>Подпрограммы.....</i>	<i>39</i>
Вопросы для самостоятельно контроля.....	41
Приложение 1.....	44
Глоссарий.....	47
Список литературы.....	49

Предисловие

KТurtle — это образовательная программная оболочка для изучения языка программирования **Лого**, которая позволяет программировать максимально легко и просто. Наличие визуального исполнителя позволяет сразу видеть результат выполнения программы, что очень важно при обучении программированию младших школьников. Имеется возможность писать команды как на английском, так и на русском языке. **KТurtle** может использоваться для обучения основам программирования детей как младшего, так и среднего школьного возраста. Рекомендуется использовать этот программный продукт как базовый для пропедевтического курса программирования в начальной школе (3-4 класс), а также в 5-7 классах, возможно в рамках факультативных курсов или в кружковой работе. Рекомендуемая продолжительность курса — 34 часа, 1 час в неделю.

KТurtle обладает замечательными особенностями, которые позволяют начать программировать легко и непринуждённо.

- Встроенный интерпретатор Лого устраняет необходимость скачивать и устанавливать дополнительные программы.
- Выполнение можно замедлить и остановить в любое время.
- Мощный редактор команд Лого с подсветкой синтаксиса, нумерацией строк и многим другим.
- Холст с результатами работы программы может быть сохранен как изображение или распечатан.
- Холст имеет функцию переброса Черепашки на другой край, когда она достигнет первого.
- Контекстная подсказка по всем командам Лого, которая вызывается простым нажатием F2.
- Все команды Лого могут быть переведены на любой язык.
- Имеется диалог с сообщениями об ошибках, облегчающий процесс отладки.

Введение

Алгоритмы и исполнители

Алгоритмы встречаются в реальной жизни повсюду, и мы, сами того не подозревая, постоянно сталкиваемся с алгоритмами, выполняем их и даже сами создаем. Термин «алгоритм» происходит от имени выдающегося мыслителя средневекового Востока Мухаммеда аль Хорезми (рис. 1) (VIII–IX вв.). Он стал родоначальником алгебры – математической дисциплины, где были сформулированы правила арифметических вычислений с многозначными числами в десятичной позиционной системе счисления. Позднее эти правила в Европе называли алгоритмами, от *algoritmi* – латинского написания имени аль Хорезми. С развитием науки и техники человечество осознавало, что можно научиться выполнять сложные действия, если их разбивать на последовательность простых. Слово «алгоритм» приобрело другой смысл, относящийся не только к арифметике.



Рис. 1. Аль Хорезми

Алгоритмом называется строго определенная последовательность действий, выполнение которых приводит к заранее предполагаемому результату.

Алгоритмом можно назвать рецепт приготовления какого-либо блюда, инструкцию по сборке велосипеда, формальную инструкцию для решения задачи.

Тот, кто исполняет алгоритм, называется **Исполнителем**. Исполнителем алгоритма может быть человек, автомат, наконец компьютер.

Для того чтобы исполнитель мог исполнять алгоритм, он должен «понимать» команды алгоритма.

Набор команд исполнителя, которые он может понимать и исполнять, называется Системой команд исполнителя (СКИ)

Свойства алгоритмов

Для того чтобы алгоритм был корректным (правильно исполнялся и приводил к желаемому результату), он должен обладать некоторыми свойствами.

1. **Понятность.** Исполнитель алгоритма должен знать, как его выполнять.

2. **Дискретность.** Алгоритм должен состоять из конечного числа отдельных шагов.

3. **Результативность.** Выполнение алгоритма должно закончиться выдачей результата (или сообщением о том, что результата в данной ситуации достичь не удалось.)

4. **Конечность.** Выполнение алгоритма должно когда-нибудь закончиться. (Если не следить за выполнением этого условия, можно написать такой алгоритм, который будет выполняться бесконечно, по кругу.)

5. **Однозначность.** Каждый шаг алгоритма может быть истолкован исполнителем одним единственным образом и не предусматривать возможности для двух и более способов выполнения этого шага. (Это очень важное свойство. Если алгоритм, написанный с нарушением этого свойства, будет исполнять человек, он может выбрать тот вариант исполнения, который кажется ему более правильным, и при этом вполне может ошибиться. Если же такой алгоритм попадет на исполнение к исполнителю с формальным списком команд, он просто не сможет быть исполненным).

6. **Массовость.** Желательно, чтобы алгоритм был универсальным, то есть подходил для решения всех однотипных задач.

Способы записи алгоритмов

Для записи алгоритмов можно использовать разные способы.

Словесный

Каждое действие алгоритма описывается словами.

Рассмотрим словесную запись алгоритма приготовления яичницы-глазуньи:

1. Приготовьте 2 яйца, столовую ложку растительного масла, сковороду и щепотку соли.
2. Нагрейте сковороду в течение 3 минут.
3. Положите на сковороду масло.

4. Разбейте на сковороду яйца.
5. Посолите яйца.
6. Подождите 5 минут, пока яичница не приготовится.
7. Переложите яичницу на тарелку.

Графический

Действия алгоритмов представлены в виде картинок. Графическую запись алгоритмов мы можем встретить на инструкции по сборке игрушки в киндер-сюрпризе, на пачке с чаем (инструкция по завариванию) или на упаковке с лапшой быстрого приготовления (рис. 2).



Рис. 2

Табличный

Все шаги алгоритма записываются в таблицу. В таблице отражается изменение значений входных данных, результата и промежуточных значений.

Рассмотрим табличную запись алгоритма вычисления площади прямоугольника. Для вычисления площади понадобятся длины сторон a и b , это будут входные данные, результат будет занесен в переменную S .

Рассмотрим выполнение алгоритма при $a=4$, $b=6$. (Табл. 1).



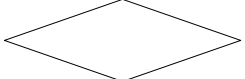

Таблица 1

№ шага	Входные данные		Результат	Примечания
	a	b	S	
1	4	6		Ввод данных
2			24	Вычисление площади
3				Вывод результата $S=24$

В виде блок-схемы

Описание алгоритма с помощью блок-схем осуществляется рисованием последовательности геометрических фигур, каждая из которых подразумевает выполнение определенного действия алгоритма. Порядок выполнения действий указывается стрелками. Внешний вид основных блоков, применяемых при написании блок-схем, приведен в табл. 2.

Таблица 2

Внешний вид блока	Пояснение
	Начало и конец алгоритма
	Действие
	Условие
	Ввод и вывод данных

Рассмотрим блок-схему алгоритма вычисления площади прямоугольника по двум его сторонам (рис. 3).

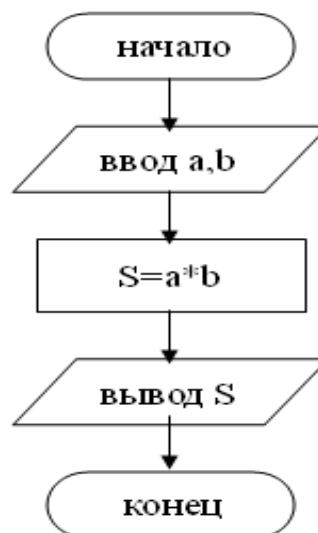


Рис. 3

Виды алгоритмов

В зависимости от последовательности выполнения действий в алгоритме выделяют алгоритмы линейные, разветвляющиеся и циклические.

Линейный

В алгоритмах линейной структуры действия выполняются однократно в заданном порядке.

Рассмотрим простой пример линейного алгоритма. Допустим, перед нами стоит задача написать алгоритм собирания портфеля. Представим данный алгоритм в словесной форме:

1. Взять дневник и выяснить расписание на завтра.
2. Положить в портфель учебники по нужным предметам.
3. Положить в портфель тетради по нужным предметам.
4. Положить в портфель пенал.
5. Положить в портфель дневник.
6. Закрыть портфель.

Рассмотренный выше алгоритм вычисления площади прямоугольника также является линейным.

Разветвляющийся

В алгоритмах разветвленной структуры в зависимости от выполнения или невыполнения какого-либо условия производятся различные последовательности действий. Каждая такая последовательность действий называется ветвью алгоритма. Следует понимать, что одновременно выполняется только одна ветвь алгоритма. Обе ветви одновременно выполняться не могут.

Рассмотрим задачу.

Винни Пух решил прогуляться в гости к Кролику. У Винни был план подкрепиться у кролика чем-нибудь вкусным, и он точно знал, что у кролика всегда есть сгущенное молоко, а иногда бывает и мед. Больше всего Винни любит мед, но в случае отсутствия меда он согласен и на сгущенное молоко. Составьте алгоритм действий для Винни.

1. Дойти до дома Кролика.
2. Поздороваться и зайти в дом.
3. Спросить у Кролика, есть ли у него дома мед.
4. Если мед есть, вежливо попросить кролика угостить гостя медом. Если же меда нет, попросить угостить гостя сгущенным

молоком.

5. Поблагодарить Кролика за угощение.

Выполняя этот алгоритм, Винни может попробовать только один десерт, либо мед, либо сгущенное молоко. Что именно ему достанется, зависит от выполнения условия, то есть от того, есть ли у кролика дома мед.

Условный алгоритм может быть представлен и в сокращенном виде. Этот вид используется в том случае, если исполнителя интересует только случай выполнения условия. Рассмотрим предыдущую задачу с немного измененным условием.

Винни Пух решил прогуляться в гости к Кролику. У Винни был план подкрепиться у кролика медом. Он точно знал, что у кролика всегда есть сгущенное молоко, а иногда бывает и мед. Но сгущенного молока Винни наелся у Кролика на прошлой неделе, и теперь он согласен только на мед. Составьте алгоритм действий для Винни.

6. Дойти до дома Кролика.

7. Поздороваться и зайти в дом.

8. Спросить у Кролика, есть ли у него дома мед.

9. Если мед есть, вежливо попросить кролика угостить гостя медом.

10. Попрощаться с кроликом.

В случае невыполнения условия Винни не будет выполнять никаких действий, а просто отправится домой.

Рассмотрим разветвляющийся алгоритм на примере поиска наибольшего числа из двух данных. Ниже приводится блок-схема алгоритма решения этой задачи (рис. 4).

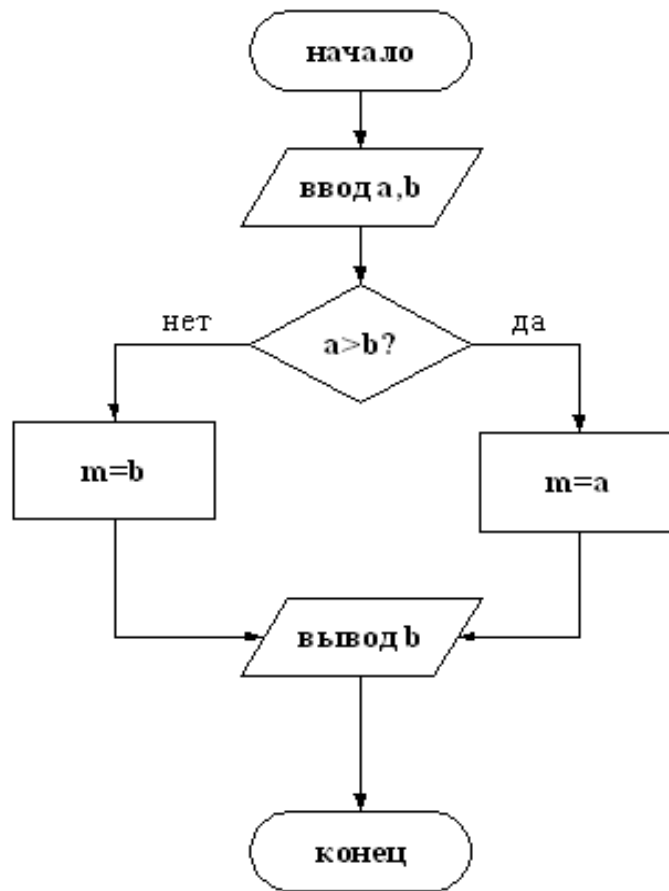


Рис. 4

Циклический

В алгоритмах циклической структуры выполняется повторяющаяся последовательность действий, называемая **телом цикла**.

