



Методические рекомендации по теме «Циклы с ограниченным количеством повторений»

Цель:

- понимание организации цикла с заданным числом повторений и решение задач с таким циклом на языке Python.

Задачи:

- расширение кругозора обучающихся в области информатики и программирования;
- знакомство с основами работы с циклами с заданным числом повторений на языке Python;
- решение программных прикладных задач с циклами с заданным числом повторений на языке Python;
- ранняя профориентация школьников, профессиональная деятельность программиста;
- развитие интеллектуальных способностей, логического и критического мышления.

Планируемые результаты

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

Предметные: обучающиеся получают представления: о циклах с заданным числом повторений в языке Python; об основных операциях с конечными циклами на языке программирования «Python»; о прикладном использовании операций с циклами с ограниченным числом повторений в программных проектах; о возможностях и особенностях применения циклов с ограниченным числом повторений в практике работы программиста.

Метапредметные: обучающиеся получают возможность владения обще предметными понятием «цикл», «повторение», «конечный цикл»; владение информационно-логическими умениями; владение умениями самостоятельно планировать пути достижения целей; умениями принятия решений и осуществления осознанного выбора; повысят уровень ИКТ-компетентности.

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

Приложение 4: Алгоритм подсчета частоты появления символа в строке (дополнительно).

Ход проведения урока

1. Организационный момент.

Мотивация на учебную деятельность.

Приветствие учащихся, сообщение темы и целей занятия (мы узнаем, про возможности работы с циклами с заданным числом повторений (итераций); научимся решать прикладные задания с помощью «конечного цикла»).

Проблемная дискуссия по вопросам:

- Что такое цикл с ограниченным числом повторений. Приведите примеры из жизни?
- Для чего может пригодиться цикл с ограниченным числом повторений при программировании в Python?
- Какой синтаксис вы бы использовали при работе с такими циклами? Попробуйте придумать свою запись?

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

- конечные циклы отличаются от циклов с условием;
- в конечных циклах нам заранее известно сколько раз должно повториться тело цикла.

Преподаватель называет ученикам тему и цели урока.

2. Вводный блок.

Тема.

Преподаватель при необходимости останавливая трансляцию, комментируя дополнительно тему занятия.

**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 **в сцене 3**; 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. Теоретический блок.

Конечный цикл.

Продолжение демонстрации ролика с дальнейшим обсуждением вопросов:

- Какие команды используются при работе с конечными циклами? Что они значат?
- Что такое «счетчик» и как он связан с «конечным циклом»?
- Какой буквой традиционно обозначается счетчик?

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.

**см. сцены 8 – 13*

5. Блок заданий.

Проекты: «5 приветов», «От и до», «Выбор цикла».

К началу демонстрации блока заданий ученики должны занять рабочие места и запустить Python (терминал IDLE) на своих компьютерах.

Блок включает **3 практических задания** для учеников с последующим разбором. Задания представляют собой 3 небольших программных проекта с использованием «конечных циклов».

После выполнения заданий ученики получают три работающих программных продукта – программа для вывода на экран заданного числа повторений текста, программа выводющая последовательность чисел, программа, которая строит восходящие и нисходящие последовательности чисел.

Блок включает в себя теоретические вставки: «Восходящий и нисходящий цикл».

На сцене разбора задания преподаватель ставит ролик на паузу и вместе с учениками проводит разбор задания.

**см. сцены 14 – 24*

6. Рефлексия. Сообщение домашнего задания.

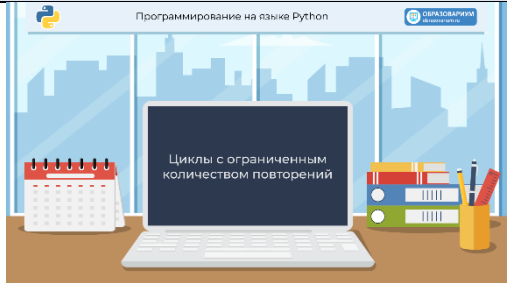
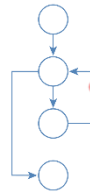
Завершаем демонстрацией ролика и кратким обобщением материалов занятия. Преподаватель возвращается к зафиксированному в ходе дискуссии в начале урока предположениям учеников и обсуждает насколько их предположения были правильными, делаются выводы.










