Методические рекомендации по теме
«Модуль turtle: рисование полных и неполных окружностей»

Цель:

- дать представление о рисовании окружностей с помощью векторной графики модуля Turtle в языке Python.

Задачи:

- знакомство с приемами рисования окружностей в векторной графике;
- программирование действий исполнителя на языке Python;
- анализ программного кода с целью определения, что выведет программа при конкретных исходных данных;
- исправление ошибок и дописывание программного кода;
- написание программного кода.

Планируемые результаты

Личностные: обучающиеся получат навыки активной коммуникации в группе, осознанной ориентировки в мире ИТ профессий, постановки собственных образовательных целей и задач, владение первичными навыками анализа и критичной оценки получаемой информации.

Предметные: обучающиеся получат представления о рисовании о расчете размеров и рисовании окружностей с помощью модуля Turtle в языке Python.

Метапредметные: обучающиеся получат возможность владения общепредметными понятиями «градусы», «исполнитель», «окружность», «векторная графика», «пиксель», «радиус». «диаметр»; информационно-логическими умениями; умениями самостоятельно планировать пути достижения целей; владения умениями принятия решений и

осуществления осознанного выбора; повышения уровня ИКТ – компетентности и расширение кругозора в области информатики и программирования; знакомство с профессиональной деятельностью программиста в рамках ранней профориентации; развитие интеллектуальных способностей, а также логического и критического мышления.

Материалы к занятию
Приложение 1: Сценарный план видеоролика
Приложение 2: Домашние задание и практика
Приложение 3: Краткие организационно-методические рекомендации по организации работы на занятии

Ход проведения урока

1. Организационный момент.
Мотивация на учебную деятельность.
Приветствие учащихся, сообщение темы и целей занятия.

2. Вводный блок.
Тема.
Преподаватель при необходимости останавливая трансляцию, комментируя дополнительно тему занятия.
Проблемная дискуссия по вопросам:

- Как нарисовать окружность с помощью исполнителя?
- Могут ли градусы быть отрицательными?
- 270 градусов и - 90 градусов для исполнителя одно и тоже или нет?
- Куда будет смотреть исполнитель если задать ему поворот на 0 градусов?

Итоги дискуссии (обобщаются преподавателем и фиксируются ответы учеников на доске, чтобы вернуться к ним и оценить правильность предположений учеников на этапе рефлексии):

- Градусы могут быть как положительными, так и отрицательными
- Направление исполнителя в 0 градусов – это вправо
- Рисовать окружность можно как замкнутый многоугольник, но правильнее использовать специальную команду

см. сцены **1 – 2 (здесь и далее приводится **Таблица «Содержание видеоролика». Приложение 1**).*

3. Блок повторения.

Блиц-опрос.

Преподаватель предлагает ученикам ответить на 5 вопросов по предыдущей теме; задания выполняются в сопровождении видеоролика с использованием таймера; ученики выполняют задания, голосуют, обсуждают результаты. Процедура голосования определяется инструкцией; учитель должен убедиться, что всем понятна процедура голосования. Преподаватель может поставить ролик на паузу и обсудить результаты голосования; объяснить правильный ответ руководствуясь материалами предыдущего занятия

см. сцены **3 – 7*

4. Практический блок.

Векторная графика

Работа с модулем turtle подразумевает в больше степени выполнение практических графических проектов, поэтому освоение нового материала организовано в формате выполнения и разбора заданий с теоретическими вставками для объяснения основных понятий.

Для организации **практической работы** ученики занимают рабочие места и запускают Python (терминал IDLE) на своих компьютерах. Для выполнения практической работы используются материалы видеоролика:

- Подключение модуля turtle

- Рисование кругов
- Рисование снеговика
- Несколько кругов
- Рисовать полукольца

После демонстрации каждого задания запускается таймер. Время работы таймера определяется сложностью задания. До завершения работы таймера ученики выполняют задания на компьютерах.
После завершения работы таймера демонстрируется разбор задания. Ученики останавливают работу и обсуждают разбор задания.
*см. сцены ***11 – 22*** *(кроме сцен с теорией)*

Практические задания разделены **теоретическими вставками,** необходимыми для работы над проектами урока:
- Синтаксис команды для рисования окружности объект.circle
- Рисование неполной окружности
- Управление пером (поднять, опустить)
- команда seth, которая разворачивает объект в нужном направлении

*см. сцены ***8 – 10, 12, 18-19***

По итогам работы ученики получают объекты, созданные с помощью векторной графики.
При необходимости преподаватель может поставить ролик на паузу и дать дополнительные пояснения по материалу; если ответы на вопросы вызывают у учеников затруднения, преподаватель может вывести нужную сцену ролика на экран для помощи ученикам.