Циклические алгоритмы бывают двух типов – **циклы со счетчиком** и **циклы с условием**.

Циклы со счетчиком

Этот вид цикла используется в тех случаях, когда точно известно количество повторений цикла. Для того, чтобы считать эти повторения, вводится специальная переменная – **счетчик цикла**.

Рассмотрим задачу.

Винни Пух с утра был очень голоден. Для утоления голода Винни необходимо съесть 8 пончиков, намазанных медом. Составьте алгоритм действий для Винни.

1. Сесть за стол
2. Достать пончик
3. Намазать пончик медом

4. Съесть пончик

5. Задуматься, не восьмой ли пончик был съеден только что. Если это не так — перейти к пункту 2, иначе перейти к следующему действию.

6. С сожалением вылезти из-за стола.

Очевидно, что алгоритм будет выполняться до тех пор, пока не будет съеден последний, восьмой пончик.

Рассмотрим более сложную задачу вычисления суммы десяти последовательно вводимых чисел.

Для начала определим начальное значение суммы (сумму обозначим переменной S). До того, как к сумме прибавили первое число, значение ее было равно нулю. ($S=0$). Так же необходимо определить начальное значение счетчика (обозначим его переменной i). Определим значение счетчика равным нулю ($i=0$). Каждое введенное число (поставим его в переменную a) будет последовательно прибавляться к сумме. Новое значение суммы при этом становится равным сумме старого значения суммы и числа a ($S=S+a$). Такие формулы называются рекуррентными, в них и в правой, и в левой части выражения встречается одна и та же переменная.

При помещении в переменную нового значения, старое значение просто исчезает, происходит как бы вытеснение старого значения новым. После того, как первое число добавлено к сумме, нужно обязательно увеличить значение счетчика i на единицу. Для этого используем уже знакомый нам прием - рекуррентную формулу. Новое значение переменной i будет равно сумме старого значения и единицы ($i=i+1$). После этого проверим, не пора ли заканчивать выполнение цикла. Если счетчик цикла равен десяти (т. е. цикл выполнен уже 10 раз и все 10 чисел прибавлены к сумме), то выполнение алгоритма можно закончить. В противном случае необходимо вернуться и проделать шаги заново, начиная с ввода числа в переменную a (старое значение a при этом исчезает, но оно уже было использовано программой, и никакой проблемы его пропажа не создаст). Выполнение алгоритма будет ходить по кругу до тех пор, пока значение счетчика не станет равным 10. На рис. 5 приведена блок-схема этого алгоритма.

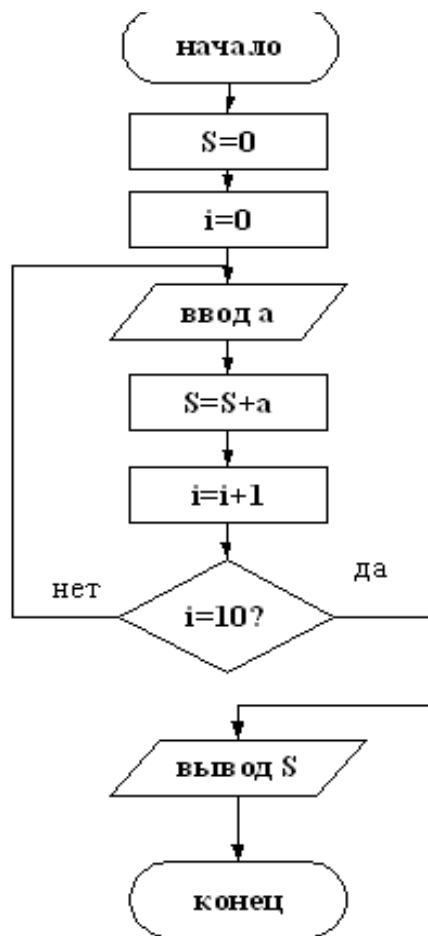


Рис. 5

Циклы с условием

Этот вид цикла используется в тех случаях, когда количество повторений неизвестно, но известно условие выполнения цикла. В этом случае цикл повторяется до тех пор, пока выполняется условие – столько раз, сколько нужно.

Рассмотрим задачу. Винни Пух очень давно не ел сладкого, и решил зайти в гости к Кролику. Винни был так голоден, что нечаянно съел у Кролика все запасы меда. Он требовал добавки до тех пор, пока мед у хозяина не закончился. Опишите алгоритм действий Винни.

1. Дойти до дома Кролика.
2. Вежливо поздороваться и войти в дом.
3. Вежливо попросить Кролика угостить друга баночкой меда.
4. Съесть угощение.
5. Поинтересоваться, есть ли еще мед. Если мед есть, вернуться к пункту 3. Если нет – перейти к следующему действию.
6. Поблагодарить за угощение.

7. Покинуть дом Кролика.

Сколько раз будет выполняться цикл зависит от того, сколько банок меда было в шкафу у Кролика.

Рассмотрим более сложную задачу. Нужно составить алгоритм для вычисления суммы чисел. Числа вводятся последовательно до тех пор, пока сумма не станет больше 50. Эта задача отличается от предыдущей тем, что мы не знаем заранее, сколько чисел потребуется ввести (рис. 6).

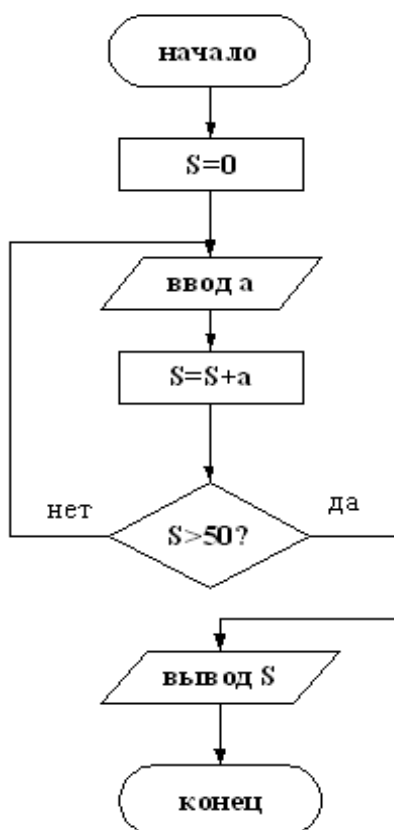


Рис. 6

Задания для самоконтроля

1. Исполнитель Трибот имеет следующую СКИ:

Вперед n (делает n шагов вперед)

Направо (поворачивается направо на 90 градусов)

Переключи (переключает положение пера из поднятого в опущенное и наоборот. По умолчанию перед началом работы перо поднято).

Составьте для Трибота алгоритмы рисования следующих фигур:

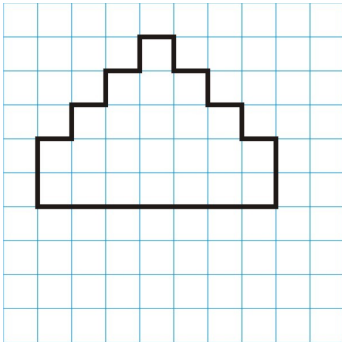


Рис.7.1

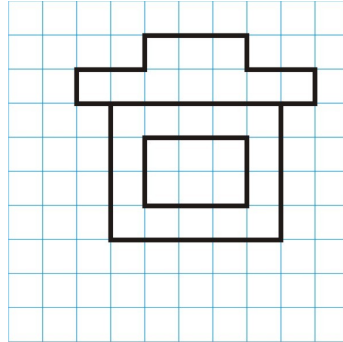


Рис. 7.2

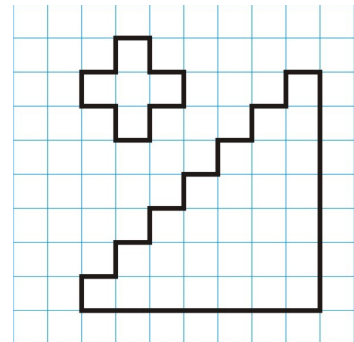


Рис.7.3

2. Придумайте своего робота-исполнителя, нарисуйте его, дайте название и определите ему собственную систему команд.
3. Представьте графическим способом алгоритм выращивания цветка из семечка.
4. Представьте графическим способом алгоритм создания снеговика.
5. Придумайте свои примеры линейных, циклических и разветвляющихся алгоритмов и запишите их любым способом.
6. Составьте алгоритм вычисления площади квадрата.
7. Составьте алгоритм выбора наименьшего из трех чисел.
8. Составьте алгоритм вычисления среднего арифметического пяти чисел. Определите вид алгоритма.
9. Составьте алгоритм нахождения произведения пяти последовательно вводимых чисел.
10. Составьте алгоритм, позволяющий дать ответ на вопрос: Сколько чисел из натурального ряда (1, 2, 3... и т. д.) нужно сложить, чтобы сумма стала больше ста?

Введение в Kturtle

Запуск программы

Запуск программы можно осуществить несколькими способами. Если используется дистрибутив **Линукс Мастер**, для запуска программы нужно в меню Пуск выбрать раздел Разное и в нем выбрать Обучение программированию на языке Лого (Kturtle) (рис. 8).

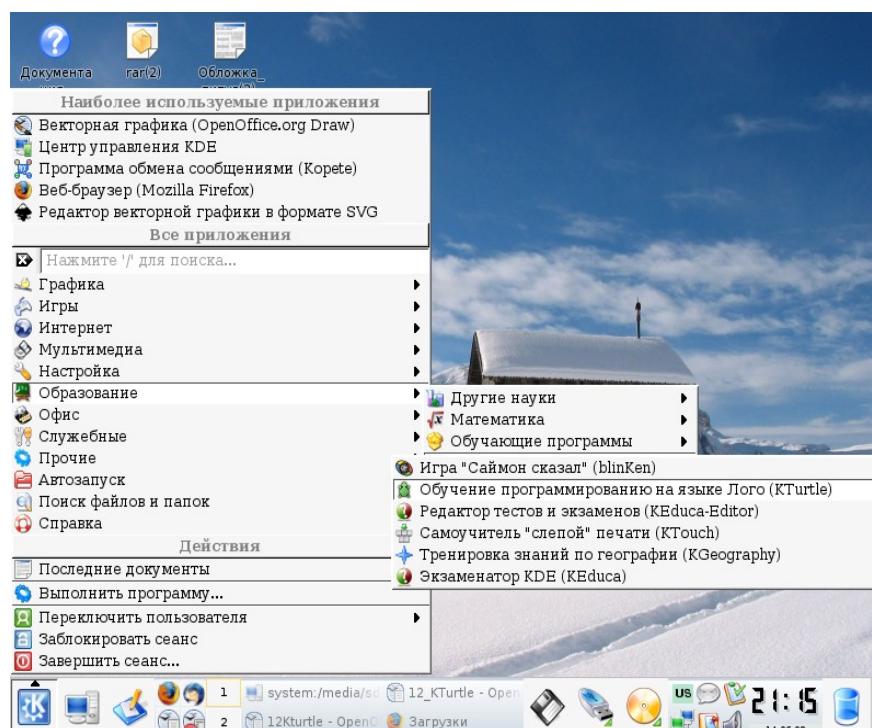


Рис. 8

Можно также запустить программу из раздела **Разработка** в группе **Образование**. Следует учесть, что если используется другой дистрибутив или дистрибутив **Линукс Мастер** настроен по другому, то запуск программы может осуществляться иначе. Можно воспользоваться менее наглядным, но зато более универсальным способом — консольным запуском программы. Для запуска консоли в **Альт-линукс Мастер** в меню **Пуск** следует выбрать раздел **Служебные** и далее **Терминал (консоль)**. В командной строке достаточно написать название редактора — KТurtle — и нажать клавишу **Enter** (рис. 9). Подробно работа с консолью описана в пособии №1 из комплекта учебных пособий.

