



### Методические рекомендации по теме

#### «Модуль turtle: обработка событий «мыши» для управления анимацией»

##### Цель:

- дать представление об управлении исполнителем при помощи мыши в модуле turtle языка Python.

##### Задачи:

- знакомство с программированием управления исполнителем при помощи мыши;
- программирование событий при нажатии различных клавиш мыши;
- 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 получают 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 получают представления об программировании мыши для управления исполнителем модуля turtle в языке Python.

*Метапредметные:* обучающиеся получают возможность владения общепредметными понятиями «событие», «исполнитель», «векторная графика», «пиксель», «координаты»; 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:

- Что такое событие в программировании?
- Какие события могут быть при управлении мышью?
- Как может себя вести исполнитель при управлении мышью?

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

- При управлении мышью событием может быть нажатие на кнопку или колесико, перемещение мыши
- Исполнитель может выполнить программу, переместиться в указатель мыши, двигаться к указателю мыши и т.д.

*\*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 большей 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
- Управление «мышью»
- поворот влево и вправо с помощью «мышы»

- Угол поворота
- Координаты «мыши» и объекта
- Условия касания

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 9 – 21 (кроме сцен с теорией)*

Практические задания разделены **теоретическими вставками**, необходимыми для работы над проектами урока:

- Команды обработчика событий
- Кнопки мыши
- Действия обработчика
- Координаты «мыши»

*\*см. сцены 8, 11, 14, 17*

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 22.*

## Приложение 1

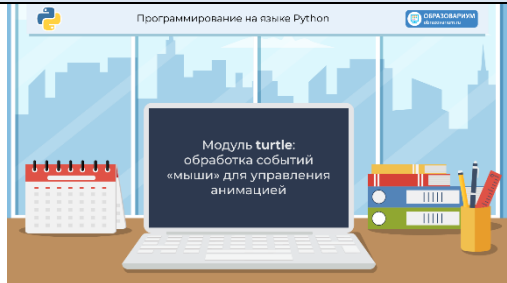
### Сценарный план видеоролика


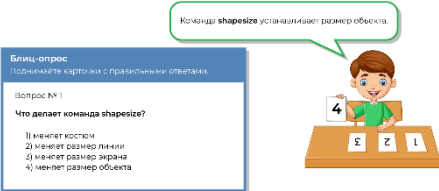
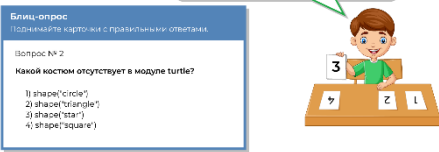
В таблице «Содержание видеоролика» представлены:

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

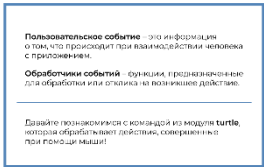




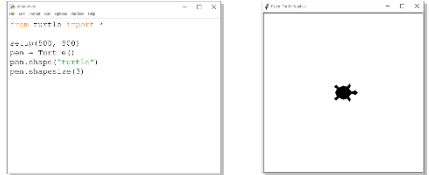
*Учитель при подготовке к уроку может ознакомиться с содержанием видеоролика в текстовом формате, при необходимости распечатать фрагменты текста или примеры заданий и задач для использования в работе с учениками. Распечатанные тексты и задания из таблицы также можно применять в качестве раздаточного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