Преподаватель дает ученикам домашнее задание к следующему занятию (*Приложение 2*). **см. сцена 25*

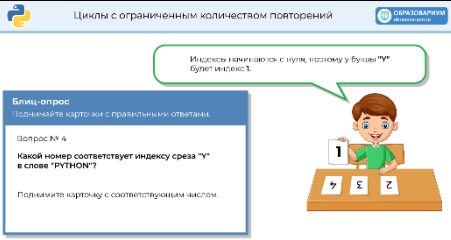
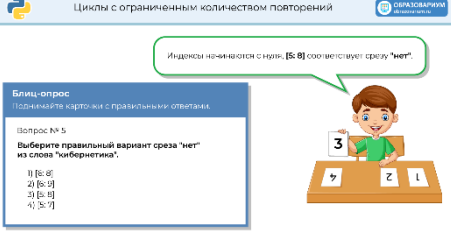
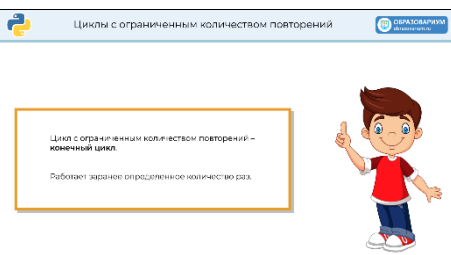
Сценарный план видеоролика


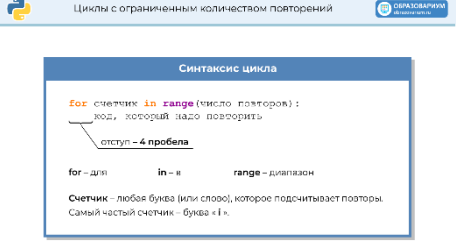
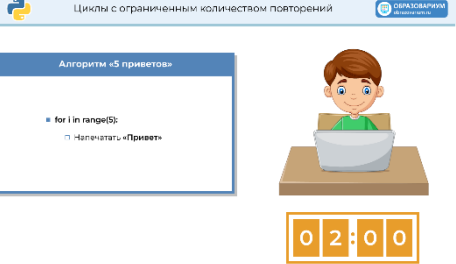
В таблице «Содержание видеоролика» представлен краткий текст из видеоролика, примеры заданий и задач, которые будут демонстрироваться на экране. Учитель при подготовке к уроку может ознакомиться с содержанием видеоролика в текстовом формате, при необходимости распечатать фрагменты текста или примеры 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.

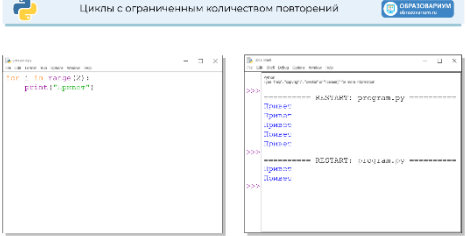
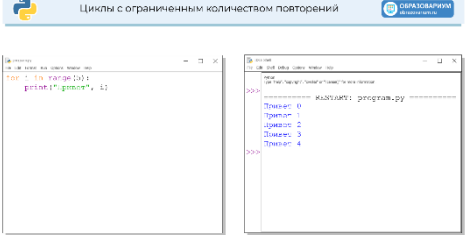
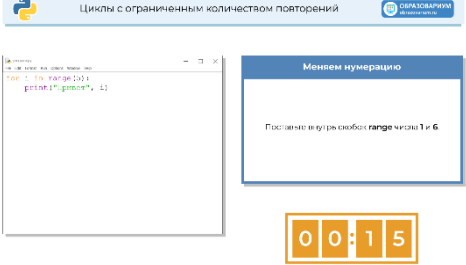
Таблица. Содержание видеоролика

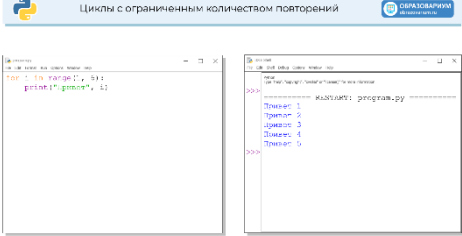
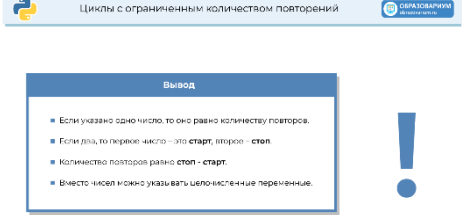
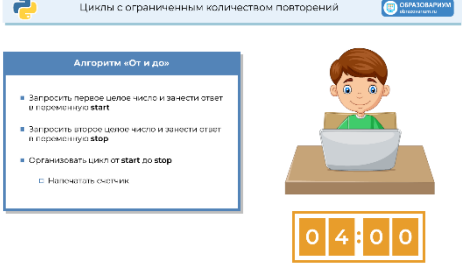
Название блока	Содержание блока и комментарии	Фрагменты из видеоролика	№ сцен
Вводный блок. Мы узнаем	Обозначаем ученикам тему и цели урока. Циклы с ограниченным количеством повторений.	 Сцена 1	1 2
	Цикл – это многократное повторение одинаковых действий. Нам известны циклические алгоритмы, в которых присутствовал цикл с условием. Познакомимся еще с одной разновидностью – конечный цикл.	 Сцена 2	

