Муниципальное бюджетное общеобразовательное учреждение
«Горловская средняя общеобразовательная школа»
Скопинского муниципального района
Рязанской области

Технологические карты занятий: Python для обработки данных

Сценарий занятия 1
Тема: История языков программирования. Компиляция и интерпретация.
Класс: 10-11
Цель занятия: знакомство обучающихся с языком программирования с историей языков программирования, с понятием компиляции и интерпретации.
Задачи занятия:
Образовательные: познакомить обучающихся с историей языков программирования, с особенностями языка; сформировать первичные знания по применению изученного материала.
Воспитательные: развивать информационную культуру обучающихся; способность к самостоятельной и коллективной деятельности.
Развивающие: совершенствование умения анализировать, сравнивать, систематизировать и обобщать, развитие коммуникативных умений обучающихся.
Планируемые результаты:
Предметные: владение информацией о языках программирования, представление об компиляции и интерпретации
Личностные: сформированность навыков сотрудничества со сверстниками; готовность и способность к образованию, в том числе самообразованию.
Метапредметные: умение контролировать и корректировать учебную деятельность.
Модель обучения: проблемное обучение.
Этапы урока:

1. Мотивация к учебной деятельности.
2. Актуализация знаний и фиксирование затруднений.
3. Построение проекта выхода из затруднения.
4. Реализация построенного проекта.
5. Знакомство с понятием искусственного интеллекта
6. Рефлексия.

Название занятие: Приобретение знание – Введение в основы программирования, понятие искусственного интеллекта	**Подход:** Интерактивное занятие с преподавателем. Самостоятельная работа с онлайн ресурсами.
Работа в парах	
Обучающиеся должны: -Повторить тему «Понятие алгоритмов, языки программирования» -Получить представление об использовании данной темы в различных сферах жизни	
Результат обучения: сформированные знания по теме «История языков программирования. Компиляция и интерпретация»	
Необходимые знания, умения и навыки: -основные знания из курса информатики средней школы	
Основные понятия: - Основные этапы развития языков программирования; - Трансляция и транслятор - Компиляция и интерпретация	

Используемые материалы: - Рабочие листы - Стикеры
Используемые ресурсы: -Интерактивный тест на сервисе Quizizz для постановки проблемной задачи. -В программе PowerPoint создаётся обучающая презентация, содержащая лекционный материал по теме. -Просмотр видеоролика с платформы УрокЦифры, посвящённого искусственному интеллекту (урок 1 2020 года) -В качестве закрепления полученного материала предлагается интерактивный тест с помощью сервиса LearningApps.
Обсуждаемые этические и социальные проблемы: - Внедрение искусственного интеллекта во все сферы жизни современного человека

Ход занятия:

№	Время (мин)	Вид работы	Описание	Цель
1	3	Мотивация к учебной деятельности	Обсуждения чат-ботов, встречающихся в известных социальных сетях и сайтах	Подготовить обучающихся к освоению учебного материала, формирование необходимых компетенций.
2	7	Актуализация знаний и фиксирование затруднений	Выполнение интерактивного теста, созданного с помощью сервиса Quizizz, организуется вводная работа по теме.	Систематизировать изученный материал, формировать необходимые компетенции.
3	3	Построение проекта выхода из затруднения	Обсуждение возникших вопросов во время выполнения интерактивного теста.	Систематизация изученного материала в курсе основной школы, постановка целей и задач занятия.
4	15	Реализация построенного проекта	Изучение нового материала по презентации PowerPoint **1. Программа. Язык программирования** Программу можно представить как набор	Сформировать представление о языках программирования.

			последовательных команд (алгоритм) для определенного объекта (исполнителя), который должен их выполнить для достижения той или иной цели. Например, условно запрограммировать можно человека, если составить ему инструкцию "как собрать компьютер", и он примется ее исполнять. Очевидно, что инструкция будет на естественном языке (русском, английском или др.). Однако, обычно принято программировать не людей, а вычислительные машины. Трудность заключается в том, что такие машина не в состоянии понять наш язык. Для "инструктирования" вычислительных машин разработаны и разрабатываются специальные языки, называемые **языками программирования**. Языки программирования характеризуются однозначностью при формировании фразы (в выражении нельзя менять слова местами), а также ограниченным набором слов-команд. Другими словами, языки программирования являются **формальными**. **2. Основные этапы развития языков программирования** **Машинный язык** — это единственный способ	

			взаимодействия с электронно-вычислительными машинами. Первые программы писались именно на нем, т. к. других средств «общения» человека и компьютера еще не было. Каждую команду машинного языка выполняет определенное электронное устройство. Данные и команды записываются в цифровом виде (например, в шестнадцатеричной или двоичной системах счисления). Понять программу на таком языке человеку очень сложно; к тому же даже небольшая программа будет состоять из множества строк кода. В довершении всего, у каждой вычислительной машины свой машинный язык; следовательно программа, написанная для одной ЭВМ, не будет работать на другой (придется писать снова). Людям, в отличие от машин, более понятны слова, чем наборы цифр. Стремление человека оперировать словами и не цифрами привело к появлению **ассемблеров**. Это языки, в которых вместо численного обозначения команд и областей памяти используются буквенные. Но тут сразу возникает проблема: машина не в состоянии понять набор букв. Необходим какой-	

			нибудь «переводчик» на ее родной машинный язык. Так был придуман **транслятор** — специальная программа, преобразующая программный код с того или иного языка программирования в машинный код. Ассемблеры и сегодня находят применение, т.к. системные программы (обслуживающие работу аппаратного обеспечения), написанные на ассемблере могут работать быстрее, чем аналогичные программы, написанные на других языках программирования. После ассемблеров наступил рассвет языков так называемого **высокого уровня**. Для этих языков потребовалось разрабатывать более сложные трансляторы. Программные коды, написанные на языках высокого уровня, обладают логичной структурой. Это облегчает разработку программы и ее отладку. В отличие от ассемблеров, которые все еще остаются привязанными к своим типам машин, языки высоко уровня обладают **переносимостью**. Т.е., написав один раз программу, программист может выполнить ее на любой машине. Следующим крупным этапом в эволюции программирования было появление **объектно-**	

