



Методические рекомендации по теме «Сложные списки и методы работы с ними»

Цель:

- дать представление об использовании сложных списков в языке Python.

Задачи:

- практика применения списков в 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 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 о понятии «список» в программировании и об использовании сложных списков в Python.

Метапредметные: обучающиеся получают возможность владения общепредметными понятиями «список», «сложный список», «структура», «индекс»; 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.

Проблемная дискуссия по вопросам:

- Как вы понимаете понятие «сложный список»?
- Что определяет сложность списка?
- Какие операции можно делать со списками?
- Приведите примеры сложных списков?

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

- Список – это упорядоченный изменяемый объем данных

– Сложный список позволяет с большей точностью структурировать данные
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.

Сложные списки.

Новый материал излагается в сопровождении видеоролика, рекомендуется разместить на доске или флип-чарте изображения объектов, сопровождающих материалы по теме.

Обсуждением вопросов по просмотренным материалам:

- Почему сложный список называют «многомерным»?
- В каких ситуациях возникает потребность создания именно сложных списков?

- Какие возможности сортировки данных нам дают сложные списки?

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. сцена 8 – 11*

5. Блок заданий.

Проекты: «Поэты России», «Информаторий».

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

«Поэты России»: включает *практическое задание 1* по созданию сложного списка, его сортировке и вывода элементов ложного списка с использованием отдельных индексов его элементов.

После выполнения задания ученики получают работающий программный продукт – программа позволяет создать и вручную заполнить данные для сложного списка, сортировать элементы списка, выводить его целиком.

«Информаторий»: включает *практическое задание 2* для учеников с последующим разбором. Задание представляет собой продолжение программного проекта про поэтов для создания возможности выводить выборочную информацию из сложного списка, а также сортировать сложный список по одному из внутренних индексов основной ячейки списка.

После выполнения задания ученики получают работающую программу, которая позволяет обрабатывать данные списка по выборочному признаку.

Блок включает обсуждение по вопросам:

- Как можно вывести отдельно определенные значения списка используя индексы?
- Какие типы индексов имеет каждая ячейка сложного списка?

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

**см. сцены 12 – 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*

Приложение 1

Сценарный план видеоролика

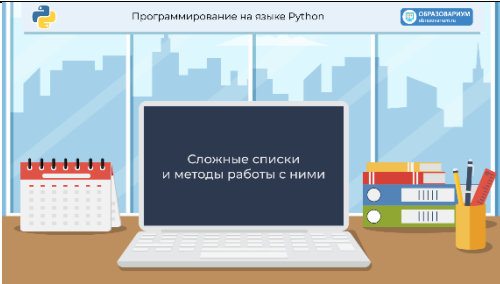
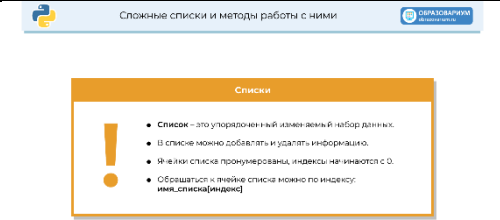
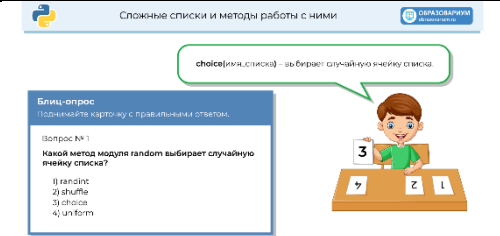
В таблице «Содержание видеоролика» представлены:

- 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.

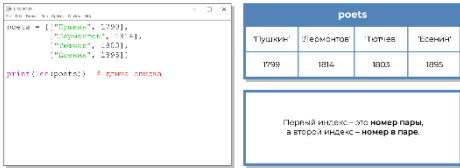
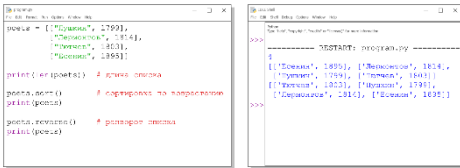
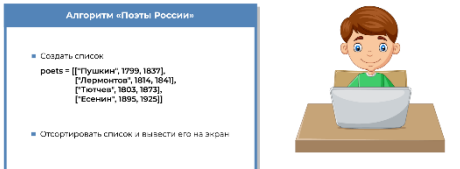
Таблица. Содержание видеоролика