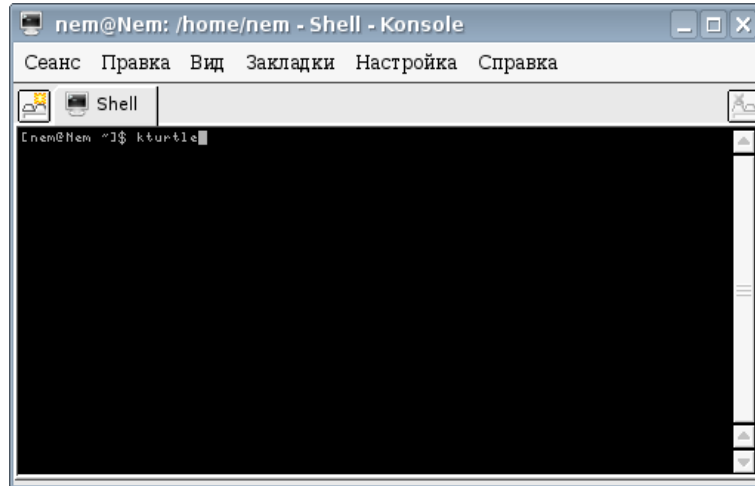


Рис. 9

Интерфейс программы

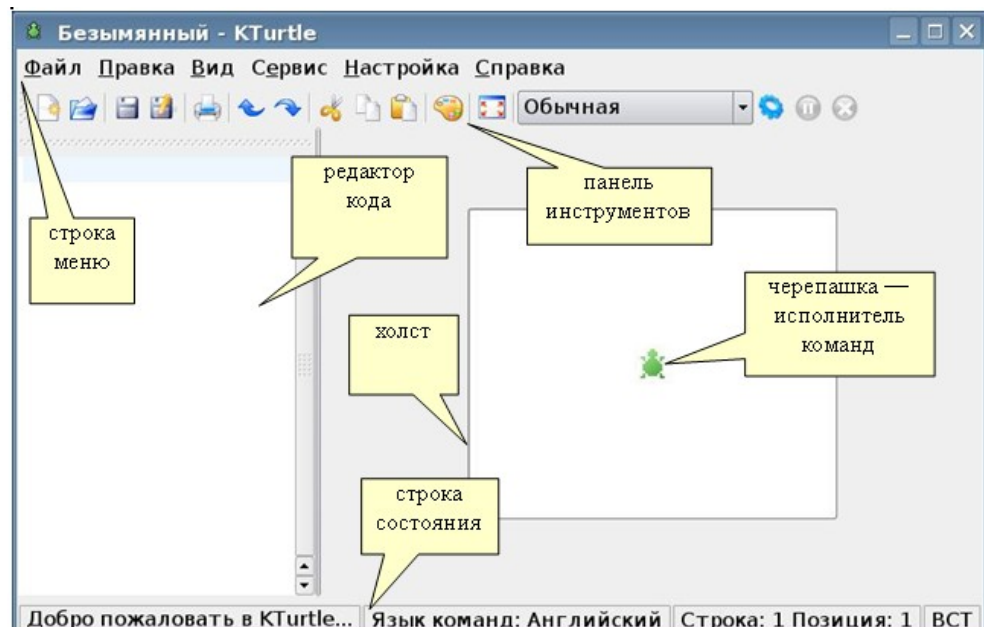


Рис. 10

Левая часть рабочей области программы (рис.10) представляет собой редактор кода. Это — территория программиста. Именно здесь будут помещаться команды для черепашки (исполнителя команд). В правой части рабочей области располагается холст. Это — территория исполнителя команд (черепашки). Выполняя команды, расположенные в области редактора кода, черепашка перемещается по холсту, рисует, и делает еще много интересного.

Также в главном окне есть меню, из которого производится управление программной оболочкой, панель инструментов, которая предо-

ставляет быстрый доступ к часто используемым командам, и панель состояния, на которой отображается различная информация KТurtle. Подробное описание команд меню можно найти в Приложении 1 данного пособия. А сейчас мы выясним, что же нужно сделать, чтобы написать первую программу для черепашки.

Подготовка к работе

Перед тем, как мы напишем свою первую программу для черепашки, необходимо выполнить некоторые действия и разобраться с основными приемами работы. Как вы уже поняли, программа пишется в левой части экрана, в редакторе кода. Редактор кода устроен таким образом, чтобы свести к минимуму ошибки программиста. Так, команды, в случае правильного написания, выделяются зеленым цветом, это дает возможность быстро заметить синтаксические ошибки. Программа для черепашки состоит из отдельных команд, практически каждая из которых имеет два варианта написания: полное и сокращенное. Вы можете использовать любой вариант.

Настройка программы

В разделе Настройка вы можете изменить некоторые параметры.

Как уже упоминалось ранее, Черепашка понимает команды как на русском, так и на английском языке. Всё дальнейшее описание рассчитано на русский вариант, поэтому перед началом работы рекомендуется выбрать русский язык. Для установки русского языка команд необходимо в меню Настройка выбрать пункт Настроить KТurtle (рис.11).

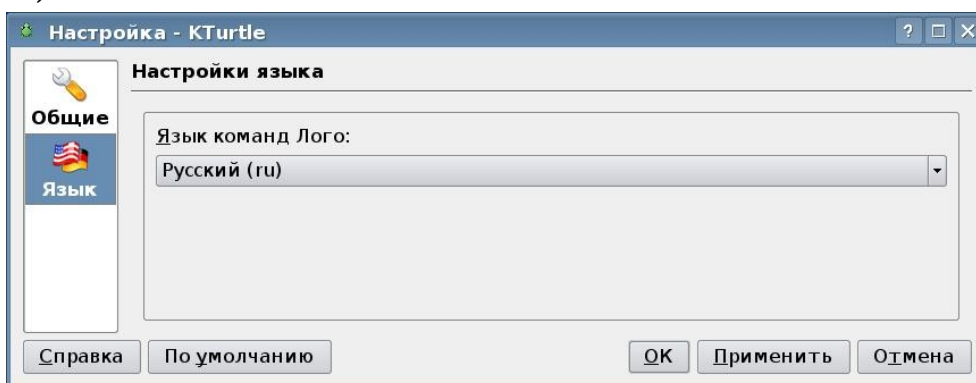


Рис. 11

Во вкладке **Язык** установите язык команд Лого **Русский**.

Во вкладке **Общие** вы можете изменить размеры холста, увеличив или уменьшив размеры, данные по умолчанию (рис.12).

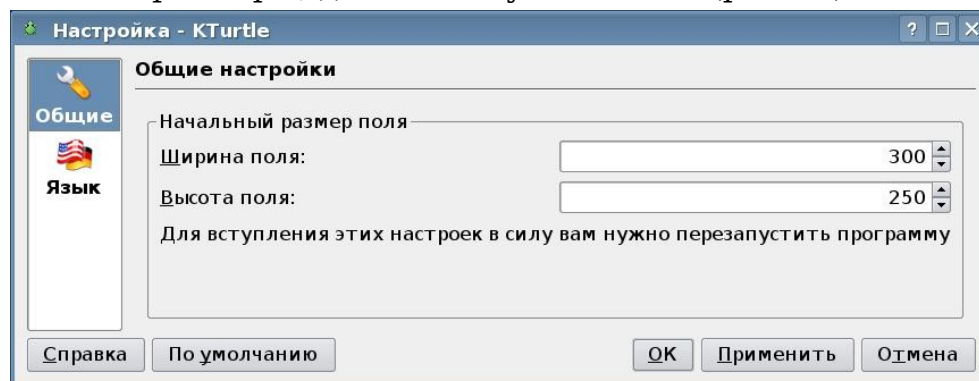


Рис. 12

Последовательность работы

После того, как вы напишете в редакторе кода примерный вариант программы, вы должны дать черепашке команду исполнить программу. Сделать это можно из меню, выбрав в разделе Файл пункт Выполнить сценарий (рис. 13). То же самое можно сделать, используя панель инструментов (просто нажать на кнопку «выполнить сценарий»). Третий вариант этого же действия — одновременное нажатие клавиш **Alt** и **Enter**.

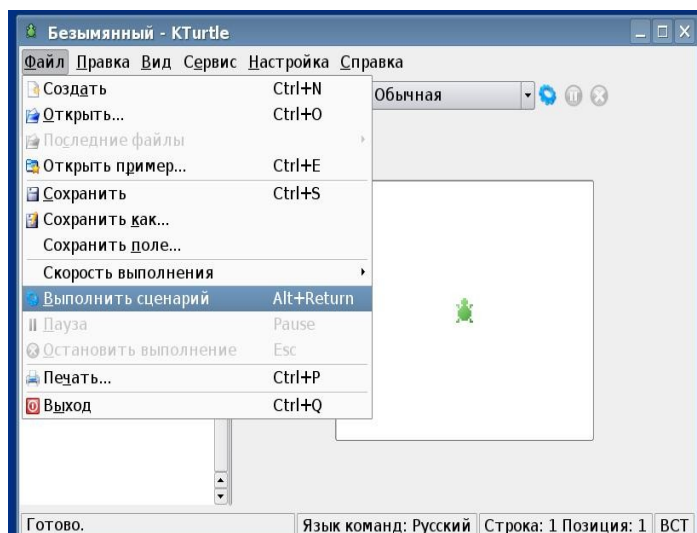


Рис. 13

Используя панель инструментов или меню, можно задать нужную скорость для черепашки — от обычной до самой медленной (рис. 14).

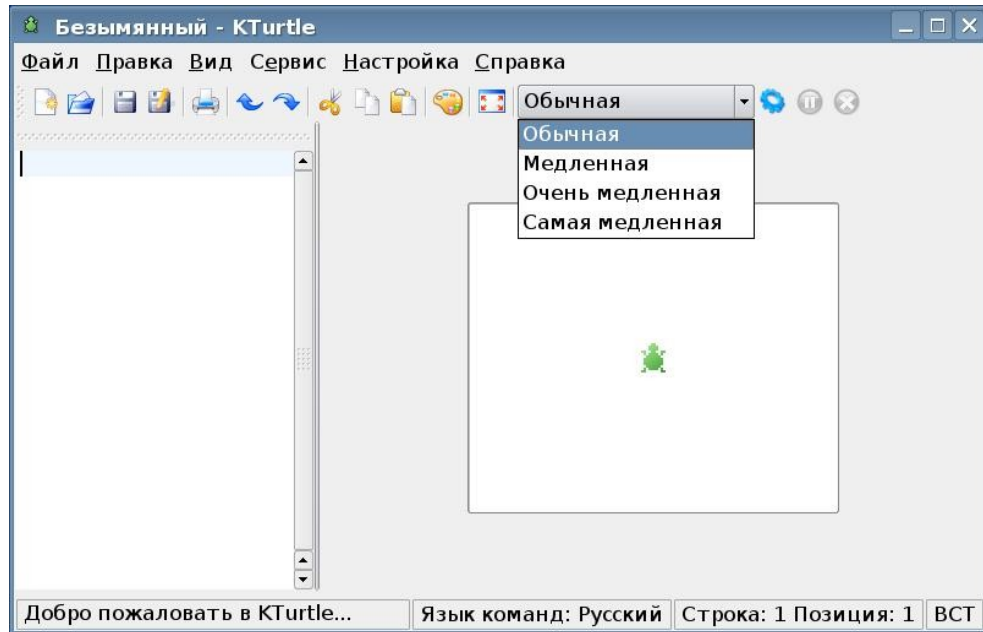


Рис. 14

После того, как вы запустите сценарий на исполнение, произойдет одно из двух возможных действий. В первом случае в вашей программе будут обнаружены ошибки, и вы получите сообщение об этом. Нужно вернуться в редактор кода и исправить ошибку, после чего снова попытаться выполнить сценарий. Это придется выполнять до тех пор, пока сценарий не будет выполнен без ошибок. Этот процесс называется **отладкой программы**.

Может случиться и так, что в вашей программе сразу не окажется ни одной ошибки. В этом случае черепашка немедленно исполнит все команды, и вы увидите на холсте результат.

Помните, что вы всегда можете внести изменения в вашу программу и запустить ее на исполнение заново.

Теперь мы знаем достаточно, чтобы попытаться написать свою первую программу на Лого!