			ориентированных языков (ООП). Их отличие от языков высокого уровня заключается в возможности отстранения от алгоритма выполнения программы. С помощью таких языков разработчик как бы оперирует виртуальными объектами. На сегодняшний день, реализация больших и сложных проектов осуществляется в основном с помощью ООП. **3. Разнообразия языков программирования** Разнообразие языков весьма велико. Это может показаться странным: если написанную программу можно перенести на любую машину и выполнить там, то зачем придумывать такое множество языков? Однако все немного сложнее. Можно сказать, что нет идеального языка, каждый чем-то хорош, а в чем-то уступает другому инструменту. Многие программисты старались и стараются придумать свой язык обладающий теми или иными преимуществами. Можно лишь условно разделить языки по определенным критериям. Например, по типу решаемых задач (язык системного или прикладного назначения), по степени ориентации на решение узкого круга задач (проблемно-	

			ориентированные или универсальные). **4. Трансляция** Ранее было сказано, что для перевода кода с одного языка программирования (например, высокого уровня) на другой (например, машинный язык) требуется специальная программа — **транслятор**. Механизм этого перевода весьма сложен, однако выделяют два основных способа **трансляции** — это **компиляция** программы или ее **интерпретация**. При компиляции исходный программный код сразу целиком переводится в машинный. Создается исполняемый файл, который уже никак не связан с исходным кодом. Выполнение исполняемого файла обеспечивается операционной системой самостоятельно. При интерпретации выполнение кода происходит построчно. Интерпретатор, выполняя программу, напрямую взаимодействует с операционной системой. Выполнение откомпилированной программы происходит быстрее, т.к. она представляет собой готовый машинный код. Однако на современных компьютерах снижение скорости выполнения при	

			интерпретации обычно не заметна.	
5	12	Знакомство с понятием искусственного интеллекта.	Просмотр видеолекции «Искусственный интеллект и машинное обучение» на платформе «Урок Цифры». Работа на онлайн-тренажёре платформы.	Сформировать представление об искусственном интеллекте и машинном обучении, а также связи с программирование.
6	5	Рефлексия	Обсуждение полученных знаний, ответ на возникшие вопросы.	
Домашнее задание: Используя интерактивный тест, созданный с помощью сервиса LearningApps, закрепить полученные знания.				

Сценарий занятия 2.

Тема: Знакомство с языком программирования Python. Ввод. Вывод. Оператор присваивания. Математические операции.

В рамках реализации смешанного обучения планируется использовать модели

Перевёрнутый класс	Дома дети работают в онлайн-режиме (обрабатывают теоретический материал: смотрят видеолекции, изучают статьи, а в классе проводятся групповые занятия, практические работы, обсуждаются возникшие вопросы)			
Смена рабочих зон	В пространстве класса выделяются рабочие зоны: зона групповой работы, зона работы с учителем, зона онлайн-работы. Учащиеся делятся на группы и по кругу через определённый промежуток времени переходят от одной зоны к другой.			
Этапы				
Подготовьте мне	Расскажите мне	Покажите мне	Позвольте мне	Помогите мне
Ресурсы	Фоксфорд, YouTube, среда разработки Wing IDE 101, Wizer.me			
	Кабинет информатики: интерактивный многофункциональный комплекс, 15 ноутбуков обучающихся			

Важно: при использовании смешанного обучения основной упор делается на формирование навыков самостоятельной работы, групповой работы, взаимопомощи и коммуникативных компетенций.

Класс: 10-11

Цель занятия: знакомство обучающихся с языком программирования Python и его особенностями; знакомство с написанием программы на языке Python и созданием скриптов.

Задачи занятия:

Образовательные: познакомить обучающихся с языком программирования, с особенностями языка; сформировать первичные знания по применению изученного материала.

Воспитательные: развивать информационную культуру обучающихся; способность к самостоятельной и коллективной деятельности.

Развивающие: совершенствование умения анализировать, сравнивать, систематизировать и обобщать, развитие коммуникативных умений обучающихся.

Планируемые результаты:

Предметные: владение информацией о языке программирования Python, представление об особенностях языка; владение понятиями «Python».

Личностные: сформированность навыков сотрудничества со сверстниками; готовность и способность к образованию, в том числе самообразованию.

Метапредметные: умение контролировать и корректировать учебную деятельность.

Модель обучения: перевёрнутый класс.

Этапы урока:

7. Предварительная работа.
8. Систематизация и обобщение знаний.
9. Усвоение нового материала.

10. Использование языка программирования Python при разработке искусственного интеллекта.
11. Рефлексия.

Ход занятия:

№	Время (мин)	Вид работы	Описание	Цель
1	30	Предварительная работа	Использование онлайн ресурса Фоксфорд и YouTube обучающиеся самостоятельно знакомятся с языком программирования Python, применением языка. При помощи сервиса Wizer.me создаётся интерактивный рабочий лист для закрепления теоретического материала	Подготовить обучающихся к освоению учебного материала, формирование необходимых компетенций.
2	25	Систематизация и обобщение знаний.	Совместный разбор темы с использование интерактивного рабочего листа, созданного на сервисе Wizer.me *1.Как можно представить программу?Программу* можно представить как набор последовательных команд (алгоритм) для объекта (исполнителя), который должен их выполнить для достижения определенной цели. *2.Чем характеризуются языки программирования.* *Языки программирования* характеризуются синтаксической однозначностью (например, в них нельзя менять местами определенные слова) и ограниченностью (строго определенный набор слов и символов). *3.Этапы развития языков программирования.* Машинный язык; ассемблер;	Систематизировать изученный материал, формировать необходимые компетенции.