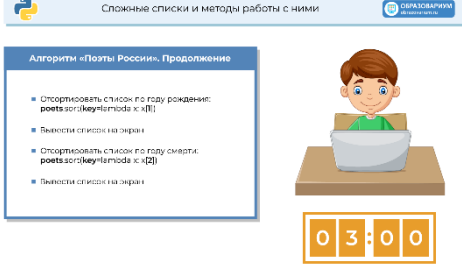
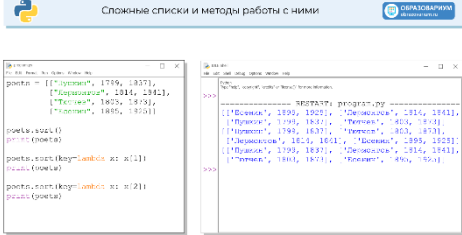
Название блока	Содержание блока и комментарии	Фрагменты из видеоролика	№ сцен
----------------	--------------------------------	--------------------------	--------

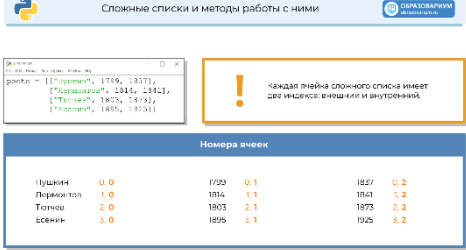
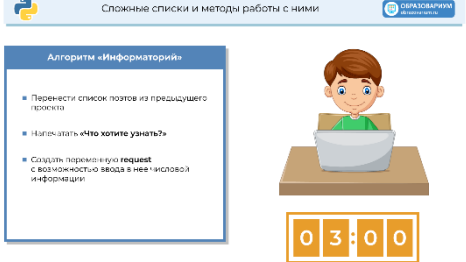
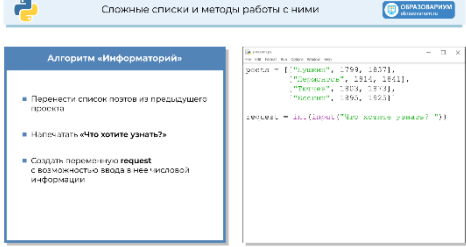
<p>Вводный блок. Мы узнаем</p>	<p><i>Обозначаем ученикам тему и цели урока.</i></p> <p>Сложные списки и методы работы с ними</p>	 <p>Сцена 1</p>	<p>1 2</p>
	<p>Список – это упорядоченный изменяемый набор данных. Изменяемый – значит можно добавлять, удалять информацию. Упорядоченный – значит ячейки расположены по порядку, начиная с нуля. Адрес ячейки называется индексом и представляет собой число в квадратных скобках.</p> <p>Теперь познакомимся со сложными списками, которые позволяют лучше структурировать имеющуюся у нас информацию.</p>	 <p>Сцена 2</p>	
<p>Блок повторения. 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. Какой метод модуля random выбирает случайную ячейку списка?</p> <ol style="list-style-type: none"> 1) randint 2) shuffle 3) choice 4) uniform <p><i>Ответ 3. choice(имя_списка) – выбирает случайную ячейку списка.</i></p>	 <p>Сцена 3</p>	<p>3 4 5 6 7</p>