Таблица. Содержание видеоролика

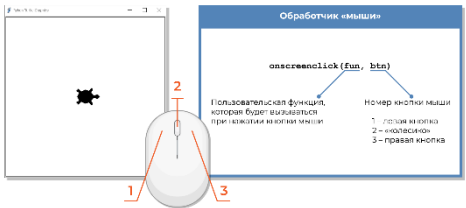
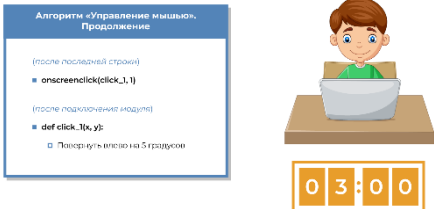
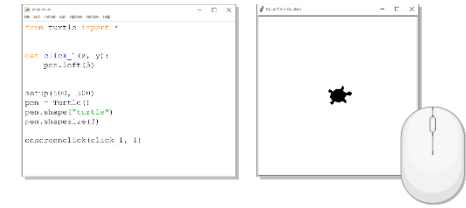
Название блока	Содержание блока и комментарии	Фрагменты из видеоролика	№ сцен
Вводный блок. Мы узнаем	Обозначаем ученикам тему и цели урока.  Модуль turtle: обработка событий «мыши» для управления анимацией	 Сцена 1	1 2

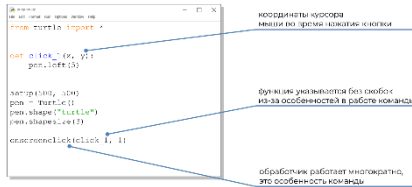
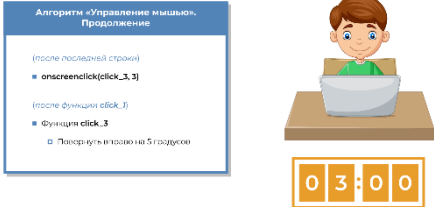
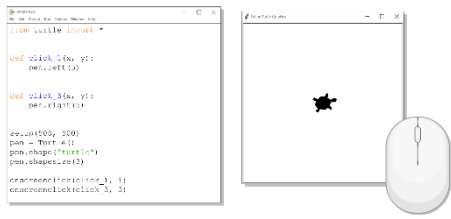
	<p>Для создания полноценных интерактивных приложений нам необходимо научиться взаимодействовать с программой с точки зрения управления событиями. Для этого у нас имеются клавиатура и «мышь». Возможности клавиатуры мы рассмотрим на следующем занятии, а сегодня узнаем – как можно заставить программу реагировать на нажатие кнопок «мыши».</p>	 <p>Сцена 2</p>	
<p>Блок повторения.</p> <p><b>Блиц-опрос</b></p>	<p><i>Повторение материала предыдущего урока; на столе имеются пронумерованные карточки; после каждого вопроса выбираем ту, номер которой, совпадает с правильным ответом.</i></p> <p><b>Первый вопрос.</b> Что делает команда shapesize?</p> <ol style="list-style-type: none"> <li>1) Меняет костюм</li> <li>2) Меняет размер линии</li> <li>3) Меняет размер экрана</li> <li>4) Меняет размер объекта</li> </ol> <p><i>Ответ № 4: Эта команда устанавливает новый размер объекта</i></p>	 <p>Сцена 3</p>	3 4 5 6 7
	<p><b>Второй вопрос.</b> Какой костюм отсутствует в модуле turtle?</p> <ol style="list-style-type: none"> <li>1) shape("circle")</li> <li>2) shape("triangle")</li> <li>3) shape("star")</li> <li>4) shape("square")</li> </ol> <p><i>Ответ 3. Команда шейп меняет костюм объекту. Имеет формы: треугольник, стрелка, квадрат, круг, и черепашка. Звезда - отсутствует</i></p>	 <p>Сцена 4</p>	