			рассвет языков высокого уровня; объектно-ориентированные языки программирования. *4.Что такое трансляторы?* *Трансляторы* — специальные программы, преобразующие программный код с языка программирования в машинный код. *5.Что происходит при компиляции?* При компиляции весь *исходный программный код* (тот, который пишет программист) сразу переводится в машинный. Создается так называемый отдельный *исполняемый файл*, который никак не связан с исходным кодом. *5.Что происходит при интерпретации?* При интерпретации выполнение кода происходит последовательно (можно сказать, строка за строкой). Операционная система взаимодействует с интерпретатором, а не исходным кодом.	
3	20	Усвоение нового материала	**История** Язык программирования Python был создан примерно в 1991 году голландцем Гвидо ван Россумом. Свое имя - Пайтон (или Питон) - получил от названия телесериала, а не пресмыкающегося. После того, как Россум разработал язык, он выложил его в Интернет, где уже целое сообщество программистов	Научить школьников основам работы в среде разработки Wing IDE 101. Познакомить со структурой простейшей программы Python.

			присоединилось к его улучшению. Python активно совершенствуется и в настоящее время. Часто выходят его новые версии. Официальный сайт http://python.org. **Особенности** Python – это интерпретируемый язык программирования: исходный код частями преобразуется в машинный в процессе выполнения специальной программой — интерпретатором. Python характеризуется ясным синтаксисом. Читать код на этом языке программирования достаточно легко, т.к. в нем мало вспомогательных элементов, а правила языка заставляют программистов делать отступы. Понятно, что хорошо оформленный текст с малым количеством отвлекающих элементов читать и понимать легче. Python – это полноценный, можно сказать универсальный, язык программирования. Он поддерживает объектно-ориентированное программирование (на самом деле он и разрабатывался как объектно-ориентированный язык). Также Python распространяется свободно на основании лицензии	

			подобной GNU General Public License. **Как писать программы** **Интерактивный режим** В основном интерпретатор выполняет команды построчно: пишешь строку, нажимаешь Enter, интерпретатор выполняет ее, наблюдаешь результат. Это очень удобно, когда человек только изучает программирование или тестирует какую-нибудь небольшую часть кода. Ведь если работать на компилируемом языке, то пришлось бы сначала написать код на исходном языке программирования, затем скомпилировать и уж потом запустить исполняемый файл на выполнение. Работать в интерактивном режиме можно в консоли. Для этого следует выполнить команду python. Запустится интерпретатор, где сначала выведется информация об интерпретаторе. Далее, последует приглашение к вводу (>>>). Запустите интерпретатор Питона. Поскольку никаких команд мы пока не знаем, то будем использовать Питон как калькулятор (возможности языка это позволяют). 2 + 5 3 * (5 - 8) 2.4 + 3.0 / 2 и т.д.	

			Наберите подобные примеры в интерактивном режиме (в конце каждого нажимайте Enter). Ответ выдается сразу после нажатия Enter (завершения ввода команды). Бывает, что в процессе ввода была допущена ошибка или требуется повторить ранее используемую команду. Чтобы не писать строку сначала, в консоли можно прокручивать список команд, используя для этого стрелки на клавиатуре. Другой вариант работы в интерактивном режиме — это работа в среде разработки IDLE, у которой есть интерактивный режим работы. В отличие от консольного варианта здесь можно наблюдать подсветку синтаксиса (в зависимости от значения синтаксической единицы она выделяется определенным цветом). Прокручивать список ранее введенных команд можно с помощью комбинаций Alt+N, Alt+P. Запустите IDLE. Попробуйте решать математические примеры здесь. **Создание скриптов** Несмотря на удобства интерактивного режима работы при написании программ на Питоне, обычно требуется сохранять исходный программный код для последующего использования. В таком	

			случае подготавливаются файлы, которые передаются затем интерпретатору на исполнение. По отношению к интерпретируемым языкам программирования часто исходный код называют скриптом. Файлы с кодом на Python обычно имеют расширение *py*. Подготовить скрипты можно в той же среде IDLE. Для этого, после запуска программы в меню следует выбрать команду **File ? New Window** (Crtl + N), откроется новое окно. Затем желательно сразу сохранить файл (не забываем про расширение *py*). После того как код будет подготовлен, снова сохраните файл (чтобы обновить сохранение). Ну и наконец, можно запустить скрипт, выполнив команду меню **Run ? Run Module** (F5). После этого в первом окне появится результат выполнения кода. (Примечание: если набирать код, не сохранив файл в начале, то подсветка синтаксиса будет отсутствовать.) Подготовьте скрипт (с примерами). Запустите его на выполнение. На самом деле скрипты можно готовить в любом текстовом редакторе (желательно, чтобы он поддерживал подсветку синтаксиса языка Python). Кроме того, существуют	

			специальные программы для разработки. Запускать подготовленные файлы можно не только в IDLE, но и в консоли с помощью команды python адрес/имя_файла. В консоли передайте интерпретатору Питона на выполнение подготовленный файл. Кроме того, существует возможность настроить выполнение скриптов с помощью двойного клика по файлу (в Windows данная возможность присутствует изначально).	
4	10	Использование языка программирования Python при разработке искусственного интеллекта.	Разъяснение связи языка программирования Python с искусственным интеллектом.	Сформировать представление о языке Python как языке разработки искусственного интеллекта.
5	5	Рефлексия	Обсуждение полученных знаний, ответ на возникшие вопросы.	