Программирование на языке Лого

Первая программа

Как известно, любая программа состоит из команд. Прежде всего, мы должны изучить несколько самых простых команд языка Лого. Начнем с изучения движения черепашки. Черепашка может перемещаться вперёд и назад на заданное число шагов, а так же поворачиваться налево и направо на заданную градусную величину угла. Важно: Черепашка не может двигаться боком, только головой или хвостом вперед. Для того чтобы она поползла в нужную сторону, необходимо сначала развернуть ее в нужном направлении.

Команды перемещения

Итак, запомните следующие команды черепашки:

вперёд (вп)

вперёд X

Перемещает черепашку на X пикселей вперёд. Когда перо опущено, черепашка будет оставлять за собой след. Эта команда также может записываться как вп.

назад (нд)

назад X

назад перемещает черепашку назад на X пикселей. Когда перо опущено Черепашку будет оставлять за собой след. Может так же записываться как нд.

налево (лв)

налево X

Предписывает черепашке повернуть на X градусов налево. Может записываться и как лв.

направо (пр)

направо X

Предписывает черепашке повернуть на X градусов направо. Может записываться и как пр.

Мы уже выяснили, что программа редко получается с первого раза. Следовательно, для достижения результата нам придется выполнять ее несколько раз. Необходимо выяснить, как убрать с поля последствия неудачных экспериментов черепашки.

Команды очистки

Существуют две команды очистки холста.

очисти (очс)

Этой командой вы можете очистить холст от всех следов. Все остальное останется по-прежнему: позиция и угол направления черепашки, цвет холста, видимость черепашки и размер холста. Может записываться и как очс.

сброс

Очищает более объёмно, нежели команда очисти. После выполнения этой команды всё будет выглядеть так, как будто вы только что запустили KТurtle. Черепашка будет расположена в центре экрана, цвет холста будет белым, цвет пера черепашки будет черным.

Управление спрайтом

Иногда сама черепашка мешает разглядеть рисунок, созданный на холсте. В этом случае ее можно спрятать:

спрячь (сч)

Скрывает Черепашку. Это полезно, когда Черепашка не уместна в ваших рисунках. Может записываться и как сч.

покажи (пж)

Делает Черепашку видимой после того, как она была скрыта. Также может записываться как пж.

Программа 1

Теперь мы знаем достаточно, чтобы написать и отладить первую программу для черепашки. В результате выполнения программы у черепашки должен появиться дом, изображенный на рис. 15.

Приступим!

Текст программы будет выглядеть следующим образом:

сброс

вперёд 50

налево 90

вперёд 50

налево 90

вперёд 50

налево 90

вперёд 50

налево 90

вперёд 50

налево 45

вперёд 35

налево 90

вперёд 35

спрячь

Если вы всё сделали правильно, после выполнения сценария на холсте появится дом (рис.15).

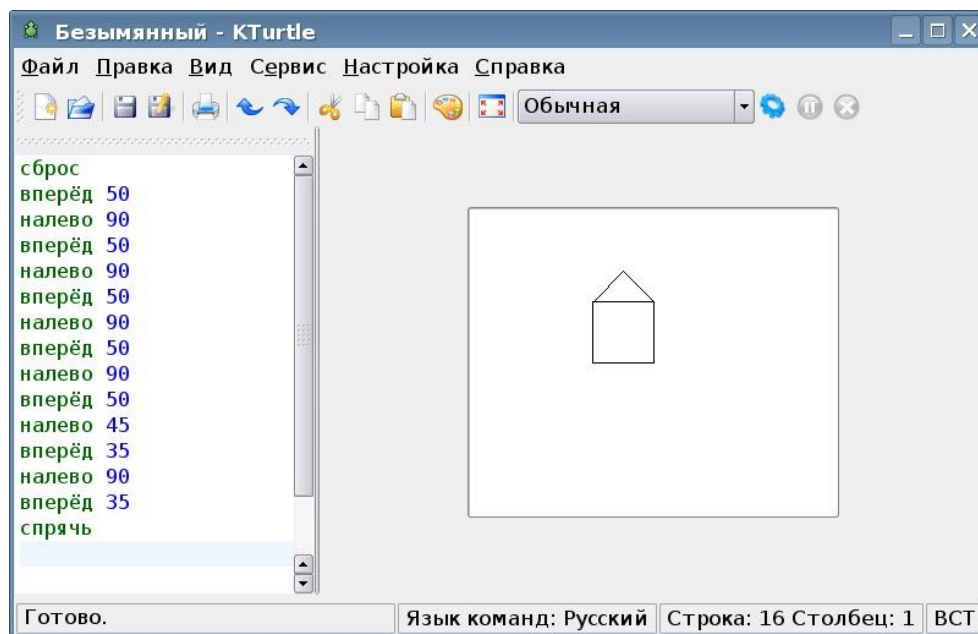


Рис.15

Сохранение проектов

Для того чтобы сохранить программу, нужно выполнить следующие действия:

В меню **Файл** выберите пункт **Сохранить**. В открывшемся окне укажите место, в которое вы будете сохранять ваш проект. Файлы с программами имеют расширение **Лого**. Вы можете сохранить таким образом недоделанную программу, а потом вернуться к ней и доделать. Для этого в меню **Файл** нужно выбрать пункт **Открыть**, после чего в открывшемся окне указать место, где была сохранена ваша программа.

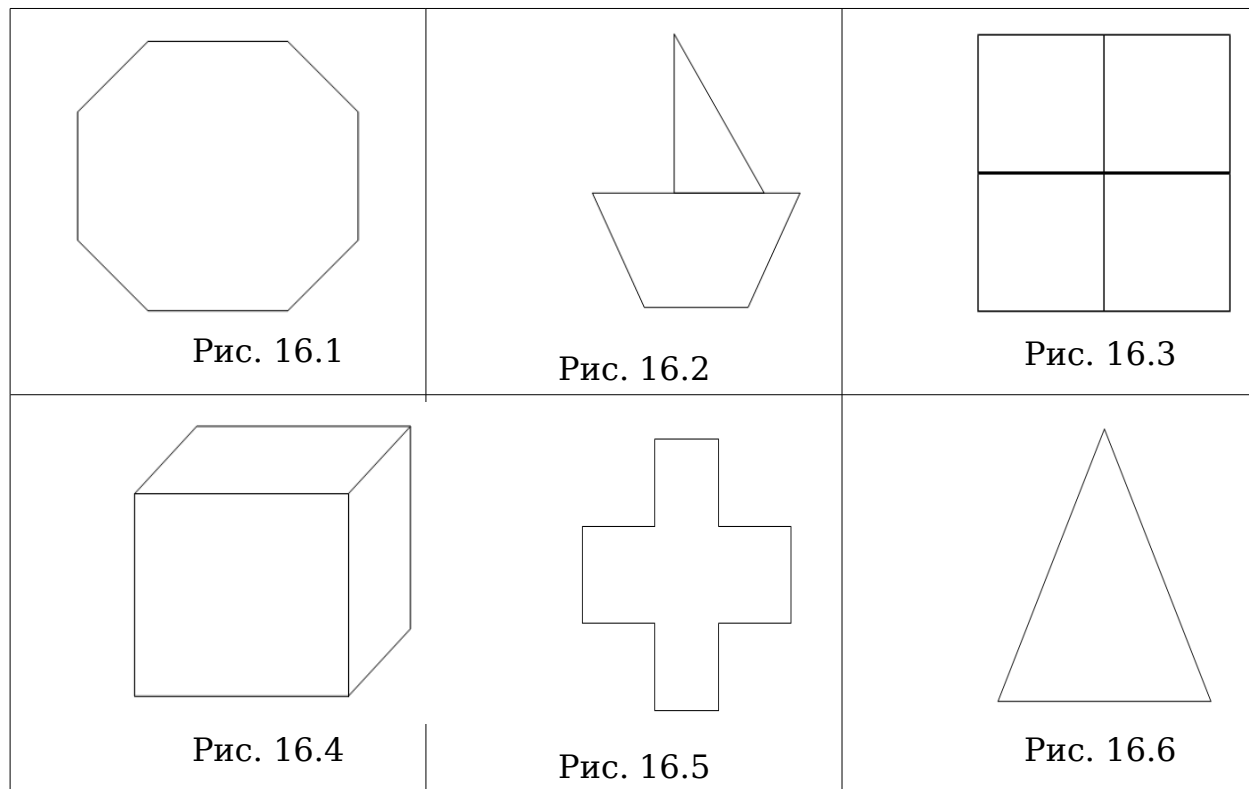
Помимо самой программы, вы можете сохранить отдельно и результат ее работы, т.е. картинку. Для этого в меню **Файл** выберите **Сохранить поле**. Картинка сохраняется со стандартным расширением **png** и может в дальнейшем быть просмотрена любой программой-просмотрщиком, а также выведена на печать.

Теперь, когда первая программа написана, можете с полным правом называть себя программистами и для закрепления результата по-

пробуйте выполнить задания, представленные на рис.16 .

Задания для самоконтроля

Напишите программу, выполняя которую черепашка рисует следующую фигуру:



Управление пером черепашки

Наша цель — научиться управлять пером черепашки. До сих пор мы просто использовали простейшие возможности пера, оно всего лишь оставляло след за черепашкой. У него есть и другие возможности.

перо_подними (пп)

Отрывает перо от холста. Пока перо оторвано, черепашка не будет рисовать линию во время перемещений. Может записываться и как **пп**.

перо_опусти (по)

перо_опусти

Опускает перо на холст. Когда перо опущено, черепашка рисует линию при перемещениях. Может записываться и как **по**.

нов_ширина_пера (ншп)

нов_ширина_пера X

нов_ширина_пера устанавливает ширину пера (ширину линии) в X

пикселей. Может записываться и как **ншп**.

нов_цвет_пера (нцп)

нов_цвет_пера R,G,B

нов_цвет_пера устанавливает цвет пера. В качестве параметров указывается интенсивность красной, зеленой и синей составляющих цвета (0..255). Может записываться и как **нцп**.

Очень удобно выбирать цвет пера, пользуясь значком в виде палитры на панели инструментов.

Нужно просто нажать на этот значок, выбрать нужный цвет и скопировать его числовой код в текст программы (рис. 17). Подробнее о цветовых палитрах можно узнать из пособия №16 из комплекта учебных пособий (Создание и редактирование векторных изображений с помощью Inkscape).

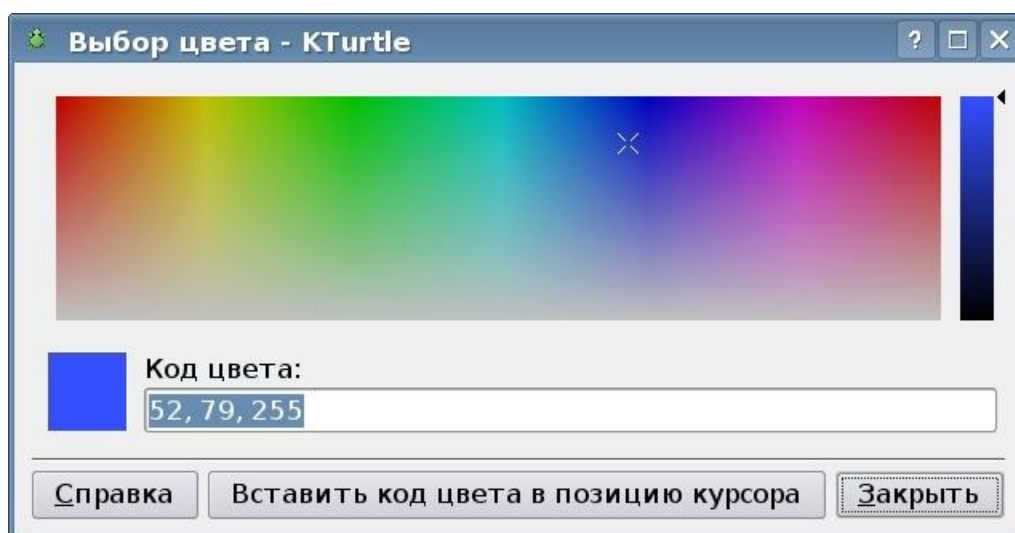


Рис. 17

Таким образом, у нас есть возможность перемещать черепашку по холсту без рисования, менять цвет и толщину пера. Используя новые знания, мы можем добавить к черепашьему домику окно и раскрасить его! (рис. 18)