	<p>Второй вопрос. Какая команда добавляет данные в список?</p> <ol style="list-style-type: none"> 1) append 2) len 3) sort 4) reverse <p><i>Ответ 1. имя_списка.append(данные) – добавляет данные в конец списка.</i></p>	<p>Сложные списки и методы работы с ними</p> <p>имя_списка.append(данные) – добавляет данные в конец списка.</p> <p>Блиц-опрос Поднимайте карточку с правильным ответом.</p> <p>Вопрос № 2 Какая команда добавляет данные в список?</p> <ol style="list-style-type: none"> 1) append 2) len 3) sort 4) reverse <p>Сцена 4</p>	
	<p>Третий вопрос. Сколько раз сработает цикл?</p> <p>for i in range (0, 7, 4):</p> <p>Поднимите карточку с соответствующим числом.</p> <p><i>Ответ 2. Цикл проработает 2 раза. Счетчик получит значения 0 и 4.</i></p>	<p>Сложные списки и методы работы с ними</p> <p>Цикл проработает 2 раза. Счетчик получит значения 0 и 4.</p> <p>Блиц-опрос Поднимайте карточку с правильным ответом.</p> <p>Вопрос № 3 Сколько раз сработает цикл? for i in range (0, 7, 4):</p> <p>Поднимите карточку с соответствующим числом.</p> <p>Сцена 5</p>	
	<p>Четвертый вопрос. Какая команда может дать дробное число?</p> <ol style="list-style-type: none"> 1) uniform 2) round 3) float 4) все три <p><i>Ответ 4. Все три команды могут дать дробное число.</i></p>	<p>Сложные списки и методы работы с ними</p> <p>Все три команды могут дать дробное число.</p> <p>Блиц-опрос Поднимайте карточку с правильным ответом.</p> <p>Вопрос № 4 Какая команда может дать дробное число?</p> <ol style="list-style-type: none"> 1) uniform 2) round 3) float 4) все три <p>Сцена 6</p>	
	<p>Пятый вопрос. Выберите правильный вариант среза "дом" из слова "рандом".</p> <ol style="list-style-type: none"> 1) [3:] 2) [4: 6] 3) [3: 7] 4) [-3: -1] 	<p>Сложные списки и методы работы с ними</p> <p>* Р А Н Д О М *</p> <p>0 1 2 3 4 5</p> <p>"дом" == "рандом"[3:]</p> <p>Блиц-опрос Поднимайте карточку с правильным ответом.</p> <p>Вопрос № 5 Выберите правильный вариант среза "дом" из слова "рандом".</p> <ol style="list-style-type: none"> 1) [3:] 2) [4: 6] 3) [3: 7] 4) [-3: -1] <p>Сцена 7</p>	

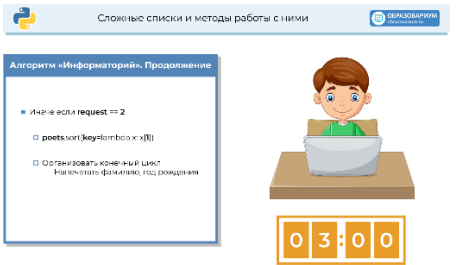
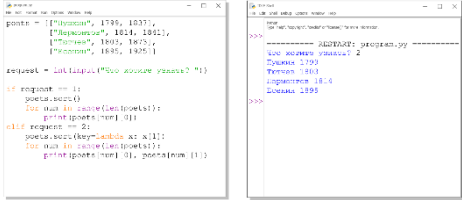
	<div>"РАНДОМ"</div> <div>0 1 2 3 4 5</div> <div>Ответ 1. "дом" == "рандом"[3:]</div>								
<div>Теоретический блок.</div> <div>Сложные списки</div>	<div>Обычный список позволяет нам хранить любое количество самой разнообразной информации, однако иногда хочется данную информацию структурировать.</div> <div>Чтобы она не была одной общей кучей данных</div>	<div><div>Сложные списки и методы работы с ними</div><div><div>Список</div><table><tr><td>42</td><td>43</td><td>'математика'</td><td>'литература'</td><td>47</td><td>'информатика'</td></tr></table></div><div>Сцена 8</div></div>	42	43	'математика'	'литература'	47	'информатика'	8 9 10 11
42	43	'математика'	'литература'	47	'информатика'				
	<div>Рассмотрим пример. У нас есть фамилии поэтов и их год рождения: Пушкин 1799 Лермонтов 1814 Тютчев, 1803, Есенин 1895 Мы можем занести все это в один список, который будет иметь восемь ячеек. При попытке отсортировать данный список по алфавиту получится ошибка, поскольку нельзя сравнивать тексты и числа одновременно друг с другом. Как же решить данную проблему?</div>	<div><div>Сложные списки и методы работы с ними</div><div><div><pre>poets = ["Пушкин", 1799, "Лермонтов", 1814, "Тютчев", 1803, "Есенин", 1895] print(len(poets)) poets.sort()</pre></div><div><pre>poets = ["Пушкин", 1799, "Лермонтов", 1814, "Тютчев", 1803, "Есенин", 1895] print(len(poets)) poets.sort()</pre></div></div><div>Сцена 9</div></div>							