<p>Блок повторения.</p> <p>Блиц-опрос</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>Первый вопрос. Какая функция не даст нам результат в виде целого числа?</p> <ol style="list-style-type: none"> 1) float 2) int 3) ceil 4) floor <p><i>Ответ 1. float возвращает дробное число.</i></p>	<div> <div>  Циклы с ограниченным количеством повторений <div>  Блиц-опрос </div> </div> <div> <p>Функция float возвращает дробное число.</p> </div> <div> <div> <p>Блиц-опрос</p> <p>Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 1</p> <p>Какая функция не даст результат в виде целого числа?</p> <p>1) float</p> <p>2) int</p> <p>3) ceil</p> <p>4) floor</p> </div> <div>  </div> </div> </div> <p>Сцена 3</p>	3 4 5 6 7
	<p>Второй вопрос. Число 5.8 округлили до 5. Какая функция не могла это сделать?</p> <ol style="list-style-type: none"> 1) int 2) ceil 3) floor 4) Все могут <p><i>Ответ 2. ceil округляет в большую сторону. ceil(5.8) == 6</i></p>	<div> <div>  Циклы с ограниченным количеством повторений <div>  Блиц-опрос </div> </div> <div> <p>Функция ceil округляет в большую сторону. ceil(5.8) == 6</p> </div> <div> <div> <p>Блиц-опрос</p> <p>Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 2</p> <p>Число 5.8 округлили до 5. Какая функция не могла это сделать?</p> <p>1) int</p> <p>2) ceil</p> <p>3) floor</p> <p>4) все могут</p> </div> <div>  </div> </div> </div> <p>Сцена 4</p>	
	<p>Третий вопрос. Какая функция подсчитает количество символов в тексте?</p> <ol style="list-style-type: none"> 1) input 2) len 3) round 4) print <p><i>Ответ 2. len возвращает количество символов в тексте..</i></p>	<div> <div>  Циклы с ограниченным количеством повторений <div>  Блиц-опрос </div> </div> <div> <p>Функция len возвращает количество символов в тексте.</p> </div> <div> <div> <p>Блиц-опрос</p> <p>Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 3</p> <p>Какая функция подсчитает количество символов в тексте?</p> <p>1) input</p> <p>2) len</p> <p>3) round</p> <p>4) print</p> </div> <div>  </div> </div> </div> <p>Сцена 5</p>	