Сценарий занятия 3
Класс 10-11
Тема урока:
Программирование линейных алгоритмов на языке Python
Цель: Обеспечить усвоение обучающимися способов записи линейных алгоритмов на языке программирования Python

Задачи:		**Планируемые результаты:**	
Образовательные:	• Сформировать первоначальные представления о вводе и выводе данных. • научиться использовать полученные данные при составлении программ по математике;	**Личностные:**	• развитие критического логического мышления учащихся – умения выделять главное, существенное, обобщать имеющуюся информацию; • повышение мотивации к самостоятельной работе; • воспитание дисциплинированности.
Коммуникативные:	• предоставление ученикам возможности конструктивно и вежливо общаться, выражать собственное аргументированное мнение и прислушиваться к мнению других; • создание на уроке	**Предметные:**	• актуализация знаний учащихся по теме «Алгоритмы, формы представления алгоритмов»; • знакомство с основными операторами линейных алгоритмов на языке

	комфортной для межличностного общения обстановки, партнерских отношений «учитель-ученик».		программирования; • обучение составлять и применять программы, используя операторы ввода, вывода и присваивания;
Регулятивные:	• создание условий для формирования адекватной самооценки, понимания собственного уровня освоения материала, умения планировать учебную задачу и поэтапно ее решать.	**Метапредметные:**	• развитие речи учеников в области предметного словаря; • формирование умения связывать уже известный материал с новым, сравнивать и анализировать; выявление значимости владения английским языком для изучения информатики
Межпредметные связи:	Математика	**Ресурсы:**	
Формы деятельности:	Фронтальная, индивидуальная		ПК учителя; мультимедийный проектор, учебник, презентация по теме, карточки с заданием.
Тип урока:	Комбинированный		

		Личностные УУД	**Познавательные УУД**	**Коммуникативные УУД**	**Регулятивные УУД**
Организационный момент					

Приветствие учителя, проверка готовности к уроку. - Какое у вас настроение? Если хорошее - улыбнитесь всем! Если нет - посмотрите друг на друга и улыбнитесь! Начнем урок!	Приветствуют учителя, проверяют готовность к уроку, объявляют об отсутствующих	Проявляют положительное отношение к урокам информатики		Умение организовывать и планировать учебное сотрудничество с учителем.	Умение настраиваться на урок.
Актуализация знаний					
Тема, которую вы изучаете - «Алгоритмизация». Вы знаете, что алгоритмические конструкции бывают 3-х основных видов. Как называется конструкция с повторением некоторых операций? (цикл) с проверкой условия? (ветвление), а какой же алгоритм представлен в эпиграфе нашего урока?(линейный) Но сами по себе алгоритмы мы уже прошли и сейчас перешли непосредственно к программированию, а это означает запись алгоритма На языке программирования. Таким образом у нас есть два понятия – В – ЛИНЕЙНЫЙ АЛГОРИТМ и С -	Отвечают на вопросы	Оценивание усваиваемого содержания, понимает свои сильные и слабые стороны	Знакомится с работой интерактивной доски. Может структурировать информацию в нужной форме	Инициативное сотрудничество в поиске и сборе информации. Умение осознанно строить речевое высказывание	Может внести необходимые дополнения и коррективы

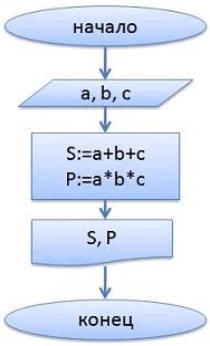
ПРОГРАММИРОВАНИЕ. Внимание вопрос. Что получится если В сложить с С? (Программирование линейного алгоритма.) Молодцы, это и есть тема нашего урока. Запишите её себе в тетрадь. Сегодня на уроке мы поговорим об очень важном разделе информатики - «Программирование линейных алгоритмов».					
Постановка цели и задачи урока. Мотивация учебной деятельности учащихся					
Сегодня на уроке мы познакомимся с операторами ввода и вывода информации, научимся составлять линейные программы. Предлагает сформулировать цель урока.	Формулируют цель урока.	Проявляют широкий интерес к новому учебному материалу	Формулирование познавательной цели самостоятельно	Умение отстаивать свою точку зрения, аргументируя её, подтверждать аргументы фактами	Обнаруживают и формулируют учебную проблему совместно с учителем
Первичное усвоение новых знаний					
Для вывода информации на экран компьютера используется оператор **print()** - выводит на экран, указанные в скобках параметры, после вывода курсор переходит в начало	Слушают объяснения учителя, записывают в тетрадь основные	Выделение существенной информации	Умение создавать структуры взаимосвязей смысловых единиц текста (выбор и организация	Умение задавать вопросы	Умение слушать с целевой установкой.

следующей строки экрана. В качестве параметров в круглых скобках может быть указан текст сообщения (текст записывается в апострофах ' ') и имя переменной (записывается без апострофов), значение которой нужно вывести. Между выводимыми элементами ставится запятая. Примеры: **print('Привет!');** На экране появится: Привет! **a=8;** **print('Я учусь в ', a,' классе');** На экране появится: Я учусь в 8 классе **x=10;y=15;** **print(x+y,' рублей');** На экране появится: 25 рублей. Команда ввода (считывания) с клавиатуры значения переменных во время работы программы: **input ( );** - считывает значение, введенные с клавиатуры. В качестве параметров в круглых скобках Перед каждой командой ввода рекомендуется выводить на экран	понятия по теме		элементов информации)		

поясняющий текст с информацией о том, что именно нужно ввести, либо этот тексти можно вводить в качестве аргемента функции. **A = input("A=")** На экране появится **a=** и после знака равно будет мигать курсор в ожидании ввода с клавиатуры значения переменной A. После ввода числа необходимо нажать Enter. **ВАЖНО:** Результат функции input(), строка символов, для того чтобы ввести число необходимо явное преобразование типов. Вести целое число: a = int(input()) Вести вещественное число: a = float (input()) Ввести логическое значение: a = bool(input())					
Первичная проверка понимания					
Приоритет выполнения операций в Python такой же, как и в математике. Достаточно только научиться правильно, записывать математические	Выполняют задания у доски	Умение адекватно судить о причинах своего успеха/неуспеха в учении,		Умение оформлять свои мысли в устной и письменной речи с учётом своих	Умение вносить необходимые дополнения и коррективы