Программа 2

сброс

нов_ширина_пера 3

нов_цвет_пера 65, 255, 65

вперёд 50

налево 90

вперёд 50

налево 90
вперёд 50
налево 90
вперёд 50
налево 90
вперёд 50
нов_цвет_пера 255, 61, 27
налево 45
вперёд 35
налево 90
вперёд 35
перо_подними
налево 45
вперёд 10
налево 90
вперёд 10
перо_опусти
нов_цвет_пера 52, 79, 255
вперёд 30
направо 90
вперёд 30
направо 90
вперёд 30
направо 90
вперёд 30
направо 90
спрячь

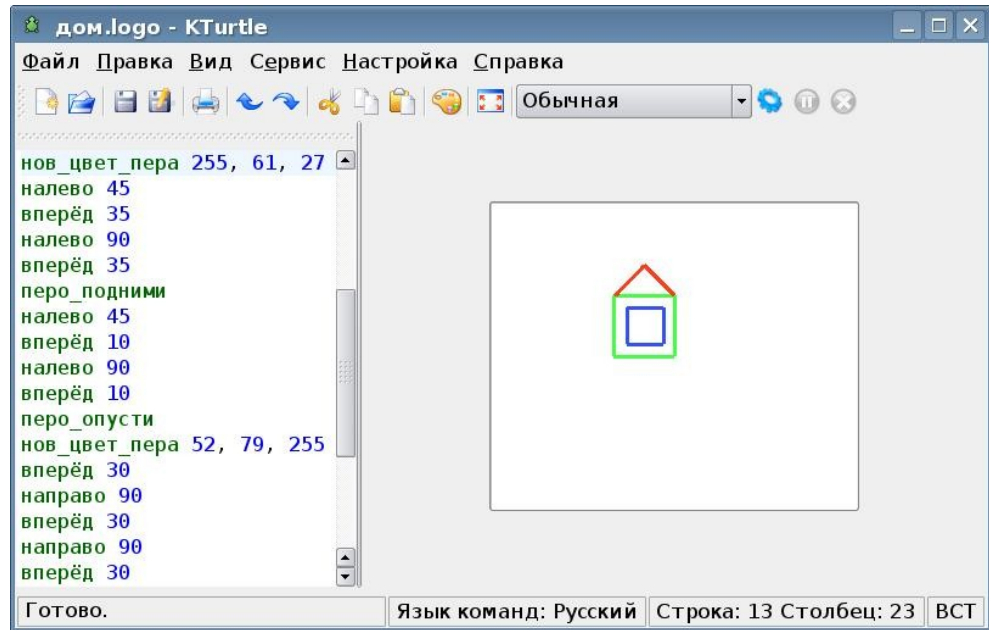
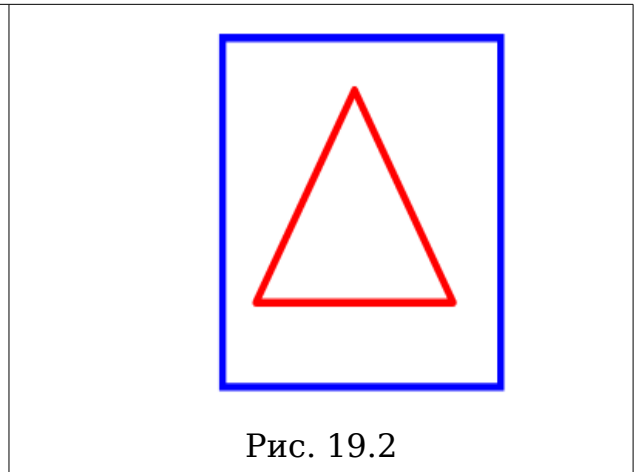
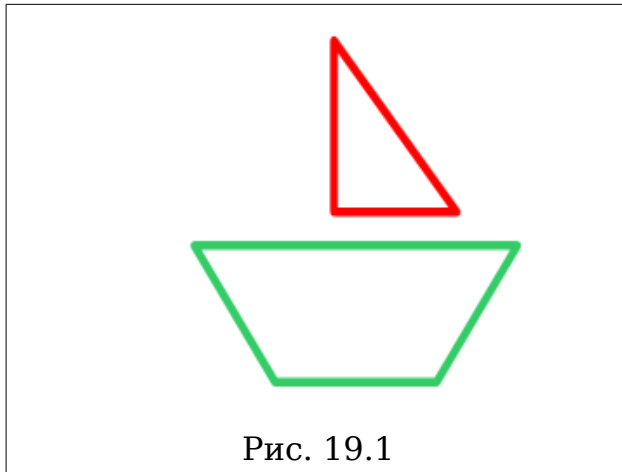


Рис. 18

Для закрепления выполните следующие упражнения.

Задания для самоконтроля

Напишите программу, выполняя которую черепашка рисует следующую фигуру (рис. 19):



Работа с холстом

Существует несколько команд для работы с холстом.

нов_размер_холста (нрх)

нов_размер_холста X,Y

С помощью этой команды можно поменять размер холста. В качестве входных параметров задаются ширина X и высота Y в пикселях. Может записываться и как **нрх**.

нов_цвет_холста (нцх)

нов_цвет_холста R,G,B

нов_цвет_холста устанавливает цвет холста. Входными параметрами является комбинация RGB. Может записываться и как **нцх**.

обёртка_вкл

Этой командой вы устанавливаете обёртку холста. Это значит, что при достижении края холста черепашка не исчезнет, а окажется на его противоположной стороне. Включение обёртки как бы делает поле замкнутым, а точнее — закольцованным. По полю со включенной обёрткой черепашка может ходить по кругу бесконечно долго.

обёртка_выкл

Этой командой вы отключаете обёртку холста. Это значит, что черепашка, подойдя к краю холста, может выйти за его границы и “исчезнуть”.

Для того чтобы понять, как именно работает обёртка, попробуйте протестировать работу двух программ:

Сброс обёртка_вкл вперёд 1000	Сброс обёртка_выкл вперёд 1000
-------------------------------------	--------------------------------------

В первом случае черепашка несколько раз проползет по холсту снизу вверх. Во втором — доползет до верхнего края поля и исчезнет за его пределами (но команда **сброс** легко вернет ее на место).

Используя команды работы с холстом, вы можете поместить дом черепашки на зеленую лужайку и расширить пространство вокруг дома. (рис. 20).

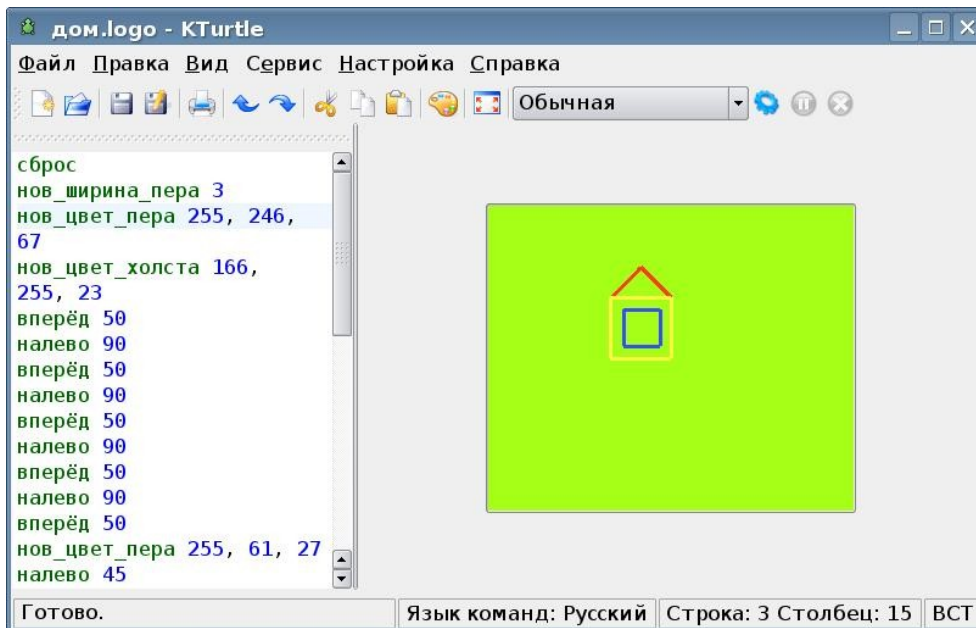


Рис. 20

Переменные в Лого. Контейнеры

Контейнеры

Контейнеры— это символы или слова, которые используются программистом для хранения чисел или текста. Контейнеры, содержащие числа, называются переменными, а содержащие текст — строками.

По умолчанию все контейнеры являются пустыми(т.е. содержат ноль) до тех пор, пока в них не будет помещено некоторое значение.

Например, запись $M=6$ означает, что переменной M присвоено значение 6. Это значение будет храниться в переменной M до тех пор, пока в нее не будет помещено новое значение. Переменной могут присваиваться не только числовые значения, но и арифметические выражения. Например: $M=3+2$. В результате этого действия переменной M будет присвоено значение 5.

Черепашка может выполнять различные арифметические действия как с числами, так и с другими переменными. Вы можете складывать (+), вычитать (-), умножать (*) и делить (/). Ниже представлены примеры использования этих операций.

$$a=34$$

$$b=18+a$$

$$c=(a+b)/2$$

$$d=23-a$$

Текстовые контейнеры

Строкой называется последовательность алфавитных символов, заключенных в кавычки. Например : «Доброе утро, Черепашка» является строкой. Строка так же может быть помещена в контейнер. Контейнер, содержащий строку называется **текстовым контейнером**. Например, $s = \text{«Привет»}$ - текстовый контейнер s содержит строку «привет».

В отличие от числовых значений, над строками нельзя производить арифметические действия. Исключением является сложение. При сложении двух строк происходит их «склеивание», начало второй строки присоединяется к концу первой. Например, в случае представленного ниже кода в контейнере s будет содержаться строка «автомобиль».

$a = \text{«авто»}$

$b = \text{«мобиль»}$

$c = a + b$

Важно: если вместо $c = a + b$ в представленном выше примере написать $c = b + a$, результатом будет строка «мобильавто».

Научившись работать со строками и текстовыми контейнерами, мы получим возможность организовывать с черепашкой диалог, т е обмениваться с ней текстовыми фрагментами. Но для этого нужно знать, каким образом Черепашка выводит данные на экран.

Получение случайных чисел

Переменной можно присвоить случайное число, выбранное из определенного диапазона.

случайное

случайное X,Y

Случайное — команда, которая имеет входные и выходные параметры. На входе требуются два числа, первое (X) задаёт нижний порог получаемых чисел (минимум), второе (Y) задаёт верхний порог (максимум). Выходной параметр - это псевдослучайное число, которое лежит в интервале(X,Y).

Например, после выполнения команды

$a = \text{случайное } 1,100$

переменной a будет присвоенное некоторое случайное число из интервала от 1 до 100.

Вывод данных на экран

Черепашка может напечатать на экране любой текст, а также содержимое любого контейнера. Для этого используется команда `напиши`.

Вы можете написать на экране любые строки и числа, комбинируя их при помощи знака `+` (рис. 21).