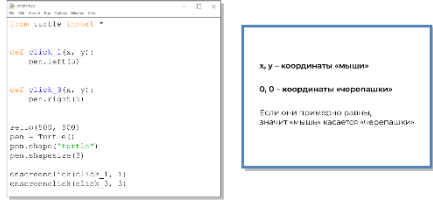
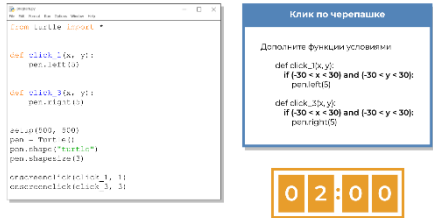
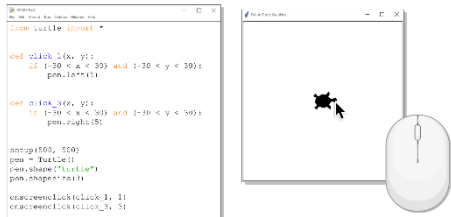
	<p><b>Третий вопрос.</b> В какой угол переместится объект из центра после данной команды: goto(300, -300)?</p> <ol style="list-style-type: none"> <li>1) правый верхний</li> <li>2) правый нижний</li> <li>3) левый верхний</li> <li>4) левый нижний</li> </ol> <p><i>Ответ 2. По оси X объект сместится вправо, а по оси Y вниз, т. е. правый нижний угол.</i></p>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p> <p>По оси X объект сместится вправо, а по оси Y вниз, т. е. правый нижний угол – правый нижний угол!</p> <p><b>Блиц-опрос</b> Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 3 В какой угол переместится объект из центра после данной команды: goto(300, -300)?</p> <ol style="list-style-type: none"> <li>1) правый верхний</li> <li>2) правый нижний</li> <li>3) левый верхний</li> <li>4) левый нижний</li> </ol> <p>Сцена 5</p>	
	<p><b>Четвертый вопрос.</b> Какая часть круга будет нарисована командой circle(18, 180)?</p> <ol style="list-style-type: none"> <li>1) 1/1</li> <li>2) 2/1</li> <li>3) 1/2</li> <li>4) 1/10</li> </ol> <p><i>Ответ 3. В команде circle() за то, какая часть круга будет нарисована отвечает второй аргумент. Полный круг – это 360. Значит, 180 – это 1/2 круга.</i></p>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p> <p>В команде circle() за то, какая часть круга будет нарисована, отвечает второй аргумент. Полный круг – это 360. Значит, 180 – это 1/2 круга.</p> <p><b>Блиц-опрос</b> Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 4 Какая часть круга будет нарисована командой circle(18, 180)?</p> <ol style="list-style-type: none"> <li>1) 1/1</li> <li>2) 2/1</li> <li>3) 1/2</li> <li>4) 1/10</li> </ol> <p>Сцена 6</p>	
	<p><b>Пятый вопрос.</b> Какая команда не имеет отношения к объекту для рисования?</p> <ol style="list-style-type: none"> <li>1) end_fill()</li> <li>2) color()</li> <li>3) begin_fill()</li> <li>4) bgcolor()</li> </ol> <p><i>Ответ 4. Команда bgcolor() задает цвет экрана и не имеет отношения к объекту для рисования.</i></p>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p> <p>Команда bgcolor() задает цвет экрана и не имеет отношения к объекту для рисования.</p> <p><b>Блиц-опрос</b> Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 5 Какая команда не имеет отношения к объекту для рисования?</p> <ol style="list-style-type: none"> <li>1) begin_fill()</li> <li>2) end_fill()</li> <li>3) color()</li> <li>4) bgcolor()</li> </ol> <p>Сцена 7</p>	