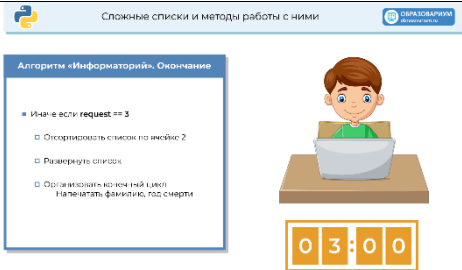
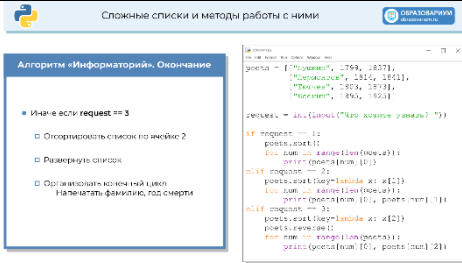
	<p>Надо сделать сложный список. Другое его название – многомерный. Суть его в том, что каждая ячейка будет еще одним небольшим списком. Вот таким образом. Теперь наш список имеет только четыре ячейки. Но почему?</p> <p>Дело в том, что считаются только основные ячейки, которыми теперь являются связанные между собой пары данных. Можно сказать, что у нас появилась некая таблица четыре на два. Всего в ней восемь элементов, но каждый теперь имеет не один индекс, а два. Первый – это номер пары, а второй – номер в паре.</p> <p>И теперь мы можем сортировать этот список по возрастанию ... Или по убыванию</p> <p>Обратите внимание, что метод <code>sort</code> берёт за основу первое значение в паре. В нашем случае это фамилия.</p> <p>Если же надо провести сортировку по году рождения, то надо указать так называемый ключ – дополнительный параметр.</p> <p>Давайте посмотрим, как это работает.</p>	<div data-bbox="1496 228 1966 539"> <p>Сложные списки и методы работы с ними</p>  <p>Сцена 10</p> </div> <div data-bbox="1496 675 1966 986"> <p>Сложные списки и методы работы с ними</p>  <p>Сцена 11</p> </div>	
<p>Блок заданий.</p> <p>Практические задания:</p> <p>Задание 1</p>	<p>Перепишите в новый проект наш список и дополните каждую пару третьим элементом – годом смерти. Не забывайте отделять запятыми данные друг от друга. Каждый мини-список имеет свои квадратные скобки. И еще не забудьте основные скобки вначале и в конце списка</p> <p><i>После окончания дикторского текста запускается таймер на 4 мин.</i></p> <p>Задание 1. Алгоритм «Поэты России»</p> <ul style="list-style-type: none"> Создать список <code>poets = ["Пушкин", 1799, 1837], ["Лермонтов", 1814, 1841], ["Тютчев", 1803, 1873], ["Есенин", 1895, 1925]]</code> Отсортировать список и вывести его на экран 	<div data-bbox="1496 1007 1966 1318"> <p>Сложные списки и методы работы с ними</p>  <p>Сцена 12</p> </div>	<p>12 13 14</p>


	<p><i>После окончания дикторского текста запускается таймер на 3 мин.</i></p> <p>Задание 1. Алгоритм «Поэты России» (продолжение)</p> <ul style="list-style-type: none"> Отсортировать список по году рождения: <code>poets.sort(key=lambda x: x[1])</code> Вывести список на экран Отсортировать список по году смерти: <code>poets.sort(key=lambda x: x[2])</code> Выведите список на экран 	 <p>Сцена 13</p>
	<p>Разбор задания 1. Ваш код может выглядеть так.</p> <pre>poets = [["Пушкин", 1799, 1837], ["Лермонтов", 1814, 1841], ["Тютчев", 1803, 1873], ["Есенин", 1895, 1925]] poets.sort() print(poets) poets.sort(key=lambda x: x[1]) print(poets) poets.sort(key=lambda x: x[2]) print(poets)</pre> <p><i>Мы получаем на экране три варианта сортировки списка. Если добавить туда другие данные: имя, отчество, количество произведений и т.д., то можно настроить сортировку и по этим параметрам</i></p>	 <p>Сцена 14</p>