5. Рефлексия. Сообщение домашнего задания.

Завершаем демонстрацией ролика и кратким обобщением материалов занятия. Преподаватель возвращается к зафиксированным в ходе дискуссии в начале урока предположениям учеников и обсуждает насколько их предположения были правильными, делаются выводы.
Преподаватель дает ученикам домашнее задание к следующему занятию (*Приложение 2*).
см. сцена* ***23.*

Приложение 1

Сценарный план видеоролика

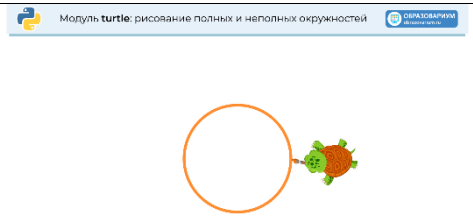
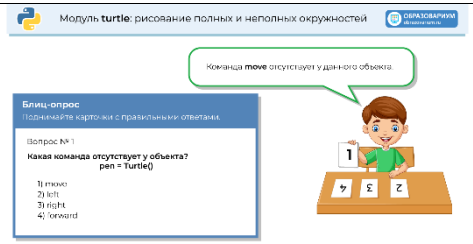
В таблице «Содержание видеоролика» представлены:

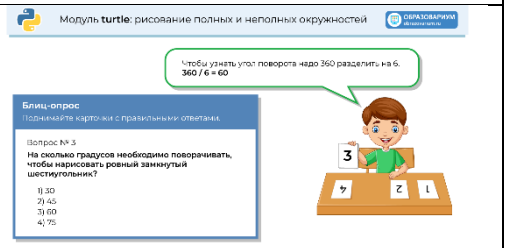
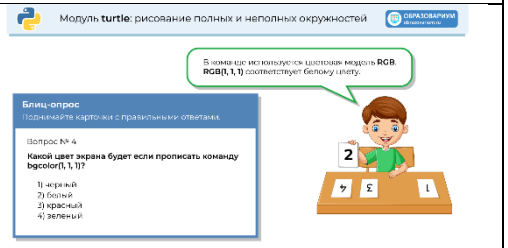
- название блоков видеоролика (тайминг);
- краткое описание содержания в каждом блоке;
- фрагменты из видеоролика, относящиеся к соответствующему блоку;
- номера сцен в каждом блоке.

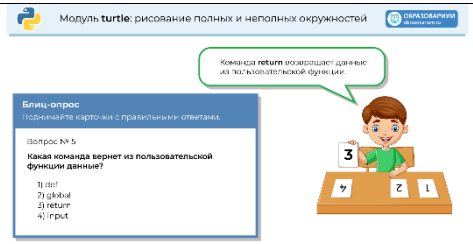
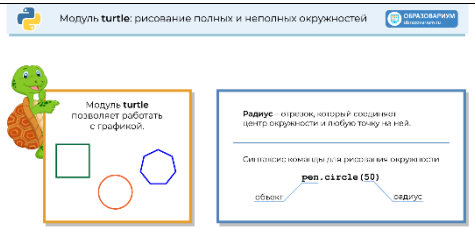
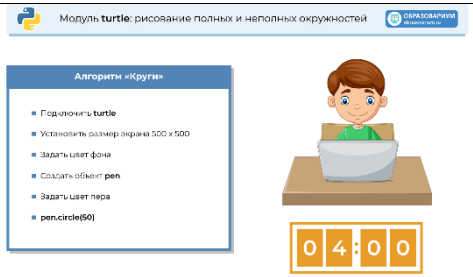
Учитель при подготовке к уроку может ознакомиться с содержанием видеоролика в текстовом формате, при необходимости распечатать фрагменты текста или примеры заданий и задач для использования в работе с учениками. Распечатанные тексты и задания из таблицы также можно применять в качестве раздаточного материала как на уроке, так и для домашних заданий.