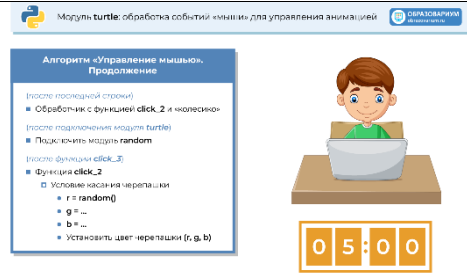
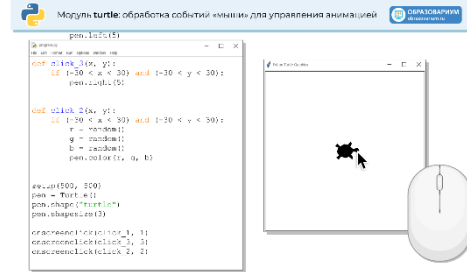
<p>Теоретический блок.</p> <p><b>Обработчик событий</b></p>	<p>Любое действие человека, связанное с кликом мыши – например, запуск или закрытие программы – называется пользовательское событие.</p> <p>Чтобы понять какое именно событие произошло и что нужно сделать, используются специальные команды, которые называются обработчик событий.</p> <p>Давай познакомимся с командой из модуля turtle, которая обрабатывает действия, совершенные при помощи мыши!</p>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>   <p>Сцена 8</p>	8
<p>Блок заданий.</p> <p><b>Практические задания:</b></p> <p>Задание 1.</p>	<p><i>После окончания дикторского текста запускается таймер на 4 мин.</i></p> <p><b>Задание 1. Алгоритм «Управление мышью»</b></p> <ul style="list-style-type: none"> <li>■ Подключить turtle</li> <li>■ Установить размер экран 500 x 500</li> <li>■ Создать объект pen</li> <li>■ Установить костюм «черепашки»</li> <li>■ Установить размер, равный 3</li> </ul>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>    <p>Сцена 9</p>	9 10
	<p><b>Разбор задания 1.</b> Код программы может выглядеть так:</p> <pre>from turtle import * setup(500, 500) pen = Turtle() pen.shape("turtle") pen.shapesize(3)</pre>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 10</p>	

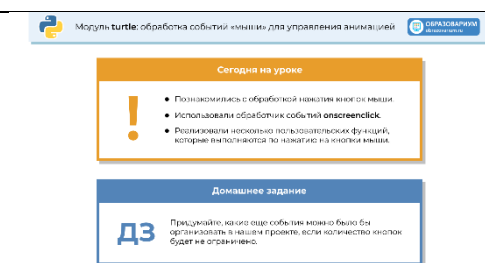


<p>Теоретический блок.</p> <p><b>Кнопки мыши</b></p>	<p>Давайте научим ее поворачиваться вправо или влево, в зависимости от того, какую кнопку мыши мы нажмем.</p> <p>Для этого надо прописать команду <code>onscreenclick(fun, btn)</code></p> <p>Также, вторым параметром, надо поставить число от одного до трех. Один – это активация левой кнопки, два – колесика, три – правая кнопка.</p>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 11</p>	11
<p>Блок заданий.</p> <p><b>Практические задания:</b></p> <p>Задание 1.</p> <p>Продолжение</p>	<p><i>После окончания дикторского текста запускается таймер на 3 мин.</i></p> <p><b>Задание 1. Алгоритм «Управление мышью». Продолжение</b></p> <p><i>после последней строки</i></p> <ul style="list-style-type: none"> <li>■ <code>onscreenclick(click_1, 1)</code></li> </ul> <p><i>после подключения модуля</i></p> <ul style="list-style-type: none"> <li>■ <code>def click_1(x, y):</code> <ul style="list-style-type: none"> <li>○ повернуть влево на 5 градусов</li> </ul> </li> </ul>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 12</p>	12 13
	<p><b>Разбор задания 1.</b> Код программы может выглядеть так:</p> <p><i>Дописывается к существующему коду</i></p> <pre> from turtle import * def click_1(x, y):     pen.left(5) setup(500, 500) pen = Turtle() pen.shape("turtle") pen.shapesize(3) onscreenclick(click_1, 1) </pre>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 13</p>	