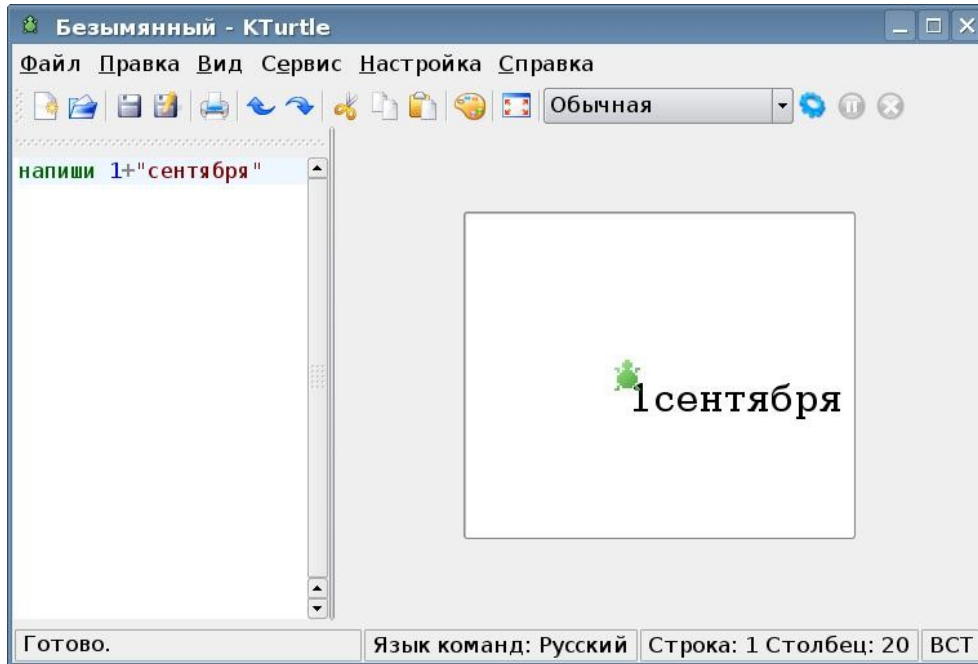


Рис. 21

Размер шрифта можно изменить, используя команду

нов_размер_шрифта

нов_размер_шрифта X

Устанавливает размер шрифта, используемого для печати. Входной параметр один, он должен быть числом. Размер задается в пикселях.

Организация диалога

Пользователь может обмениваться данными с черепашкой через диалоги. Диалог реализуется при помощи двух команд:

сообщение

сообщение X

Команде **сообщение** требуется передать строку (X), она будет показана в появившемся окне (рис. 22).

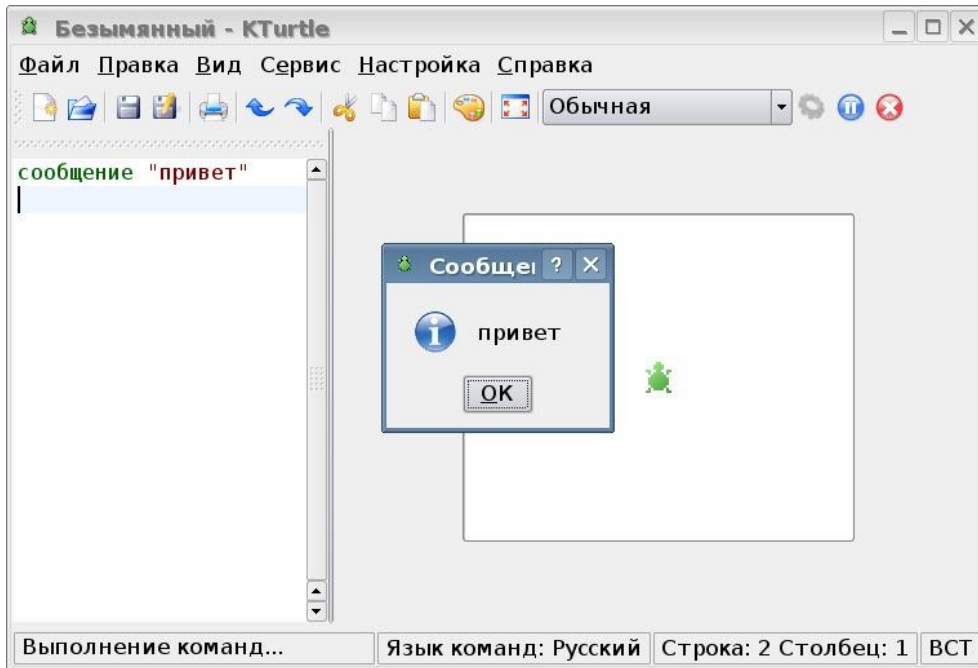


Рис. 22

окно_вопроса

окно_вопроса X

Команде **окно_вопроса** требуется передать строку, она будет показана в появившемся окне, аналогично команде **сообщение**. Но, кроме этого, в окне будет поле для ввода числа или текста (рис. 23, 24).

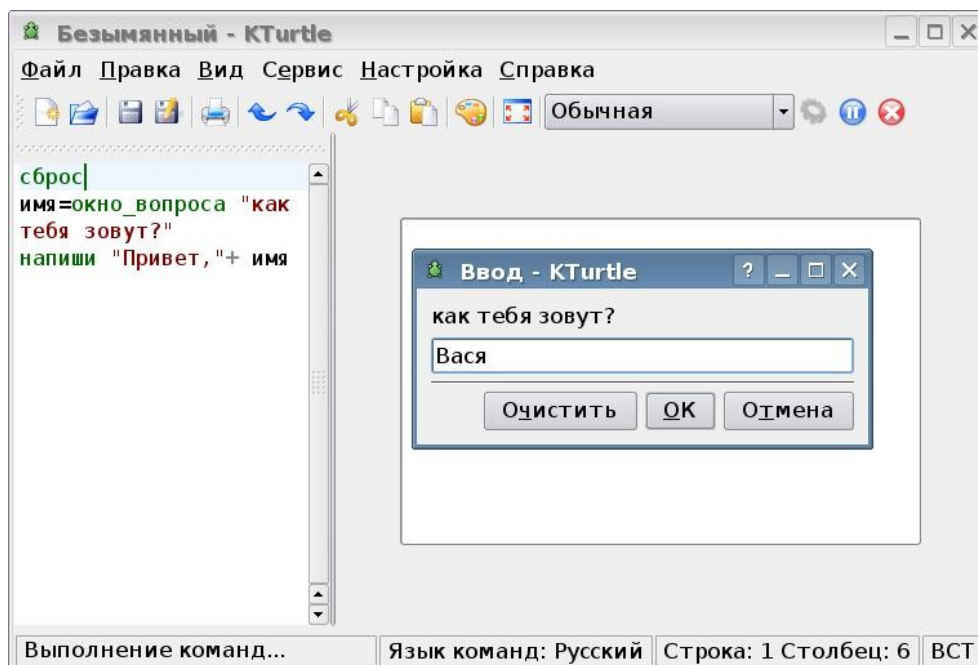


Рис. 23

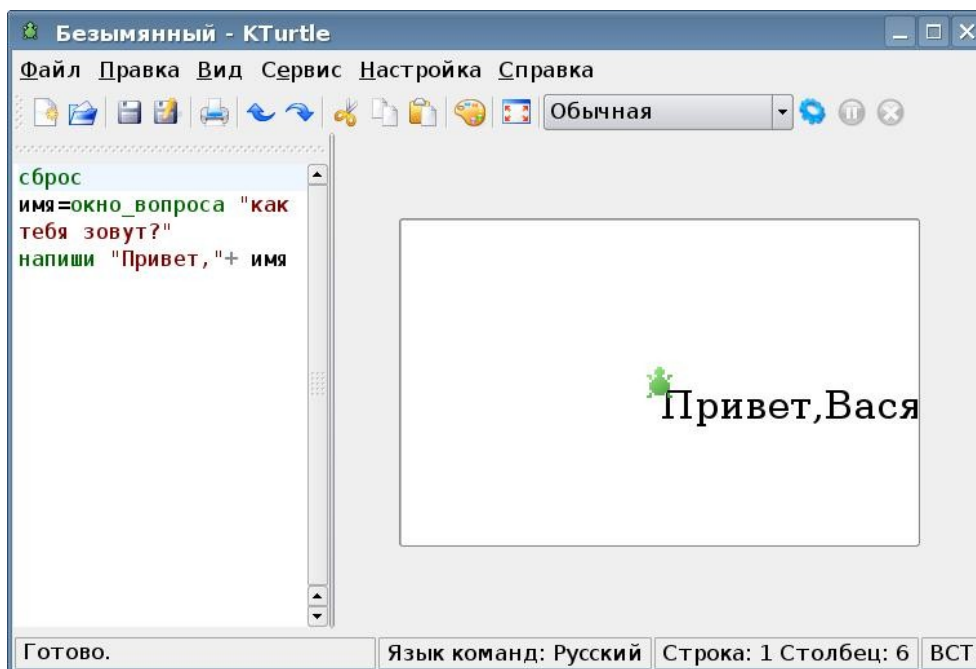


Рис. 24

Задания для самостоятельного выполнения

1. Напишите программу, которая спрашивает возраст пользователя, и вычисляет, сколько лет ему осталось до совершеннолетия
2. Напишите программу для решения следующего уравнения:
 $5+x=28$
3. Напишите программу, которая спрашивает возраст у трех пользователей и вычисляет их средний возраст
4. Напишите программу, которая печатает на экране 3 случайных числа из интервала от 5 до 15
5. Напишите программу, выполняя которую Черепашка чертит на поле квадрат, длина стороны которого — случайное число из интервала от 10 до 20
6. Напишите программу, выполняя которую Черепашка чертит на поле квадрат со стороной, заданной пользователем путем диалога.

Условный оператор

Иногда в программе необходимо выполнить ту или иную последовательность действий, в зависимости от некоторого условия. Если условие выполняется, будет произведена одна группа действий, а если не выполняется — то другая. Очевидно, что невозможно выполнить сразу обе группы действий, в каждом конкретном случае будет выполнена только одна группа.

Для того, чтобы организовать в программе выбор в зависимости от условия, используется команда если.

если условие [первая группа команд]

иначе [вторая группа команд]

Вместо условия нужно поместить *логическое выражение*.

Логическое выражение — это такое выражение, про которое точно можно сказать, истинно оно или ложно. Совершенно точно известно, что $5 > 6$ — ложь, $3 = 8$ — ложь, а $4 < 8$ — истина. Эти выражения являются логическими и могут быть использованы в качестве условия. Правильная запись условий представлена в таблице 3.

Таблица 3

$a == b$	a равно b
$a != b$	a не равно b
$a > b$	a больше b
$a < b$	a меньше b
$a >= b$	a больше или равно b
$a <= b$	a меньше или равно b

Пример

Черепашка получает от пользователя через диалог два числа. Ей нужно выбрать большее из этих чисел и сообщить об этом пользователю.

Текст программы будет выглядеть следующим образом:

a =окно_вопроса «введи первое число»

b =окно_вопроса «введи второе число»

если $a > b$ [

 напиши a

]

иначе [

 напиши b

]

На рис. 25 представлен запрос программы на ввод числа. На рис. 26 представлен результат выполнения программы при $a=66$ и $b=35$.