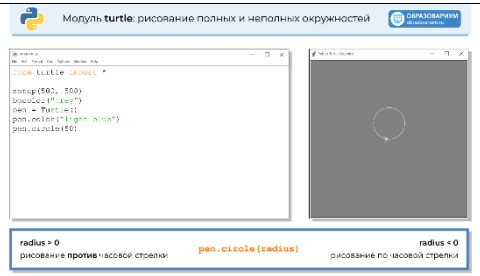
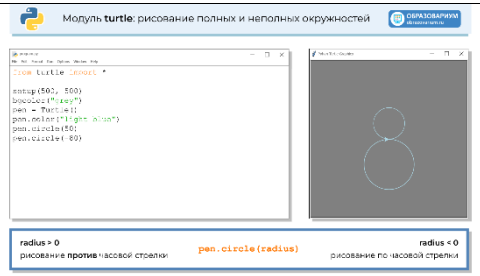
Таблица. Содержание видеоролика

Название блока	Содержание блока и комментарии	Фрагменты из видеоролика	№ сцен

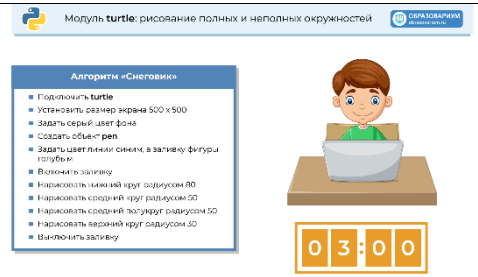
Вводный блок. **Мы узнаем**	*Обозначаем ученикам тему и цели урока.* Модуль turtle: рисование полных и неполных окружностей	 **Сцена 1**	1 2
	Продолжаем изучать возможности модуля **turtle**, связанные с созданием геометрических фигур. Сегодня мы познакомимся с командой, которая помогает создавать окружности.	 **Сцена 2**	
Блок повторения. **Блиц-опрос**	*Повторение материала предыдущего урока; на столе имеются пронумерованные карточки; после каждого вопроса выбираем ту, номер которой, совпадает с правильным ответом.* **Первый вопрос**. Какая команда отсутствует у объекта? pen = Turtle () 1) move 2) left 3) right 4) forward *Ответ № **1**: Команда* move *отсутствует у данного объекта.*	 **Сцена 3**	3 4 5 6 7

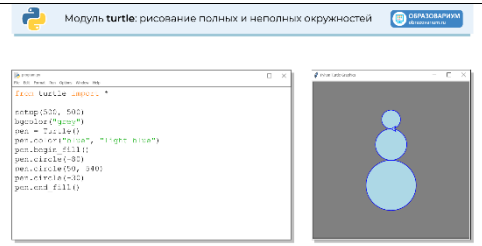
	Второй вопрос. Какая команда не имеет отношения к объекту для рисования? 1) color () 2) bgcolor () 3) begin_fill () 4) end_fill () *Ответ **2**. Команда* bgcolor *устанавливает цвет экрана и не имеет отношения к объекту для рисования*	 Сцена 4	
	Третий вопрос. На сколько градусов необходимо поворачивать, чтобы нарисовать ровный замкнутый шестиугольник? 1) 30 2) 45 3) 60 4) 75 *Ответ **3.** Чтобы узнать угол поворота надо 360 разделить на шесть. Это будет 60*	 Сцена 5	
	Четвертый вопрос. Какой цвет экрана будет если прописать команду bgcolor (1,1,1)? 1) черный 2) белый 3) красный 4) зеленый *Ответ **2.** Цвета находятся в диапазоне от ноля до одного. Данный цвет имеет максимальные параметры, следовательно он самый светлый*	 Сцена 6	

	Пятый вопрос. Какая команда вернет из пользовательской функции данные? 1) def 2) global 3) return 4) input *Ответ **3.** Это сможет сделать команда* return.	 Сцена 7	
Теоретический блок. **Рисование окружностей**	Мы уже умеем рисовать различные правильные многоугольники. Пришла пора познакомиться с окружностями. Определяющим признаком окружности является радиус, который измеряется в пикселях. А чтобы нарисовать круг - необходимо использовать команду **circle**. Радиус – отрезок, который соединяет центр окружности и любую точку на ней. Синтаксис команды для рисования окружности объект.**circle**(радиус).	 Сцена 8	8 9 10
Блок заданий. **Практические задания:** Задание 1.	*После окончания дикторского текста запускается таймер на 4 мин.* **Задание 1. Алгоритм «Круги»** ▪ Подключить **turtle** ▪ Задать размер экрана 500 x 500 ▪ Задать цвет фона ▪ Создать объект **pen** ▪ Задать цвет пера ▪ **pen.circle(50)**	 Сцена 9	11 12