<p>Теоретический блок.</p> <p><b>Обработчик</b></p>	<p>Вероятно, возникли вопросы</p> <ul style="list-style-type: none"> <li>- что это за икс и игрек в нашей пользовательской функции?</li> </ul> <p>Дело в том, что обработчик в обязательном порядке передает координаты мыши. Нам они сейчас не нужны, но это его не волнует. Если мы создадим функцию без этих аргументов – получим ошибку</p> <ul style="list-style-type: none"> <li>- а почему функция, внутри обработчика, вызывается без скобок?</li> </ul> <p>Это особенности некоторых служебных команд. Можно сказать – исключение из правил</p> <ul style="list-style-type: none"> <li>- третий: а почему обработчик работает многократно, хотя никакого цикла нет?</li> </ul> <p>Ответ такой же, как и на предыдущий вопрос</p>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 14</p>	<p>14</p>
<p>Блок заданий.</p> <p><b>Практические задания:</b></p> <p>Задание 1.</p> <p>Продолжение</p>	<p>После окончания дикторского текста запускается таймер на 3 мин.</p> <p><b>Задание 1. «Управление мышью». Продолжение</b></p> <p>после последней строки</p> <ul style="list-style-type: none"> <li>onscreenclick(click_3, 3)</li> </ul> <p>после функции click_1</p> <ul style="list-style-type: none"> <li>Функция click_3 <ul style="list-style-type: none"> <li>Повернуть вправо на 5 градусов</li> </ul> </li> </ul>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 15</p>	<p>15 16</p>
	<p><b>Разбор задания 1.</b> Код программы будет выглядеть так:</p> <pre> from turtle import * def click_1(x, y):     pen.left(5) def click_3(x, y):     pen.right(5) setup(500, 500) pen = Turtle() pen.shape("turtle") pen.shapesize(3) onscreenclick(click_1, 1) onscreenclick(click_3, 3) </pre>	<p>Модуль turtle: обработка событий «мышь» для управления анимацией</p>  <p>Сцена 16</p>	

<p>Теоретический блок.</p> <p><b>Координаты</b></p>	<p>А как же сделать так, чтобы черепашка вращалась лишь тогда, когда мы кликнем именно по ней?</p> <p>Вот для этого нам и понадобятся <math>x, y</math> из наших функций. Ведь это координаты мыши, и если они будут близки к координатам черепашки – значит есть касание</p> <p><math>x, y</math> – координаты «мыши»  <math>0, 0</math> – координаты «черепашки»</p> <p>Если они примерно равны – значит «мышь» касается «черепашки»</p>	 <p>Сцена 17</p>	<p>17</p>
<p>Блок заданий.</p> <p><b>Практические задания:</b></p> <p>Задание 2.</p>	<p><i>После окончания дикторского текста запускается таймер на 2 мин.</i></p> <p><b>Задание 2. Клик по черепашке</b></p> <p><i>Дополните функции условиями</i></p> <pre>def click_1(x,y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         pen.left (5) def click_3(x,y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         pen.right (5)</pre>	 <p>Сцена 18</p>	<p>18 19</p>
	<p><b>Разбор задания 2.</b> Код программы может выглядеть так:</p> <p><i>Дописывается к уже существующему коду</i></p> <pre>from turtle import * def click_1(x, y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         pen.left(5) def click_3(x, y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         pen.right(5) setup(500, 500) pen = Turtle() pen.shape("turtle") pen.shapesize(3)</pre>	 <p>Сцена 19</p>	

	<pre>onscreenclick(click_1, 1) onscreenclick(click_3, 3)for i in range(10):     move()</pre>		
<p>Блок заданий. <b>Практические задания:</b> Задание 1.</p>	<p><i>После окончания дикторского текста запускается таймер на 5 мин.</i></p> <p><b>Задание 1. Алгоритм «Управление мышью». Продолжение</b> (после последней строки)</p> <ul style="list-style-type: none"> <li>• Обработчик с функцией click_2 и «колесико» после подключения модуля turtle</li> <li>• Подключить модуль random после функции click_3</li> <li>• Функция click_2             <ul style="list-style-type: none"> <li>○ Условие касания черепашки                     <ul style="list-style-type: none"> <li>▪ r = random()</li> <li>▪ g = ....</li> <li>▪ b = ...</li> <li>▪ Установить цвет черепашки (r, g, b)</li> </ul> </li> </ul> </li> </ul>	 <p>Сцена 20</p>	20 21
	<p><b>Разбор задания 1.</b> Код программы может выглядеть так: <i>Дописывается к уже существующему коду</i></p> <pre>from turtle import * from random import * def click_1(x, y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         pen.left(5) def click_3(x, y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         pen.right(5) def click_2(x, y):     if (-30 &lt; x &lt; 30) and (-30 &lt; y &lt; 30):         r = random()         g = random()         b = random()         pen.color(r, g, b)</pre>	 <p>Сцена 21</p>	