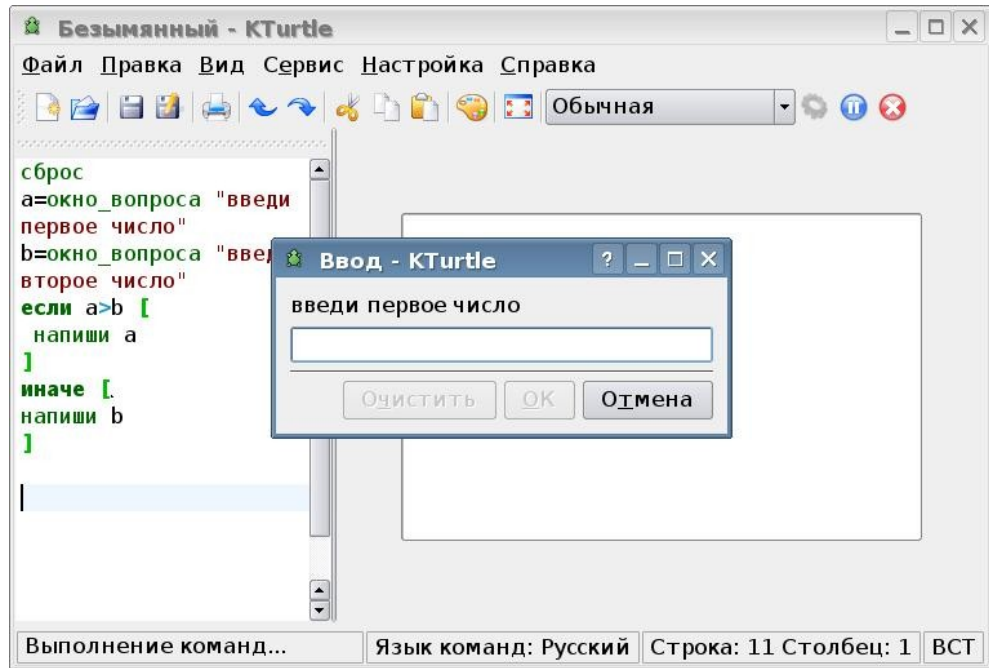


Рис. 25

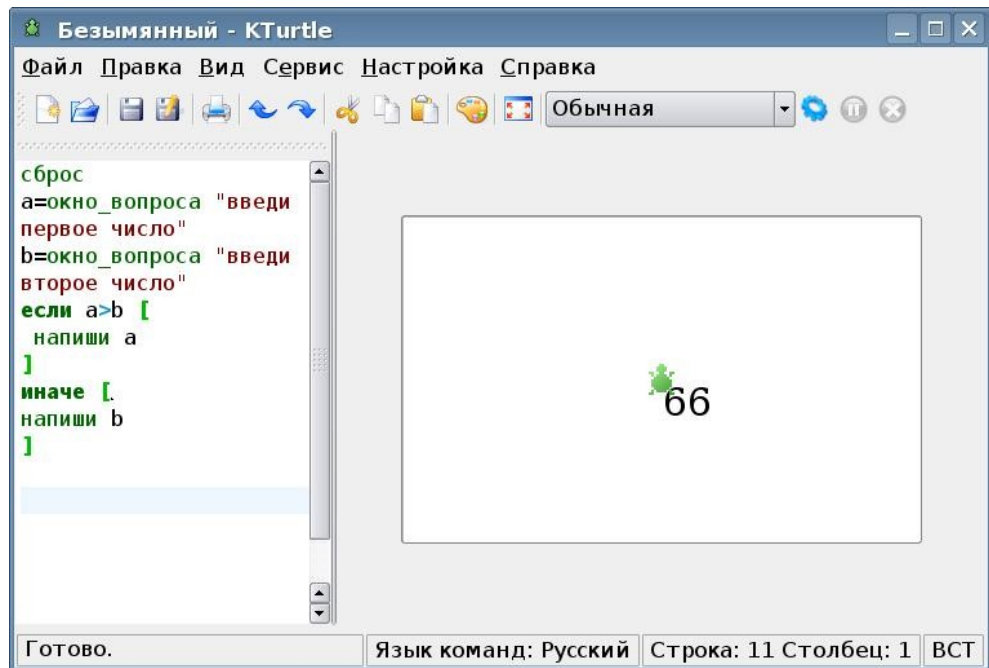


Рис. 26

Сложные условия

Можно задавать сложные условия, состоящие из простых логических выражений и логических операций **и**, **или** и **не**.

В случае, если логические выражения соединены операцией **и**, обе стороны должны быть равны «истина» для получения ответа «истина» на весь вопрос (Табл. 4).

Таблица 4

А	В	А и В
ложь	ложь	ложь
ложь	истина	ложь
истина	ложь	ложь
истина	истина	истина

Если используется операция **или**, хотя бы одна сторона должна быть равна «истина» для того, чтобы ответ был «истина».

Таблица 5

А	В	А или В
ложь	ложь	ложь
ложь	истина	истина
истина	ложь	истина
истина	истина	истина

Если используется операция **не**, значение выражения меняется на противоположное. Истина превратится в ложь, а ложь — в истину.

Таблица 6

А	не А
ложь	ложь
истина	истина

Повторение команд

Часто бывает необходимо выполнить одну и ту же последовательность команд несколько раз подряд. Вы уже сталкивались с этим при рисовании, например, квадрата. Для того, чтобы не писать много раз одно и то же, используются циклы. Самый простой вариант цикла мы можем использовать в том случае, если нам точно известно количество повторений.

Цикл со счетчиком

для начальное число **до** конечное число [...]

Для - это цикл «со счётчиком».

Например:

для $x = 1$ до $4[$
вперёд 40

направо 90
]

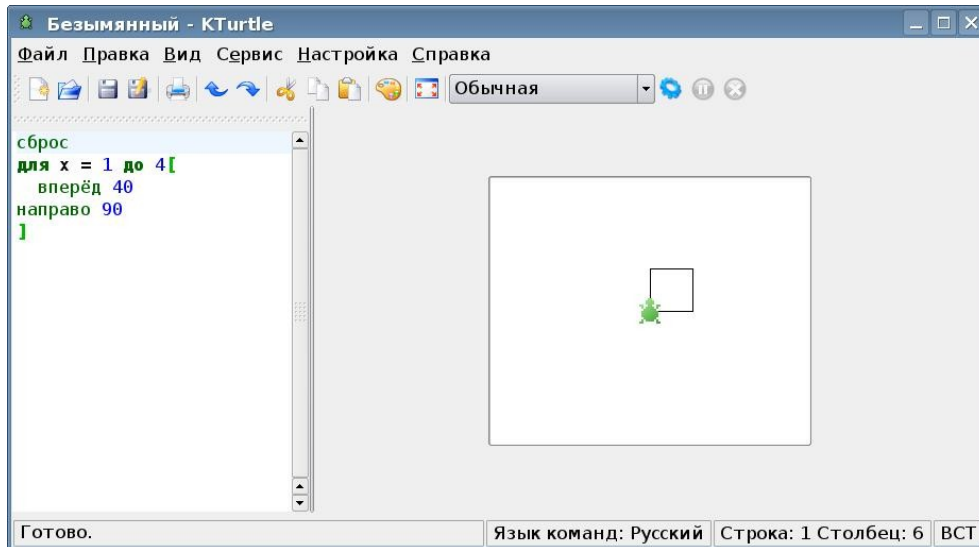


Рис. 27

Каждый раз, когда выполняется код в скобках, значение x увеличивается на 1, и так до тех пор, пока x не станет равным 4. После завершения выполнения программы на холсте появится квадрат (см.рис. 24)

Если заставить черепашку двигаться очень маленькими шажками и после каждого шага поворачиваться на очень маленький угол в одну и ту же сторону, можно получить нарисованный круг (рис. 28).

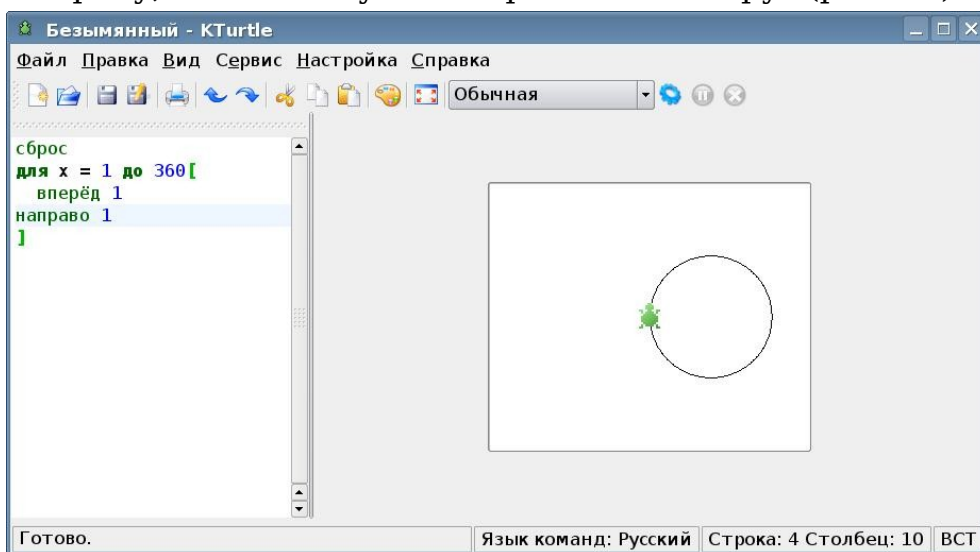


Рис. 28

Если делать шаги и повороты больше, получатся разнообразные многоугольники (рис.29 и 30).

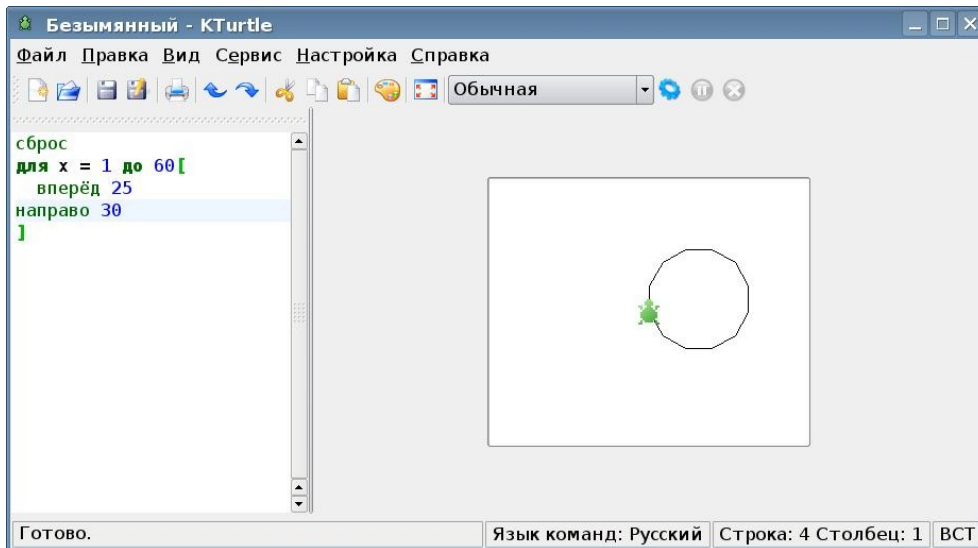


Рис. 29

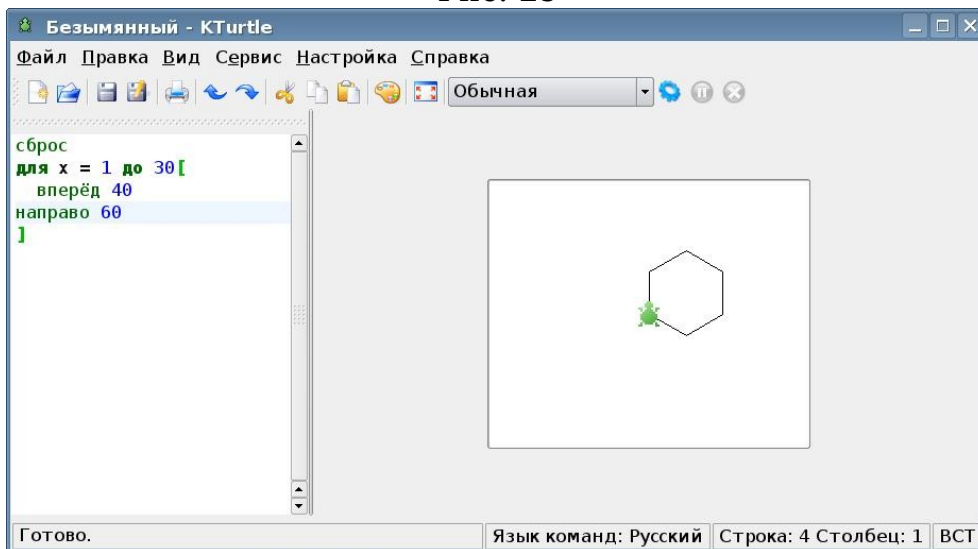


Рис. 30

Цикл с условием

Цикл с условием используется в том случае, если количество повторений заранее не известно. Но зато известно условие, при котором будет выполняться цикл. Повторения будут выполняться до тех пор, пока условие выполняется.

Цикл "пока"

пока вопрос [...]