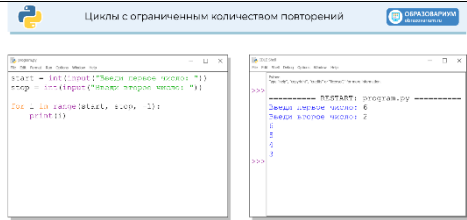

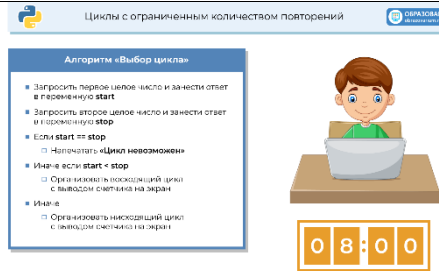
	<p>Четвертый вопрос. Поднимите номер карточки, соответствующий индексу среза "Y" в слове "PYTHON".</p> <p><i>Ответ 1. Индексы начинаются с нуля, поэтому у буквы "Y" будет индекс 1.</i></p>	 <p>Циклы с ограниченным количеством повторений</p> <p>Индексы начинаются с нуля, поэтому у буквы "Y" будет индекс 1.</p> <p>Блиц-опрос Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 4 Какой номер соответствует индексу среза "Y" в слове "PYTHON"?</p> <p>Поднимите карточку с соответствующим числом.</p> <p>Сцена 6</p>	
	<p>Пятый вопрос. Выберите правильный вариант среза "нет" из слова "кибернетика"</p> <p>1) [6:8] 2) [6:9] 3) [5:8] 4) [5:7]</p> <p><i>Ответ 3. Индексы начинаются с нуля, [5:8] соответствует срезу "нет".</i></p>	 <p>Циклы с ограниченным количеством повторений</p> <p>Индексы начинаются с нуля, [5:8] соответствует срезу "нет".</p> <p>Блиц-опрос Поднимайте карточки с правильными ответами.</p> <p>Вопрос № 5 Выберите правильный вариант среза "нет" из слова "кибернетика".</p> <p>1) [6:8] 2) [6:9] 3) [5:8] 4) [5:7]</p> <p>Сцена 7</p>	
<p>Теоретический блок.</p> <p>Конечный цикл.</p>	<p><i>При необходимости преподаватель может поставить ролик на паузу и дать дополнительные пояснения по материалу</i></p> <p>Цикл, с ограниченным количеством повторений еще называют конечным циклом. Для начала разберемся, чем конечный цикл отличается от цикла с условием.</p>	 <p>Циклы с ограниченным количеством повторений – конечный цикл.</p> <p>Работает заданное определенное количество раз.</p> <p>Сцена 8</p>	<p>8 9 10 11 12 13</p>

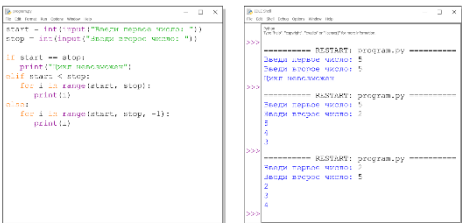
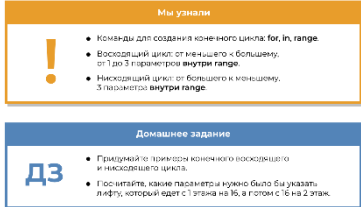
	<p>Например, если нужно подняться по лестнице, количество ступеней в которой неизвестно. В этом мы случае мы получим цикл с условием: идти пока лестница не закончится.</p> <p>В примере, когда известно, что в лестнице 15 ступеней будет использован конечный цикл: сделать 15 шагов</p>	 <p>Сцена 9</p>	
	<p>Для организации такого цикла используется сочетание нескольких команд: for счетчик in range(число повторов): код, который надо повторить</p> <p>Строки, которые надо повторить начинаются с отступа в четыре пробела. Счетчик – это любая буква или слово (но не служебное), которое подсчитывает повторы, в качестве счетчика используют букву «i».</p>	 <p>Сцена 10</p>	
<p>Блок заданий. Практические задания: Задание 1 Задание 2</p>	<p><i>После окончания дикторского текста запускается таймер на 2 мин.</i></p> <p>Задание 1. Алгоритм «5 приветов» for i in range(5): напечатать «Привет»</p>	 <p>Сцена 11</p>	<p>11 12 13 14 15 16 17 18 19</p>

	<p>Разбор задания 1. Наш код будет выглядеть так.</p> <pre>for i in range(5): print("Привет")</pre> <p>Если заменить в скобках 5 на 2, снова запустить программу, то будет «2 привет».</p> <p><i>После окончания времени провести разбор решения, анализируя код.</i></p>	 <p>Сцена 12</p>	<p>20 21</p>
	<p>Разбор задания 1 (продолжение). Если нужно вывести значения счетчика, то код может выглядеть так.</p> <pre>for i in range(5): print("Привет", i)</pre> <p>Где-то мы это видели?! Индекс в срезах!</p> <p><i>После окончания времени провести разбор решения, анализируя код.</i></p>	 <p>Сцена 13</p>	
	<p><i>После окончания дикторского текста запускается таймер на 15 сек.</i></p> <p>Задание 1. Алгоритм «5 приветов» (продолжение)</p> <p>Возвращается стандартное оформление блока заданий: Меняем нумерацию</p> <p>Поставьте внутри скобок range числа 1 и 6.</p>	 <p>Сцена 14</p>	