	<pre> setup(500, 500) pen = Turtle() pen.shape("turtle") pen.shapesize(3) onscreenclick(click_1, 1) onscreenclick(click_3, 3) onscreenclick(click_2, 2) </pre>		
<p>Блок завершения занятия.</p> <p><b>Рефлексия.</b></p> <p><b>Сообщение домашнего задания</b></p>	<p><i>Завершаем демонстрацией ролика и кратким обобщением материалов занятия.</i></p> <p><b>Подведем итоги.</b></p> <p>Мы узнали:</p> <ul style="list-style-type: none"> <li>▪ Приемы с обработкой нажатия кнопок мыши.</li> <li>▪ Как использовать обработчик событий onscreenclick</li> <li>▪ Как реализовать несколько пользовательских функций, которые выполняются по нажатию на кнопки мыши</li> </ul> <p><i>Преподаватель дает ученикам домашнее задание к следующему занятию (Приложение 2).</i></p>	 <p>Сцена 22</p>	22

## Приложение 2

### Домашнее задание

Какие еще события можно было бы организовать в нашем проекте, если количество кнопок мыши будет не ограничено?

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

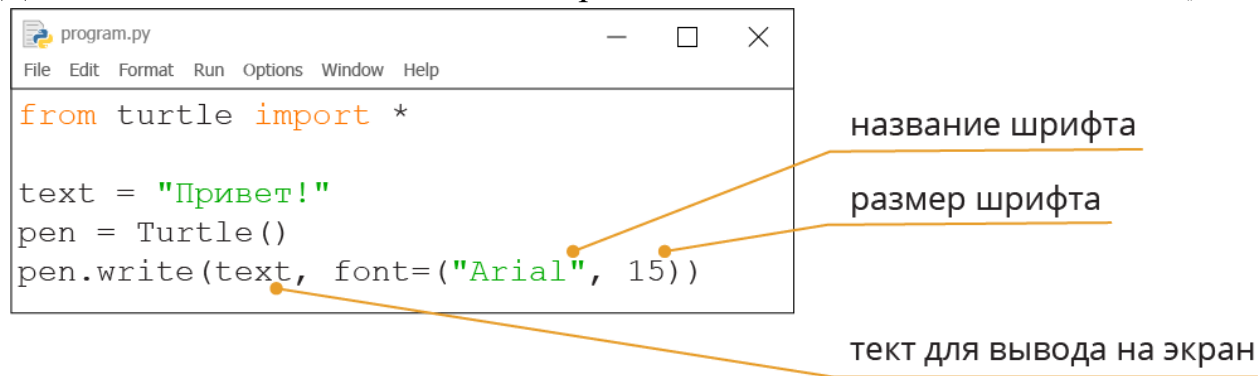
### Практика

Проект «Точки на плоскости»

С помощью модуля turtle создайте программу для автоматического обозначения точек и их координат на плоскости. Создайте функцию, которая будет перемещать объект в координаты курсора мыши, оставлять отпечаток объекта и выводить на экран координаты отпечатка. Создайте обработчик, который будет вызывать эту функцию по нажатию на левую кнопку мыши.

Чтобы нарисовать «точки на плоскости» поменяйте форму пера на круг.

Для того, чтобы вывести текст на экран, можно использовать метод **write()**.