<p>Теоретический блок.</p>	<p>В рассмотренном проекте весь список выводится целиком. Как сделать так, чтобы можно было выводить отдельно фамилии, и отдельно даты?</p> <p>Каждая ячейка у нас теперь имеет два индекса: внешний и внутренний. Внешние начинаются от нуля до трех, внутренние – от нуля до двух. Если мы возьмем наш первоначальный список, то фамилии будут иметь следующие индексы.</p> <p>Ячейки года жизни вторым индексом будут иметь единицу, а годы смерти – двойку. Чем-то это похоже на систему координат, где есть ось икс и ось игрек.</p>	 <p>Сложные списки и методы работы с ними</p> <p>Сцена 15</p>	<p>15</p>
<p>Блок заданий. Практические задания: Задание 2</p>	<p><i>После окончания дикторского текста запускается таймер на 3 мин.</i></p> <p>Создадим новый проект и перенесем в него наш список поэтов.</p> <p>Задание 2. Алгоритм «Инфоматорий»</p> <ul style="list-style-type: none"> • Перенести список поэтов из предыдущего проекта. • Напечатать «Что хотите узнать?» • Создать переменную request с возможностью ввода в нее числовой информации 	 <p>Сложные списки и методы работы с ними</p> <p>Сцена 16</p>	<p>16 17</p>
	<p>Разбор задание 2. Код программы будет выглядеть так:</p> <pre>poets = [["Пушкин", 1799, 1837], ["Лермонтов", 1814, 1841], ["Тютчев", 1803, 1873], ["Есенин", 1895, 1925]] request = int(input("Что хотите узнать? "))</pre>	 <p>Сложные списки и методы работы с ними</p> <p>Сцена 17</p>	

<p>Теоретический блок.</p>	<p>Теперь решим, какую информацию мы можем запросить.</p> <p>Варианты запроса:</p> <ol style="list-style-type: none"> 1 – только фамилии по алфавиту, 2 – фамилии + год рождения по возрастанию 3 – фамилия + год смерти по убыванию 	<div data-bbox="1509 228 1966 260">  Сложные списки и методы работы с ними  </div> <div data-bbox="1509 308 1966 478"> <div data-bbox="1525 308 1787 443"> <p>Варианты запроса</p> <ol style="list-style-type: none"> 1 – только фамилии по алфавиту 2 – фамилия + год рождения по возрастанию 3 – фамилия + год смерти по убыванию </div>  </div> <p>Сцена 18</p>	<p>18</p>
<p>Блок заданий. Практические задания: Задание 2</p>	<p><i>После окончания дикторского текста запускается таймер на 3 мин.</i></p> <p>Задание 2. Алгоритм «Инфоматорий». Продолжение</p> <ul style="list-style-type: none"> • Перенести список поэтов из предыдущего проекта. • Если request == 1 <ul style="list-style-type: none"> ○ Отсортируйте список по умолчанию ○ for num in range(len(poets)): print(poets[num][0]) <p><i>Создадим первое условие, заодно узнаем, как выводить часть ячеек, используя цикл.</i></p>	<div data-bbox="1509 592 1966 624">  Сложные списки и методы работы с ними  </div> <div data-bbox="1509 639 1966 863"> <div data-bbox="1503 639 1727 842"> <p>Алгоритм «Инфоматорий». Продолжение</p> <ul style="list-style-type: none"> • Если request == 1 <ul style="list-style-type: none"> ○ Отсортируйте список по умолчанию ○ for num in range(len(poets)): print(poets[num][0]) </div>  <div data-bbox="1783 807 1917 863"> <p>03:00</p> </div> </div> <p>Сцена 19</p>	<p>19 20</p>
	<p>Разбор задание 2 (продолжение). Код программы будет выглядеть так: <i>... дописывается к существующему программному коду</i></p> <pre> if request == 1: poets.sort() for num in range(len(poets)): print(poets[num][0]) </pre>	<div data-bbox="1509 1066 1966 1098">  Сложные списки и методы работы с ними  </div> <div data-bbox="1509 1114 1966 1321"> <div data-bbox="1503 1114 1727 1316"> <p>Алгоритм «Инфоматорий». Продолжение</p> <ul style="list-style-type: none"> • Если request == 1 <ul style="list-style-type: none"> ○ Отсортируйте список по умолчанию ○ for num in range(len(poets)): print(poets[num][0]) </div> <div data-bbox="1738 1114 1960 1316"> <pre> poets = [("Пушкин", 1799, 1837), ("Лермонтов", 1814, 1841), ("Тютчев", 1803, 1873), ("Маяковский", 1899, 1929)] def sort_by_death_year(poets): return sorted(poets, key=lambda poet: poet[2]) if request == 1: poets.sort() for num in range(len(poets)): print(poets[num][0]) </pre> </div> </div> <p>Сцена 20</p>	