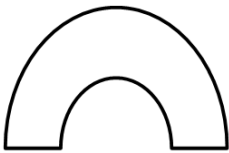

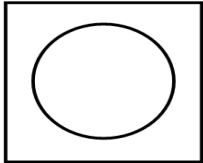
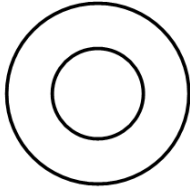
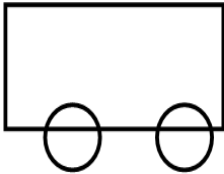
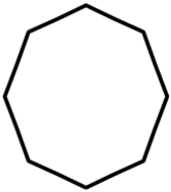
Управляющий оператор **пока** очень похож на **если**. Разница в

том, что пока будет повторять код, расположенный между скобками, до тех пор, пока ответом на вопрос не станет «ложь».

Закрепите свои знания, выполнив следующие задания.

Задания для самоконтроля

Напишите программу, выполняя которую Черепашка рисует следующую фигуру (рис. 31):

 Рис. 31.1	 Рис. 31.2	 Рис. 31.3
 Рис. 31.4	 Рис. 31.5	 Рис.31.6

Подпрограммы

Черепашка имеет достаточно большой набор команд. Однако, этим ее возможности не ограничиваются. Есть возможность научить черепаху новым командам. Для того, чтобы Черепашка могла использовать новую команду наравне с остальными, необходимо ее «научить». Для этого используется команда **выучи**.

Создаваемые вами команды могут принимать входные параметры и возвращать различные значения.

Рассмотрим, как можно научить черепаху рисовать квадрат. Для того, чтобы нарисовать квадрат, нужно знать один параметр — длину стороны. Обозначим этот параметр буквой *x*, а команду для рисования квадрата так и назовем — квадрат (Вы можете выбрать любое название для вашей команды).

```
выучи квадрат x[
повтори 4 [
вперед x
направо 90
```

Теперь команду **квадрат** можно использовать наравне с другими, вместо x используя любое число.

Использование подпрограмм позволяет избежать ненужной работы, состоящей в написании одинаковых групп операторов.

Вопросы для самостоятельного контроля

1. Строго определенная последовательность действий, выполнение которых приводит к заранее запланированному результату называется...

- а) программой
- б) алгоритмом

2. Укажите правильную запись команды

- а) вперед 50
- б) вперёд50
- в) вперёд 50
- г) вперёд 5o

3. Укажите правильную запись команды

- а) пр 90
- б) вправо 90
- в) вправо90

4. К какому типу алгоритма можно отнести следующий фрагмент:

для x=1 до 4[
вперёд 10
направо 90
]

- а) линейный
- б) циклический
- в) разветвляющийся

5. К какому типу алгоритма можно отнести следующий фрагмент:

вперёд 10
направо 90
вперёд 10
направо 90
вперёд 10
направо 90

- а) линейный
- б) циклический
- в) разветвляющийся

6. Какое значение окажется в переменной `s` после выполнения фрагмента программы

```
a= «каз»  
s= «за»  
e= «ак»  
s=a+e  
а) казак  
б) заказ  
в) ак
```

7. Какую геометрическую фигуру нарисует Черепашка, выполнив следующий фрагмент программы:

```
для x=1 до 4[  
вперёд 30  
направо 90  
]  
а) круг  
б) линию  
в) ступеньки  
г) квадрат
```

8. Какую геометрическую фигуру нарисует Черепашка, выполнив следующий фрагмент программы:

```
для x=1 до 4[  
вперёд 30  
влево 90  
]  
а) программа не будет работать  
б) линию  
в) ступеньки  
г) квадрат
```

9. Укажите строку, содержащую `ghfd` запись:

```
а) a=случайное число  
б) a=случайное(20,100)  
в) a=случайное 20,100  
г) a=случайное 20
```

10. Что делает программа, приведенная ниже?
`a=окно_вопроса "введи первое число"`

```
b=окно_вопроса "введи второе число"  
если a<b [  
  напиши a  
]  
иначе [  
  напиши b  
]  
а) выбирает меньшее из двух чисел  
б) выбирает большее из двух чисел  
в) выдает ошибку
```

Ключ

1Б 2В 3А 4Б 5А 6А 7Г 8А 9В 10А

Приложение 1

Описание пунктов меню

Меню **Файл**

Создать

Файл->Создать (Ctrl-N)

Создаёт новый, пустой файл Лого.

Открыть

Файл->Открыть... (Ctrl-O)

Открывает Лого файл.

Последние файлы

Файл->последние файлы

Открывает файлы Лого, которые недавно открывались.

Открыть примеры

Файл->Открыть примеры (Ctrl-E)

Показывает папку, в которой расположены файлы примеров Лого.

Для того, чтобы эти файлы открылись на вашем языке необходимо предварительно выбрать его из пункта Настройка KТurtle меню Настройка.

Выполнить команды

Файл->Выполнить команды (Alt-Return)

Запускает на выполнение команды Лого, введенные в редакторе кода.

Сохранить

Файл->Сохранить (Ctrl-S)

Сохраняет текущий открытый файл Лого.

Сохранить как

Файл->Сохранить как...

Сохраняет текущий открытый файл Лого в выбранном месте.

Сохранить поле

Файл->Сохранить поле

Сохраняет текущие рисунки на холсте в файл изображения.

Пауза

Файл->Пауза (Pause)

Приостанавливает выполнение

Остановить выполнение

Файл->Остановить выполнение (Escape)

Прекращает выполнение команд. Этот пункт доступен только когда команды выполняются.

Скорость выполнения

Файл->Скорость выполнения

Если скорость 'Обычная' (по умолчанию), трудно уследить за тем, что происходит на холсте. С помощью этого пункта можно замедлить скорость выполнения, чтобы проследить за каждым его шагом.

Печать

Файл->Печать... (Ctrl-P)

Печатает либо код, находящийся в редакторе, либо рисунки на холсте.

Выход

Файл->Выход (Ctrl-Q)

Выход из KТurtle.

Меню Правка

Правка->Отменить действие (Ctrl-Z)

Отменяет последние изменения в коде. KТurtle имеет неограниченное число отмен!

Правка->Повторить (Ctrl-Shift-Z)

Возвращает отмененные изменения в коде.

Правка->Вырезать (Ctrl-X)

Вырезает выделенный текст из редактора в буфер обмена.

Правка->Копировать (Ctrl-C)

Копирует выделенный текст из редактора в буфер обмена.

Правка->Вставить (Ctrl-V)

Вставляет текст из буфера обмена в редактор.

Правка->Найти... (Ctrl-F)

Поиск выражений в тексте.

Правка->Продолжить поиск(F3)

Поиск следующего вхождения фразы в тексте.

Правка->Заменить... (Ctrl-R)

Позволяет произвести замену текста в редакторе.

Меню Вид

Вид->Полноэкранный режим (Ctrl-Shift-F)

Позволяет переключаться в полноэкранный режим.

Примечание: В полноэкранном режиме, во время выполнения команд, все, кроме холста, скрыто. Это позволяет создавать "полноэкранные" программы в KТurtle.

Вид->Показать нумерацию строк (F11)

Позволяет включать/выключать показ нумерации строк в редакторе. Может быть полезно при поиске ошибок.

Меню Сервис

Сервис->Выбор цвета (Alt-C)

Вызывает инструмент выбора цвета. Используя color picker можно легко подобрать нужный цвет и вставить его цифровое представление в код программы.

Сервис->Снять отступ

Снимает все отступы на выделенных строках.

Сервис->Закомментировать (Ctrl-D)

Данная команда добавляет символы комментария (#) в выделенную строку. Все, следующее за символом комментария до конца строки, игнорируется при выполнении кода. Комментарии позволяют программисту пояснить назначение того или иного куска кода или скрыть от интерпретатора часть кода, чтобы он не выполнялся.

Сервис->Раскомментировать (Ctrl-Shift-D)

Удаляет символы комментария с выбранной строки.

Меню Настройка

Настройка->Показать/Скрыть панель инструментов

Управляет отображением панели инструментов

Настройка->Показать/Скрыть строку состояния

Управляет отображением строки состояния.

Настройка->Расширенные настройки

Здесь можно настроить опции, которые обычно в настройке не нуждаются. Подменю Расширенные настройки имеет три пункта: Настроить редактор (стандартный диалог настроек редактора Kate), Комбинации клавиш (стандартный диалог настроек KDE) и Настройка панели инструментов (стандартный диалог настройки панели инструментов KDE).

Настройка->Настройка KТurtle...

Применяется для конфигурирования KТurtle. Здесь можно поменять язык команд Лого или установить новый исходный размер холста.

Меню Справка

Справка->Руководство KТurtle

Вызывает руководство по KТurtle.

Справка->Что это? (Shift-F1)

Контекстная справка

Справка->Помощь по ... (F2)

вызывает помощь по тому элементу кода, рядом с которым находится курсор в редакторе.

Справка->О KТurtle

Здесь вы найдёте информации об авторах KТurtle и лицензиях, с которыми она распространяется.

Глоссарий

А

Алгоритм - строго определенная последовательность действий, выполнение которых приводит к заранее предполагаемому результату.

Д

Диалог — способ организации обмена данными между программой и пользователем.

Г

Градусы - единицы измерения углов или поворотов. Полный разворот - это 360 градусов, половина разворота - это 180 градусов и четверть разворота - 90 градусов. Входными параметрами команд налево, направо и направление являются углы в градусах.

И

Исполнитель алгоритма — это некоторая абстрактная или реальная (техническая, биологическая или биотехническая) система, способная выполнить действия, предписываемые алгоритмом

К

Контейнеры- это символы или слова, которые используются программистом для хранения чисел или текста.

Л

Логическое выражение — выражение, относительно которого точно можно сказать, истинно оно или ложно.

О

Отладка программы — процесс выявления в программе скрытых или явных ошибок и их исправления. Ошибки в программе бывают двух типов — синтаксические и логические. Первые приводят к тому, что запуск программы оказывается невозможен до их исправления, как правило, это ошибки в написании команд или параметров. Наличие в программе логических ошибок приводит к тому, что программа работает неправильно. Эти ошибки не может найти компьютер, их может выявить только программист.

П

Палитра RGB

RGB-комбинации используются для описания цветов. “R” отвечает за красный, “G” за зелёный и “B” за синий цвета. Например, рассмотрим комбинацию 255,0,0: первое число, отвечающее за красный, равно 255, а два остальных равны 0, это говорит о том, что данная комбинация передаёт чистейший красный цвет. Каждая составляющая комбинации лежит в диапазоне от 0 до 255.

П

Переменная - величина, которая в ходе выполнения программы может менять своё значение. В лого переменными называются контейнеры, содержащие числа.

Пиксель - точка на экране. Если вы посмотрите на экран с очень близкого расстояния, вы увидите, что ваш монитор использует пиксели. Пиксель - наименьшая частица, которая может быть нарисована на экране. Множеству команд требуется количество пикселей в качестве входных параметров.

Подпрограмма - часть программы, которая может быть вызвана из основной программы.

Программа - алгоритм, записанный на языке, понятном компьютеру, т е на языке программирования.

С

Случайное число - число, которое случайным образом выбирается из заданного интервала. На самом деле, ничего случайного в компьютере не происходит, и за выбор этого числа отвечает специальный алгоритм. Поэтому правильнее называть эти числа псевдослучайными.

Спрайт - это небольшая картинка, перемещаемая по экрану. Наша Черепашка, к слову, является спрайтом.

Строка - называется последовательность алфавитных символов, заключенных в кавычки

Список литературы

Использованная литература

1. <http://edu.kde.org/kturtle/> Сайт проекта
2. Макарова Н.В. Программа по информатике (системно-информационная концепция). СПб.:Питер, 2001
3. Информатика. Методическое пособие для учителей. 7 класс. Под ред. Н.В. Макаровой. СПб.:Питер, 2001
4. Начала информатики. Язык Лого/ Под ред. Б.Сендова. - М: Наука, 1990
5. <http://computerhistory.narod.ru>. Виртуальный музей вычислительной техники

Рекомендуемая литература

1. http://community.livejournal.com/logo_ru/ logo. Первые шаги в программировании. Виртуальное сообщество
2. <http://school.ort.spb.ru/library/logo/>
3. <http://el.media.mit.edu/logo-foundation/index.html>
4. <http://www.geocities.com/CollegePark/lab/2276/>