выражения на языке Python. Десятичная запятая в обозначается точкой. **1)**Записать по правилам языка Python следующие выражения: *Дети поочередно работают у доски (остальные учащиеся контролируют и проверяют).* 37(25+87,5)-17(4,6+1,9) *37*(25+87.5)-17*(4.6+1.9)* $\frac{a+2b-3c}{5a+4}$ *(a+2*b-3*c)/(5*a+4)* **2)***Коллективное выполнение заданий: (на местах)* ***а) Что выведется на экран?*** *a=12; b=7;* *print(' Разность ', a ,' и ', b ,' равна ', a-b);* ***в) Найти и исправить все ошибки, допущенные в этой программе. Запишите программу без ошибок.*** *print('ввести числа')* *a=input()* *P=a*b* *print('площадь равна')*	Дети составляют программу по образцу прошлой задачи	связывая успехи с усилиями, трудолюбием.	Умение выбирать смысловые единицы текста и устанавливать отношения между ними. Смысловое чтение.	учебных и жизненных речевых ситуаций	

3.*Учитель показывает правила составления программы:* Составим программу вычисляющую s = a / b. *print('введите число a ')* *a = int(input())* *print('введите число b ')* *b = int(input())* *s=a/b;* *print('a/b =',s);* ***4.***Составьте программу, вычисляющую p=a*b . **5.** Дети в тетрадях составляют блок-схему. Составить блок - схему к программе. *print('введите число a ');* *a = int(input())* *print('введите число b ')* *b = int(input())* *p=a*b* *print('a*b =',p)* (После выполнения задания, на экран выводится правильная блок-схема, учащиеся проверяют правильность выполнения задания)					

Первичное закрепление					
Работа по разно уровневым карточкам. **1 вариант.** Написать программу, выводящую информацию: Скоро Новый год! **2 вариант.** Составить программу по алгоритму:  **Дополнительное задание.** Написать программу нахождения стоимости покупки. Значения цены и количества вводятся с клавиатуры. Добавьте комментарии к программе.	Выполняют задания по вариантам	Оценивание усваиваемого содержания, понимает свои сильные и слабые стороны.	Выбор наиболее эффективных способов решения задач в зависимости от конкретных условий	Умение задавать вопросы	Понимание причины своего неуспеха и нахождение выхода из этой ситуации
Информация о домашнем задании, инструктаж по его выполнению					
Формулирует домашнее задание и объясняет его выполнение Составить программу нахождения площади и периметра шестиугольника.	Записывают домашнее задание				Умение адекватно оценивать трудность
Рефлексия (подведение итогов занятия)					
Организует фиксирование нового	Анализируют		Передают содержание в	Высказывают свою	

содержания, рефлексию, самооценку учебной деятельности.	соответствие результатов требованиям конкретной учебной задачи		сжатом , выборочном или развёрнутом виде	точку зрения и пытаются её обосновать	

Сценарий занятия 4.

Тема: Условный оператор IF

Цель занятия: организовать учебную деятельность обучающихся по формированию понятия условного оператора IF в языке программирования Python, умения реализовывать алгоритмическую конструкцию «Ветвление» на языке программирования Python

Задачи:

Образовательные: познакомить учащихся с условным оператором и научить применять его при составлении программ на языке программирования Python.

Развивающие: развитие мыслительной деятельности, речи, алгоритмического стиля мышления.

Воспитательные: воспитание эмоционально-положительной направленности на практическую деятельность, интереса к информатике, личной ответственности за результаты своей работы.

Тип занятия: открытие нового знания знаний.

Вид занятия: урок-презентация, урок-лабораторное занятие.

Ресурсы: компьютеры, проектор, среда программирования Python IDLE, презентация.

Межпредметные связи: математика.

Основные понятия: условный оператор.

Ход урока (дидактическая структура урока)

1. Мотивационный этап(2 мин)	УУД: - Личностные - Коммуникативные
Приветствие	

2. Проверка и разбор домашнего задания (10 мин)	УУД: - Личностные - Познавательные - Регулятивные - Знаково-символические - Коммуникативные
Заданное на прошлом уроке домашнее задание: Установить на компьютер среду программирования IDLE. (Скачивание из интернета по инструкции или сохранение установочного файла на съёмный носитель). Написать программы: 1) Ввести основания и высоту трапеции и вывести площадь трапеции. 2) Получить случайное трехзначное число, вывести это число и сумму его отдельных цифр. 3) Программа, которая рассчитывает возраст человека в часах.	

3. Теоретическая часть (10 мин)	УУД: - Личностные - Познавательные

	- Знаково-символические - Коммуникативные

Откройте тетради и запишите тему урока: «Условный оператор if».
На прошлом уроке мы научились составлять линейные программы на языке Python.
Сегодня мы изучим конструкцию «ветвление» или «условный оператор if».
Если перевести на русский язык, конструкция условного оператора означает следующее:
Если <выполняется условие> делать: какие-то действия.
Например:

```
if a>b:
   print(a)
```

«Если a больше b, то вывести a».
Или:

```
if x==y:
   z=x+y
   z=z*z
```

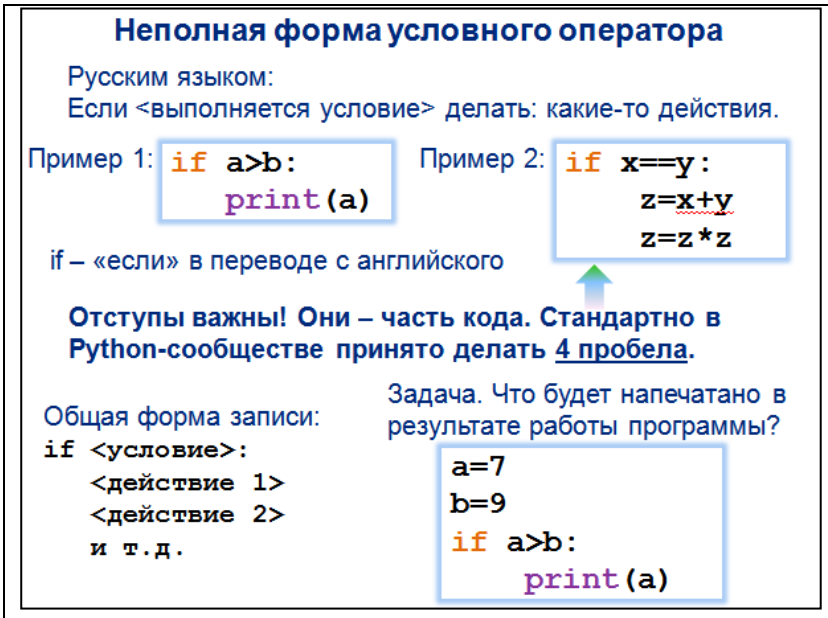
«Если x равен y, то z присвоить значение x+y, и возвести z в квадрат».
Отступы важны! Они – часть кода. Действия будут выполнятся только в том случае, если все они записаны с отступами, и причём с одинаковым количеством отступов.
Стандартно в Python-сообществе принято делать 4 отступа.
Общая форма записи неполной формы условного оператора:

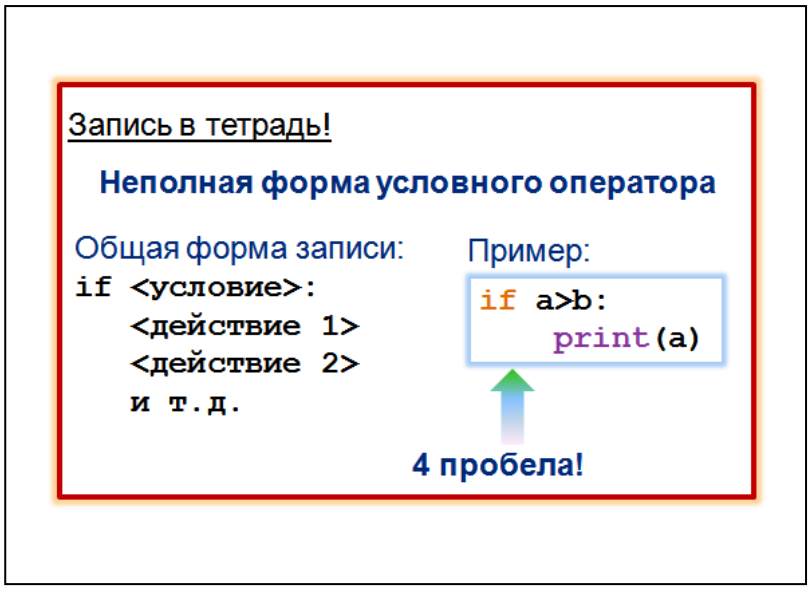
```
if <условие>:
    <действие 1>
    <действие 2>
    и т.д.
```

Задача. Что будет напечатано в результате работы программы?

```
a=7
b=9
if a>b:
   print(a)
```

(Ответ: ничего)





Это была неполная форма условного оператора. Но у условного оператора также есть и полная форма. Русским языком она звучит так:

Если <выполняется условие>: делать какие-то действия. Иначе: делать другие действия.

Иначе означает «если условие не выполняется».

Например:

```
if a>b:
   print(a)
else:
   print(b)
```

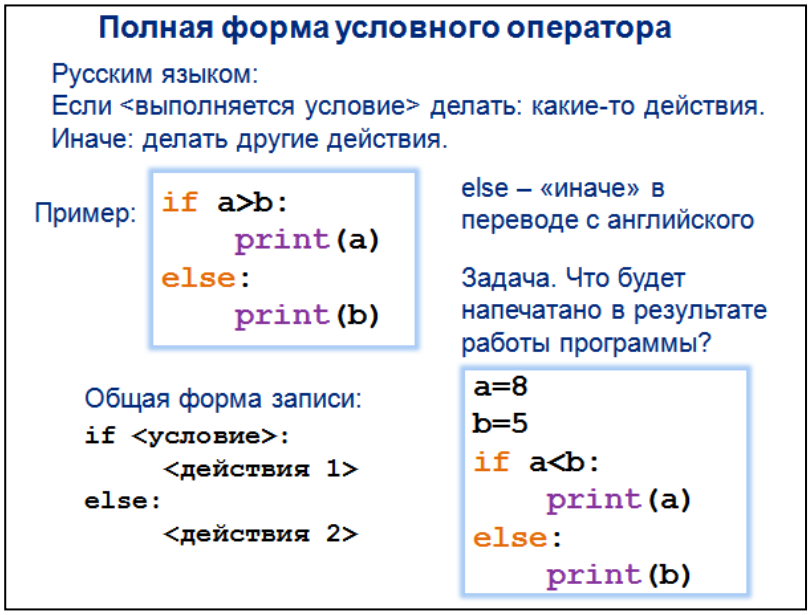
«Если a больше b, то вывести a, иначе вывести b.

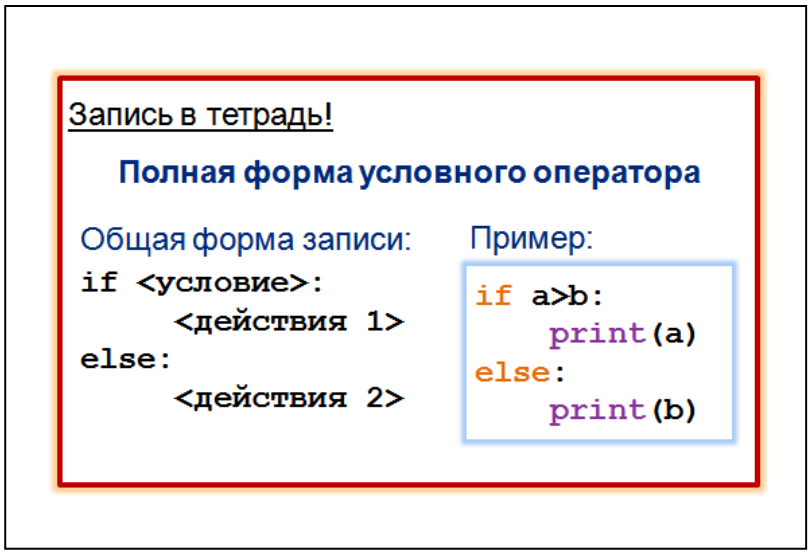
Общая форма записи неполной формы условного оператора:

```
if <условие>:
    <действия 1>
else:
    <действия 2>
```