	<p>После окончания дикторского текста запускается таймер на 3 мин.</p> <p>Задание 2. Алгоритм «Инфоматорий». Продолжение</p> <ul style="list-style-type: none"> Иначе если request == 2 <ul style="list-style-type: none"> poets.sort(key=lambda x: x[1]) Организовать конечный цикл <ul style="list-style-type: none"> Напечатать фамилию, год рождения <p>Теперь создадим второе условие. Если введено число два, то отсортировать по году рождения (по ячейке номер один).</p>	 <p>Сцена 21</p>	<p>21 22</p>
	<p>Разбор задание 2(продолжение). Код программы будет выглядеть так: ... дописывается к существующему программному коду</p> <pre> elif request == 2: poets.sort(key=lambda x: x[1]) for num in range(len(poets)): print(poets[num][0], poets[num][1]) </pre> <p>Мы можем также выводить сначала год рождения, а потом фамилию. Надо всего лишь поменять вторые индексы</p>	 <p>Сцена 22</p>	

	<p>После окончания дикторского текста запускается таймер на 3 мин.</p> <p>Задание 2. Алгоритм «Инфоматорий». Продолжение</p> <ul style="list-style-type: none"> Иначе если request == 3 <ul style="list-style-type: none"> Отсортировать список по ячейке 2 Развернуть список Организовать конечный цикл <p>Напечатать фамилию, год смерти</p> <p><i>Третье условие – если введено число три, то отсортировать по году смерти (ячейка номер два) и развернуть список наоборот.</i></p>	<div data-bbox="1503 228 1962 499">  <p>Сложные списки и методы работы с ними</p> <p>Алгоритм «Инфоматорий». Окончание</p> <ul style="list-style-type: none"> Иначе если request == 3 <ul style="list-style-type: none"> Отсортировать список по ячейке 2 Развернуть список Организовать конечный цикл <p>Напечатать фамилию, год смерти</p> <p>03:00</p> </div> <p>Сцена 23</p>	<p>23 24</p>
	<p>Разбор задание 2(продолжение). Код программы будет выглядеть так: ... дописывается к существующему программному коду</p> <pre> elif request == 3: poets.sort(key=lambda x: x[2]) poets.reverse() for num in range(len(poets)): print(poets[num][0], poets[num][2]) </pre> <p><i>При желании – мы можем добавить еще условий, чтобы разнообразить его функционал.</i></p>	<div data-bbox="1503 667 1962 930">  <p>Сложные списки и методы работы с ними</p> <p>Алгоритм «Инфоматорий». Окончание</p> <ul style="list-style-type: none"> Иначе если request == 3 <ul style="list-style-type: none"> Отсортировать список по ячейке 2 Развернуть список Организовать конечный цикл <p>Напечатать фамилию, год смерти</p> <pre> poets = [("Пушкин", 1799, 1837), ("Лермонтов", 1814, 1841), ("Тютчев", 1803, 1873), ("Маяковский", 1899, 1928)] request = int(input("Введите номер задачи: ")) if request == 1: poets.sort() print(poets[0][0]) elif request == 2: poets.sort(key=lambda x: x[2]) print(poets[-1][0]) elif request == 3: poets.sort(key=lambda x: x[2]) poets.reverse() for num in range(len(poets)): print(poets[num][0], poets[num][2]) </pre> </div> <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> <p>Мы узнали:</p> <ul style="list-style-type: none"> ▪ Сложные списки используют, если есть данные, логически связанные друг с другом. ▪ По умолчанию параметром для сортировки или разворота является первая внутренняя ячейка. ▪ Доступ к ячейке осуществляется по двум индексам: внешнему и внутреннему. <p><i>Преподаватель дает ученикам домашнее задание к следующему занятию (Приложение 2).</i></p>	 <p>Сцена 25</p>	<p>25</p>
--	--	---	-----------