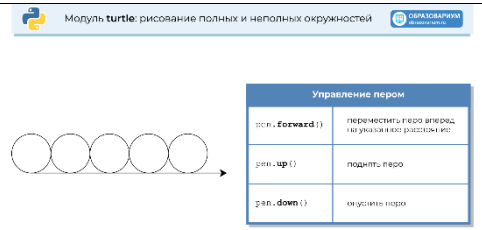
	Разбор задания 1. Код программы может выглядеть так: from turtle import * setup(500, 500) bgcolor("grey") pen = Turtle() pen.color("light blue") pen.circle(50) *pen.circle(radius): radius > 0 рисование против часовой стрелки,* *radius < 0 рисование по часовой стрелки*	 Сцена 10	
Блок заданий. **Практические задания:** Задание 1. Продолжение	Давайте мы превратим наш круг в цифру восемь. Для этого необходимо нарисовать второй круг с отрицательным радиусом. Чтобы лучше смотрелось – сделаем радиус нижнего круга больше, чем у верхнего. **Задание 1. Алгоритм «Круги». Продолжение** Добавьте один круг так, чтобы получилась цифра 8.	 Сцена 11	
Теоретический блок. **Неполная окружность**	Теперь трансформируем нашу цифру восемь в снеговика, состоящего из трех окружностей. И здесь возникает проблема: как перенести перо, чтобы нарисовать верхний круг? Самый простой способ – нарисовать половину среднего круга, чтобы перо оказалось наверху. Если у команды circle указать второй аргумент, то можно нарисовать часть круга. Данный аргумент – количество градусов. **Неполная окружность** pen. cicrcle (50) # целый круг; pen. cicrcle (50, 180) # полкруга; pen. cicrcle (50, 90) # четверть круга	 Сцена 12	12

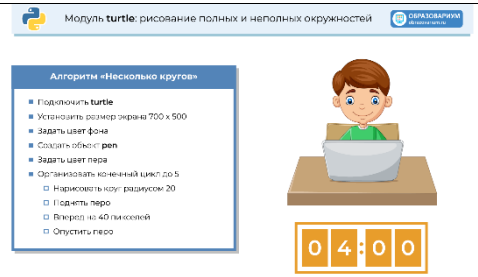
<table>
<tr>
<td rowspan="2">Блок заданий.
Практические задания:
Задание 2.</td>
<td>После окончания дикторского текста запускается таймер на 3 мин.

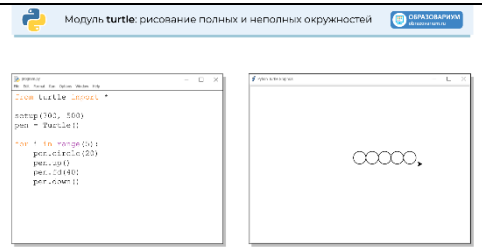
Задание 2. Алгоритм «Снеговик»
▪ Подключить turtle
▪ Задать размеры экрана 500 x 500
▪ Задать серый цвет фона
▪ Создать объект pen
▪ Задать цвет линии синим, а заливку фигуры голубым
▪ Включить заливку
▪ Нарисовать нижний круг радиусом 80
▪ Нарисовать средний круг радиусом 50
▪ Нарисовать средний полукруг радиусом 50
▪ Нарисовать верхний круг радиусом 30
▪ Выключить заливку</td>
<td> 