	<p>Разбор задания 1 (окончание). Если нужно вывести значения счетчика, то код может выглядеть так. Теперь числа будут от одного до пяти включительно.</p> <pre>for i in range(1, 6): print("Привет", i)</pre> <p><i>После окончания времени провести разбор решения, анализируя код.</i></p>	 <p>Сцена 15</p>	
	<p>Подведем предварительный итог. Цикл работает столько раз, какое число указано внутри команды range. По умолчанию отсчет начинается с нуля. Чтобы изменить порядок – надо указать два числа: старт и стоп. Количество повторов равно стоп минус старт Вместо чисел можно указать целочисленные переменные.</p>	 <p>Сцена 16</p>	
	<p><i>После окончания дикторского текста запускается таймер на 4 мин.</i> Задание 2. Алгоритм «От и до»</p> <ul style="list-style-type: none"> – Запросить первое целое число и занести ответ в переменную start – Запросить второе целое число и занести ответ в переменную stop – Организовать цикл от start до stop Напечатать счетчик 	 <p>Сцена 17</p>	

<p>Разбор задания 2. Введем 1 и 6, и увидим, что код работает. Код программы может выглядеть так.</p> <pre>start = int(input("Введи первое число: ")) stop = int(input("Введи второе число: ")) for i in range(start, stop): print(i)</pre>	<div><div>Циклы с ограниченным количеством повторений</div><div><div><pre>start = int(input("Введи первое число: ")) stop = int(input("Введи второе число: ")) for i in range(start, stop): print(i)</pre></div><div><pre>----- ПОЧАТОК: програма.py ----- Задан диапазон чисел: 1 Введи второе число: 6 1 2 3 4 5</pre></div></div></div>	Сцена 18
<p>Разбор задания 2.</p> <p>Если ввести два одинаковых числа? Цикл работать не будет. Это понятно, ведь старт совпал со стоп.</p> <p>Если число старт будет больше числа стоп? По идее числа должны пойти в обратную сторону, по нисходящей? Но нет, цикл по-прежнему не работает.</p>	<div><div>Циклы с ограниченным количеством повторений</div><div><div><pre>start = int(input("Введи первое число: ")) stop = int(input("Введи второе число: ")) for i in range(start, stop): print(i)</pre></div><div><pre>----- ПОЧАТОК: програма.py ----- Задан диапазон чисел: 5 Введи второе число: 5</pre></div></div><div>start == stop – цикл не работает</div></div>	Сцена 19
<p><i>После окончания дикторского текста запускается таймер на 15 сек.</i></p> <p>Задание 2. Алгоритм «От и до» (продолжение)</p> <p>– Поставить внутри скобок range третье число: -1</p>	<div><div>Циклы с ограниченным количеством повторений</div><div><div><pre>start = int(input("Введи первое число: ")) stop = int(input("Введи второе число: ")) for i in range(start, stop): print(i)</pre></div><div><div>Меню нумерации</div><div>Поставить внутри скобок range третье число: -1</div></div></div><div>00:15</div></div>	Сцена 20

	<p>Разбор задания 2 (продолжение). Введем start больше, чем stop. Код дописан следующими образом:</p> <pre>start = int(input("Введи первое число: ")) stop = int(input("Введи второе число: ")) for i in range(start, stop, -1): print(i)</pre> <p><i>После окончания времени провести разбор решения, анализируя код.</i></p>	 <p>Сцена 21</p>	
<p>Теоретический блок 2.</p> <p>Конечный цикл.</p>	<p><i>При необходимости преподаватель может поставить ролик на паузу и дать дополнительные пояснения по материалу</i></p> <p>Таким образом, мы можем делать нисходящий цикл, для этого старт должен быть больше стопа и необходим третий параметр в виде отрицательного числа.</p> <p>Мы можем делать восходящий цикл – для этого старт должен быть меньше стопа. Третий параметр можно не указывать.</p>	 <p>Сцена 22</p>	22
<p>Блок заданий.</p> <p>Практические задания:</p> <p>Задание 3</p>	<p><i>После окончания дикторского текста запускается таймер на 8 мин.</i></p> <p>Задание 3. Алгоритм «Выбор цикла»</p> <p>Запросить первое целое число и занести ответ в переменную start Запросить второе целое число и занести ответ в переменную stop Если start == stop Напечатать «цикл невозможен» Иначе если start < stop Организовать восходящий цикл с выводом счетчика на экран Иначе Организовать нисходящий цикл с выводом счетчика на экран</p>	 <p>Сцена 23</p>	23 24