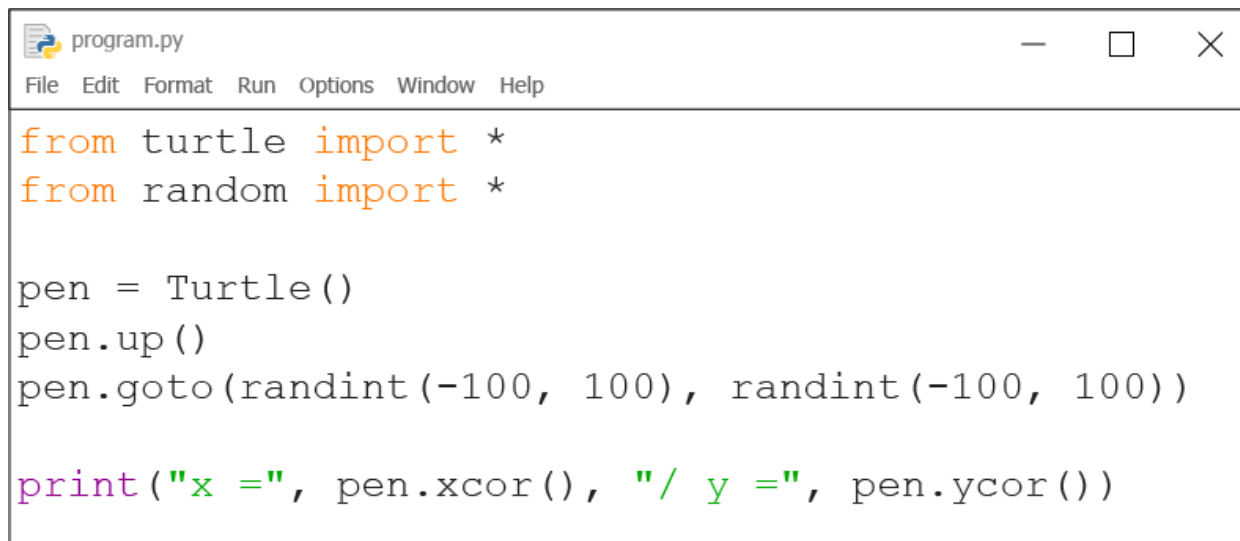


Такой код напечатает Привет! рядом с пером.

### Проект «Три кнопки»

С помощью модуля turtle создайте три функции и обработчики для вызова этих функций по нажатию на кнопки мыши. Первая функция должна перемещать объект в точку координат курсора мыши. Вторая функция оставляет отпечаток, а третья произвольным образом меняет костюм. Для второй и третьей функции сделайте проверку, что кликнули по объекту или рядом с ним.

Чтобы определить координаты объекта используйте методы **xcor()**, **ycor()**.



```
from turtle import *
from random import *

pen = Turtle()
pen.up()
pen.goto(randint(-100, 100), randint(-100, 100))

print("x =", pen.xcor(), "/ y =", pen.ycor())
```

Такой код выведет координаты пера на экране

### Краткие организационно-методические рекомендации по организации работы на занятии

«Модуль turtle: обработка событий «мыши» для управления анимацией».

**Перед началом** занятия необходимо повторить информацию про декартовы координаты точек на плоскости, как координаты зависят от размера нашего экрана и где могут находиться точки (относительно центра) с некими произвольно заданными координатами.

**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 окончании блока – отметить тех, у кого наилучший результат. Далее карточки необходимо собрать.

**Создаваемый проект** позволит понять принцип работы обработчика событий «мыши» – команды `onscreenclick`.

Проект состоит из нескольких частей, каждую из которых можно тестировать после создания.

**Особенное внимание** необходимо уделить функциям, создаваемым обработчиком. Точнее их обязательным аргументам. Это две переменные, которые получают в обязательном порядке координаты «мыши» в момент клика. Для удобства – они названы X и Y. Поскольку в модуле отсутствует команда, определяющая касание «мыши» с объектом, то для реализации данного кода используется принцип сравнения их координат. При размере костюма равным трем, объект, независимо от костюма, «занимает» на экране квадрат, со стороной 60 пикселей (по 30 в каждую сторону от центра). Поэтому под «касанием» мы понимаем нахождение «мыши» в этом участке экрана.

**Если вы используете** «мышь» более чем с тремя кнопками, то у вас появляются новые возможности. Однако понять, какая кнопка имеет номер начиная с 4-х можно только опытным путем. Как пример – однажды удалось активировать 6 из 7ми кнопок геймерской «мыши».