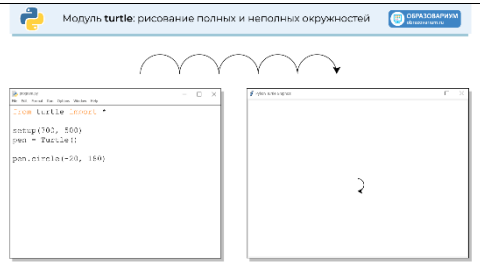
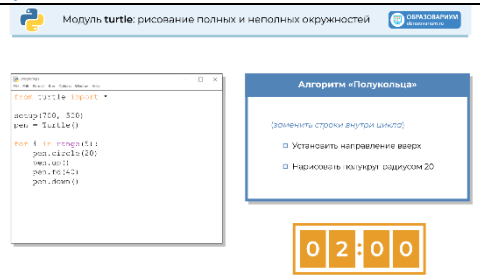
Сцена 13</td>
<td rowspan="2">13
14</td>
</tr>
<tr>
<td>Разбор задания 2. Код программы будет выглядеть так:
from turtle import *
setup(500, 500)
bgcolor("grey")
pen = Turtle()
pen.color("blue", "light blue")
pen.begin_fill()
pen.circle(-80)
pen.circle(50, 540)
pen.circle(-30)
pen.end_fill()</td>
<td> 

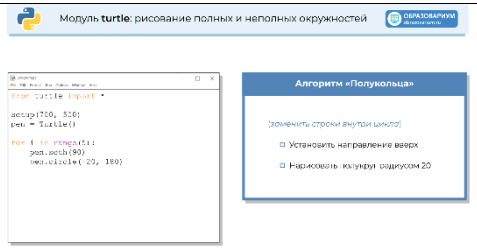
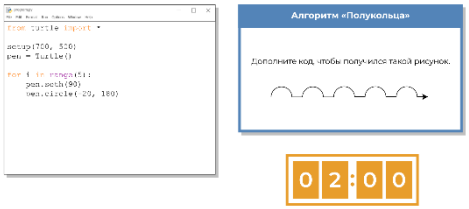
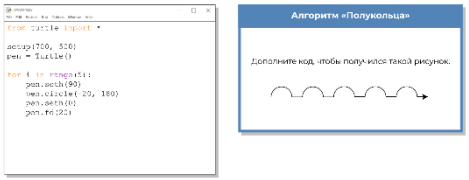
Сцена 14</td>
</tr>
</table>

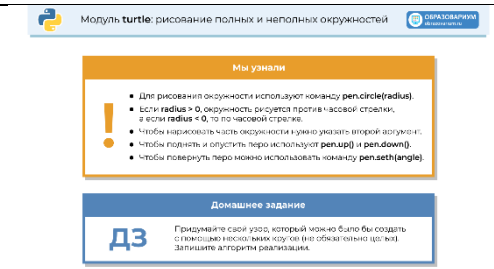
Теоретический блок. **Рисуем окружностями**	Теперь попробуем создать вот такой рисунок. Для этого нам потребуется передвинуть перо в сторону от нарисованного круга. Как это сделать мы знаем – командой forward. Но тогда при переносе будет рисоваться линия. Нельзя ли сделать так, чтобы перо можно было сначала поднять, а потом опустить? Можно, эти команды так и называются: up и down, то есть вверх и вниз. **Управление пером** pen.up() – поднять перо; pen.down() – опустить перо	 Сцена 15	15

Блок заданий.
Практические задания:
Задание 3.
	После окончания дикторского текста запускается таймер на 4 мин.
	
	Задание 3. Алгоритм «Несколько кругов»
	▪ Подключить **turtle**
	▪ Задать размер экрана 800 x 500
	▪ Задать цвет фона
	▪ Создать объект pen
	▪ Задать цвет пера
	▪ Организовать конечный цикл до 5
	▪ Нарисовать круг радиусом 20
	▪ Поднять перо
	  ○ Вперед на 40 пикселей
	  ○ Опустить перо
		
		Сцена 16
			16
			17

	Разбор задания 3. Код программы может измениться так:
	import turtle
	turtle.setup(800, 500)
	pen = turtle.Turtle()
	for i in range(5):
	  pen.circle(20)
	  pen.up()
	  pen.fd(40)
	  pen.down()
		