	<p>Разбор задания 3. Протестируем наш код с разными входными данными:</p> <pre> start = int(input("Введи первое число: ")) stop = int(input("Введи второе число: ")) if start == stop: print("Цикл невозможен") elif start < stop: for i in range(start, stop): print(i) else: for i in range(start, stop, -1): print(i) </pre> <p><i>После окончания времени провести разбор решения, анализируя код.</i></p>	 <p>Сцена 24</p>	
<p>Блок завершения занятия. Рефлексия. Сообщение домашнего задания</p>	<p><i>Завершаем демонстрацией ролика и кратким обобщением материалов занятия.</i></p> <p>Подведем итоги:</p> <ul style="list-style-type: none"> — узнали, что команды для создания конечного цикла: for, in, range — восходящий цикл: от меньшего к большему, от 1 до 3х параметров внутри range; — нисходящий цикл: от большего к меньшему, 3 параметра внутри range. <p><i>Преподаватель дает ученикам домашнее задание к следующему занятию (Приложение 2).</i></p>	 <p>Сцена 25</p>	25

Приложение 2

Домашнее задание

Придумайте примеры конечного восходящего и нисходящего цикла.

Посчитайте, какие параметры нужно было бы указать лифту, который едет с этажа 1 на 16, а потом с 16 на 2 этаж

Практика

Проект «Скорости циклов»

Цель проекта: сравнить скорость работы циклов **while** и **for**.

Запросите количество повторов. Организуйте с помощью циклов **while** и **for** вывод на экран чисел от 0 до количества повторов (не включая последний).

Определите время выполнения каждого из циклов и выведите его на экран.

Для определения времени используйте метод **monotonic()** из библиотеки **time**.

```
from time import monotonic

start = monotonic()

# код, длительность выполнения
# которого нужно отследить

end = monotonic()
delta_time = end - start
```

зафиксировали
начальное время

зафиксировали
конечное время

вычислили длительность
выполнения кода

Этот метод возвращает значение текущего времени в долях секунды, но контрольная точка (точка отсчета) не определена, поэтому можно использовать только разницу между значениями при вызовах этого метода.

** Обратите внимание, время зависит от мощности компьютера и продолжительности цикла.*

Проект «Слово по буквам»

Запросите слово или фразу. Выведите эту фразу по буквам на экран, каждая буква выводится на новой строке.

Для перебора букв в строке **word** можно использовать

for i in word:

В этом случае в **i** будут поочередно помещаться буквы из строки, записанной в переменную **word**.

Приложение 3

Краткие организационно-методические рекомендации по организации работы на занятии

«Циклы с ограниченным количеством повторений».

В начале занятия можно рассмотреть домашнее задание – собственная функция модуля **math**. Поинтересуйтесь, что ребята придумали, выясните – имеется ли эта возможность в модуле и если нет – как функция могла бы работать.

Отдельно надо проговорить что такое модуль, зачем нужен, как подключается и с какими функциями математического модуля мы познакомились (**floor**, **ceil**), чем они отличаются от **round**. Что еще запомнилось ребятам из возможностей математического модуля?

Перед просмотром блока повторения из ролика необходимо раздать дидактический материал для выполнения заданий из блока повторение (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.

После теоретического блока можно записать на доске служебные команды, необходимые для создания конечного цикла. Обращайте внимание на то, что, если строка заканчивается двоеточием – следующий блок команд идет с отступом, как в операторах условия.

Дополнения к первым двум заданиям носят минимальный характер, поэтому и время отпущено на это мало. Эти проекты носят тренировочный характер.

Главное, чтобы ребята поняли какие параметры функции **range** за что отвечают.

Если материал будет усвоен быстро – можно дать дополнительную информацию о третьем параметре, который является шагом для счетчика и может применяться и в восходящем цикле. С его помощью можно выводить не все числа подряд, а с настраиваемым интервалом.

Основной проект урока – задание 3. Перед его началом можно повторить информацию об использовании и синтаксисе операторов условия.

Приложение 4

Алгоритм «Подсчёт частоты появления символа в строке»

Изучите предложенный алгоритм и напишите код программы. Вы можете сверить свой код с образцом решения.

Условие

Даны фраза и буква.

Задача: определить сколько раз заданная буква встречается в заданной фразе, не используя метод count().

Алгоритм «Частота появления символа в строке»

- Запросить фразу и занести ее в переменную **text**
- Запросить букву и занести ее в переменную **x**
- Создать переменную **k** с нулевым значением
- Организовать конечный цикл для прохода по всем символам текста
 - Если текущий символ равен заданному символу
 - Увеличить **k** на 1
- Вывести значение **k**