Задача. Что будет напечатано в результате работы программы?

```
a=8
b=5
if a<b:
   print(a)
else:
   print(b)
```





Часто встречаются задачи с большим количеством условий и действий, которые нужно произвести при выполнении этих условий. Конструкции if-else не хватает, и тогда на помощь приходит оператор elif. Русским языком он объясняется так:
Если <выполняется условие 1>: делать такие-то действия.
Иначе если <выполняется условие 2>: делать другие действия.
Иначе если <выполняется условие 3>: делать третьи действия.
Иначе: делать что-то ещё.
Последнее «иначе» означает «если никакие из вышеперечисленных действий не выполняются». Присутствие «иначе» не обязательно. Например:

```
cost = 1500
```

```
if cost < 1000:
   print ( "Скидок нет" )
elif cost < 2000:
   print ( "Скидка 2%" )
elif cost < 5000:
   print ( "Скидка 5%" )
else:
   print ( "Скидка 10%" )
```

Что будет напечатано в результате работы программы? (Ответ: Скидка 2%)

Оператор elif

Русским языком:

Если <выполняется условие 1>: делать такие-то действия.
Иначе если <выполняется условие 2>: делать другие действия.
Иначе если <выполняется условие 3>: делать третьи действия.
Иначе: делать что-то ещё.

Пример:

```
cost = 1500
if cost < 1000:
  print ( "Скидок нет." )
elif cost < 2000:
  print ( "Скидка 2%." )
elif cost < 5000:
  print ( "Скидка 5%." )
else:
  print ( "Скидка 10%." )
```

Что будет напечатано?

Общая форма записи:

```
if <условие>:
   <действия 1>
elif <условие>:
   <действия 2>
elif <условие>:
   <действия 3>
…
else:
   <действия n>
```

Запись в тетрадь!

Оператор elif

Общая форма записи:

```
if <условие>:
   <действия 1>
elif <условие>:
   <действия 2>
elif <условие>:
   <действия 3>
…
else:
   <действия n>
```

Пример:

```
cost = 1500
if cost < 1000:
  print("Скидок нет.")
elif cost < 2000:
  print("Скидка 2%.")
elif cost < 5000:
  print "Скидка 5%.")
else:
  print("Скидка 10%.")
```

Знаки отношений:

> больше

< меньше

== равно

>= больше или равно

<= меньше или равно

!= не равно

Запись в тетрадь!

Знаки отношений:

> больше
< меньше
== равно
>= больше или равно
<= меньше или равно
!= не равно

Сложные условия.

Чтобы составить сложное условие используются операторы:

and - «и»

or - «или»

not - «не»

Например:

```
if a>0 and a<10 or a==100:
    print(a)
```

Будет ли напечатано а, если а равно 7? А если а равно 20?

Приоритет :

1) отношения (<, >, <=, >=, ==, !=)
2) not («НЕ»)
3) and («И»)
4) or («ИЛИ»)

Сложные условия

Чтобы составить сложное условие используются операторы:

and - «и»
or - «или»
not - «не»

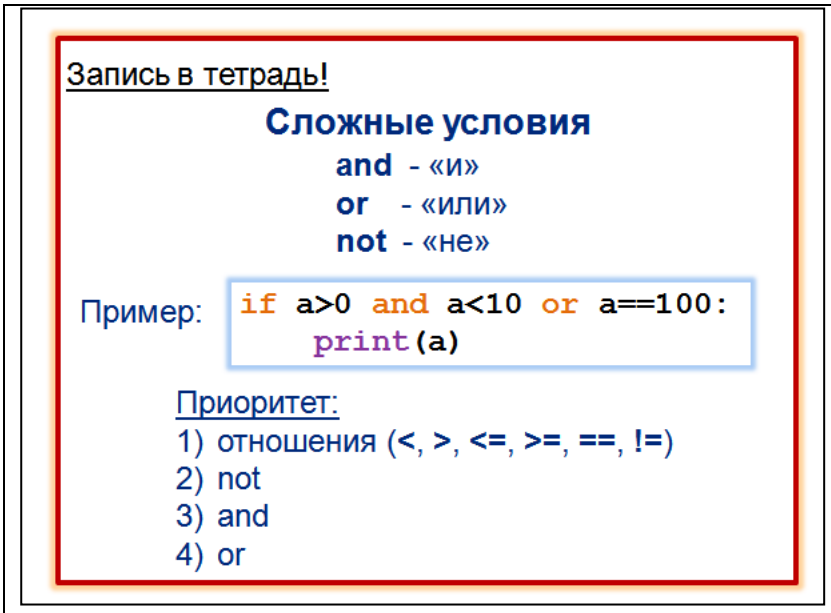
Пример:

```
if a>0 and a<10 or a==100:
    print(a)
```

Будет ли напечатано а, если а=7? А если а=20?

Приоритет:

1) отношения (<, >, <=, >=, ==, !=)
2) not
3) and
4) or



4. Работа на компьютерах (20 мин)	УУД: - Личностные - Регулятивные - Познавательные - Знаково-символические - Коммуникативные

Учащиеся пишут программы на компьютерах под руководством учителя.
Задачи:

1) Ввести целое число. Если это число больше 5, то вывести сообщение: «Это число больше пяти».
2) Ввести целое число. Если оно является положительным, то прибавить к нему 1; в противном случае вычесть из него 2. Вывести полученное число.
3) Проверить, принадлежит ли число, введенное с клавиатуры, интервалу (-9;2).
4) Написать программу "Предсказатель". Программа должна просить пользователя ввести вопрос, на который можно ответить однозначно, то есть "да" или "нет". После чего пользователю случайным образом выдаётся ответ, например: "Да", "Нет", "Определённо да!", "Ни в коем случае!", "Конечно же нет! И хватит задавать глупые вопросы!" и тому подобные. Вариантов ответов должно быть не меньше четырёх.
5) Ввести число a. Определить и вывести сообщение о том, чётное оно или нечётное. Для определения чётности числа используйте остаток от деления на 2: если a%2==0, то a – чётное.
6) Определить, является ли треугольник со сторонами a, b, c равнобедренным.
7) По номеру дня недели вывести его название.
8) Даны целочисленные координаты точки на плоскости. Если точка совпадает с началом координат, то вывести 0. Если точка не совпадает с началом координат, но лежит на оси OX или OY, то вывести соответственно 1 или 2. Если точка не лежит на координатных осях, то вывести 3.