		Сцена 17	

Теоретический блок. **Рисуем половинами кругов**	Теперь попробуем создать вот такой узор, состоящий из половинок круга. Понятно, что нам понадобится цикл, а самому полукругу указывать в параметре радиус – отрицательное число. Но первая проблема в том, что перо начинает рисовать полукруг не вправо, а вниз! Нетрудно догадаться, что, если указанную команду поставить в цикл – получится просто круг. В чем дело? В направлении пера. Оно начинает свое движение в тот момент, когда смотрит вправо. То есть расположено под углом ноль градусов, как хорошо видно на этом рисунке.	 Сцена 18	18 19
	Для этого имеется команда **seth**, которая разворачивает объект в нужном направлении, независимо от того, куда он был повернут. Как и в случае с радиусом – можно указывать как положительное, так и отрицательное число. Смена направления pen.seth(0) – вправо; pen.seth(90) – вверх; pen.seth(180) – влево; pen.seth(-90) - вниз	 Сцена 19	
Блок заданий. **Практические задания:** Задание 3.	*После окончания дикторского текста запускается таймер на 2 мин.* **Задание 3. Алгоритм «Полукольца».** *(заменить строки внутри цикла)* • Установить направление вверх • Нарисовать полукруг радиусом 20	 Сцена 19	19 20

	Разбор задания 3. Код программы поменяется: from turtle import * setup(800, 500) pen = Turtle() for i in range(5): pen.seth(90) pen.circle(-20, 180)	 Сцена 20	
Блок заданий. **Практические задания:** Задание 3. Продолжение	*После окончания дикторского текста запускается таймер на 3 мин.* **Задание 3. Алгоритм «Полукольца». Продолжение** Дополните код, чтобы получился такой рисунок. 	 Сцена 21	21 22
	Разбор задания 3. *Дописывается к существующему коду* pen.seth(0) pen.fd(20)	 Сцена 22	

Блок завершения занятия. **Рефлексия. Сообщение домашнего задания**	*Завершаем демонстрацией ролика и кратким обобщением материалов занятия.* **Подведем итоги**. Мы узнали: ▪ Для рисования окружности используют команду pen.circle(radius). ▪ Если radius > 0, окружность рисуется против часовой стрелки, а если radius < 0, то по часовой стрелке. ▪ Чтобы нарисовать часть окружности нужно указать второй аргумент. ▪ Чтобы поднять и опустить перо используют pen.up() и pen.down() ▪ Чтобы повернуть перо можно использовать команду pen.seth(angle) *Преподаватель дает ученикам домашнее задание к следующему занятию (Приложение 2).*	 Сцена 23	23

Приложение 2

Домашнее задание

Какие другие узоры можно было бы создать с помощью нескольких окружностей (не обязательно полных)? Придумайте свою идею и запишите алгоритм ее реализации.

Задание можно выполнить на компьютере и представить результат и код в виде файла или снимка экрана, или распечатки.

Практика

Проект «Светофор»

С помощью модуля turtle создайте изображение светофора.

Проект «Радуга»
С помощью модуля turtle создайте изображение радуги.
Код может быть основан на принципе рисования полукруга вперед и назад (для возвращения в исходную точку).
Нарисуйте 7 линий подходящих цветов, увеличив толщину линии.
Для увеличения толщины линии измените размер пера с помощью метода **pensize**().

program.py

File Edit Format Run Options Window Help

```
from turtle import *

pen = Turtle ()
pen.pensize(5)  # так задается размер пера
```

Приложение 3

Краткие организационно-методические рекомендации по организации работы на занятии
«Модуль turtle: рисование полных и неполных окружностей».

Перед началом занятия необходимо повторить понятия, с которыми нам предстоит работать (градусы, пиксели), а также новые команды (установка размера, движение, повороты, заливка фигур и изменение цвета объекта для рисования).

Перед просмотром блока повторения из ролика необходимо раздать дидактический материал для выполнения заданий (по 4 пронумерованных карточки).

Во время голосований карточками можно останавливать ролик и вести учет правильных ответов. По окончание блока – отметить тех, у кого наилучший результат. Далее карточки необходимо собрать.

На уроке будет создано несколько проектов с одинаковыми установочными данными: размер экрана, цвет пера, цвет фона. Поэтому можно переносить данный код из одного проекта в другой ради экономии времени.

Особое внимание уделите тому, что радиус круга и направление пера могут быть не только положительным, но и отрицательным числом. Иногда это даже удобнее.
Например, минус 90 понять проще чем 270 Во время объяснения заданий № 2 и 3 можно поставить ролик на паузу и проговорить с ребятами алгоритм создания рисунка от первого лица.
Основная путаница чаще всего связана с представлением того, что направление ноль градусов – это вправо. Тем более что, например, в программе Скретч это как раз - вверх. Обратите на это особое внимание.