Приложение 2

Домашнее задание

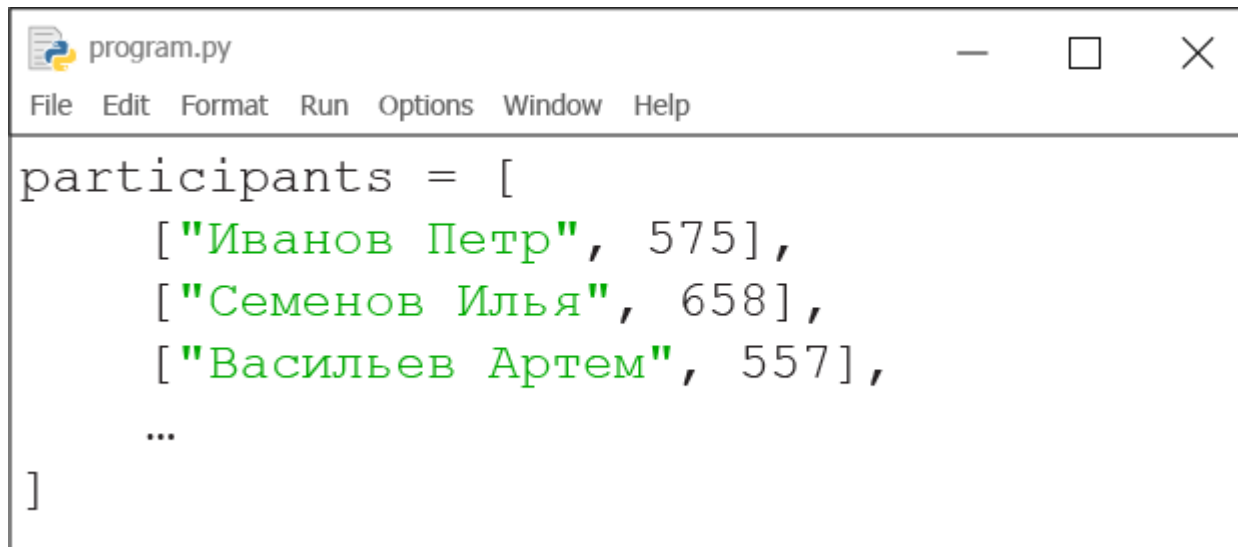
Какие еще варианты вывода информации можно было бы осуществить в проекте данного урока?

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

Практика

Проект «Победители олимпиады»

Результаты олимпиады поместите в сложный список, в каждой ячейке которого будет содержаться информация об участнике олимпиады и набранных баллах. Например, так:

A screenshot of a Python IDE window titled "program.py". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

```
participants = [  
    ["Иванов Петр", 575],  
    ["Семенов Илья", 658],  
    ["Васильев Артем", 557],  
    ...  
]
```

Выведите на экран тройку победителей олимпиады.

Проект «График дежурств в походе»

Создайте три списка с исходными данными: первый список с именами ребят, которые пошли в поход, второй – список обязанностей, третий – обозначение дней похода. Количество туристов и количество обязанностей должно совпадать. Случайным образом распределите обязанности между ребятами и составьте график дежурств. Учтите, что обязанности у одного человека, по возможности, не должны повторяться. В итоге у вас должен получиться сложный список. Выведите график дежурств на экран.

Краткие организационно-методические рекомендации по организации работы на занятии

«Сложные списки и методы работы с ними».

В начале занятия необходимо повторить материал по теме «Списки». Что это такое, какими свойствами обладает, синтаксис, методы, способы вывод на экран. Не лишним будет напомнить такой термин как «индекс», который также как и в срезах, начинается с нуля. Поинтересуйтесь, какие примеры придумали ребята в качестве домашнего задания. Это могут быть варианты жеребьевки спортивных команд, билеты на экзамен, лотерея.

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.

Главной сложностью при работе со списками является понимание индексации ячеек. Для этого можно во время теоретической части поставить ролик на паузу. И записать список с данными на доске с указанием индексов.

В обоих проектах используется один и тот же список, который можно скопировать для экономии времени. Можно не ограничиваться 4-мя персонами и добавить еще других поэтов. В этом случае различные типы сортировки будут гораздо нагляднее. Сортировка по ключу осуществляется через так называемую лямбда-функцию. Поскольку тема функций в курсе не рассматривается, то нужно просто дать ребятам формулу в качестве готового решения.

Еще следует обратить внимание, что метод **reverse** только переворачивает список, а не сортирует его по убыванию.