Выставление оценок за работу на уроке.

5. Домашнее задание (3 мин)	УУД: - Личностные - Коммуникативные
Домашнее задание: Написать программы: 1) Дано целое число. Если оно является положительным, то умножить его на 3; в противном случае вычесть из него 100. Вывести полученное число. 2) Определить, является ли число a делителем числа b. 3) Определить возможность существования треугольника по сторонам. (Треугольник существует только тогда, когда сумма любых двух его сторон больше третьей).	

Сценарий занятия 5.
Класс: 10-11
Тема урока: Цикл с предусловием WHILE.
Цель урока: организовать учебную деятельность обучающихся по формированию понятия цикла с предусловием в языке программирования Python, умения реализовывать циклическую алгоритмическую конструкция на языке программирования Python
Задачи:
образовательные: познакомить учащихся с циклом с предусловием и научить применять его при составлении программ на языке программирования Python.
развивающие: развитие мыслительной деятельности, речи, алгоритмического стиля мышления.
воспитательные: воспитание эмоционально-положительной направленности на практическую деятельность, интереса к информатике, личной ответственности за результаты своей работы.
Тип урока: открытие нового знания.
Вид урока: урок-презентация, урок-лабораторное занятие.
Ресурсы: компьютеры, проектор, среда программирования Python IDLE, презентация.
Межпредметные связи: математика.
Основные понятия: цикл с предусловием.
Ход урока (дидактическая структура урока)

1. Организационный этап (2 мин)	УУД: - Личностные - Коммуникативные
Приветствие, перекличка.	

2. Проверка и разбор домашнего задания (10 мин)	УУД: - Личностные - Познавательные - Регулятивные - Знаково-символические - Коммуникативные
Заданное на прошлом уроке домашнее задание: Написать программы: 9) Дано целое число. Если оно является положительным, то умножить его на 3; в противном случае вычесть из него 100. Вывести полученное число. 10) Определить, является ли число a делителем числа b. 11) Определить возможность существования треугольника по сторонам. (Треугольник существует только тогда, когда сумма любых двух его сторон больше третьей).	

3. Теоретическая часть (10 мин)	УУД: - Личностные

	- Познавательные - Знаково-символические - Коммуникативные

Откройте тетради и запишите тему урока: «Цикл WHILE (цикл с предусловием)».
На прошлом уроке мы изучили конструкцию ветвления или «условный оператор if».
Сегодня мы изучим конструкцию «цикл WHILE» или «цикл с предусловием» и попробуем написать первые игры.
WHILE – «пока» в переводе с английского
Посмотрим на пример цикла WHILE:

```
n=0
while n<3:
    n=n+1
```

Если объяснять русским языком, цикл WHILE означает следующее:
Пока <выполняется условие>: делать какие-то действия.
«Пока n меньше 3, прибавлять к n единицу»
Цикл повторяется, пока условие истинно, если же нет, цикл заканчивается.
Скажите, чему будет равно n после завершения цикла? (Ответ: 3)
Для того, чтобы увидеть, что происходит в теле цикла, создадим и запустим следующую программу:

```
n=0
while n<5:
    n=n+1
    print(n)
```

Программа выведет:
1
2
3
4
5
На первом шаге цикла n=1, на втором n=2 и так далее. Когда n станет равным пяти и программа выведет число 5, снова будет проверяться условие. Но условие не будет выполняться, так как 5<5 - не верно. И произойдёт выход из цикла.

WHILE – «пока» в переводе с английского

Русским языком:
Пока <выполняется условие>: делать какие-то действия.

Пример:

```
n=0
while n<3:
    n=n+1
```

Чему будет равно n после завершения цикла?

4 пробела!

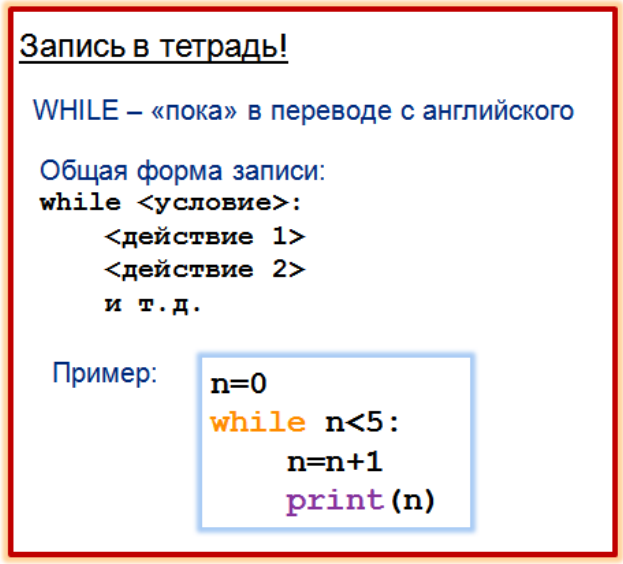
Запустить программу на компьютере:

```
n=0
while n<5:
    n=n+1
    print(n)
```

Запись в тетрадь:

while – «пока» в переводе с английского
Общая форма записи:
while <условие>:
<действие 1>
<действие 2>
и т.д.
Пример:

```
n=0
while n<5:
    n=n+1
    print(n)
```



Бесконечный цикл.

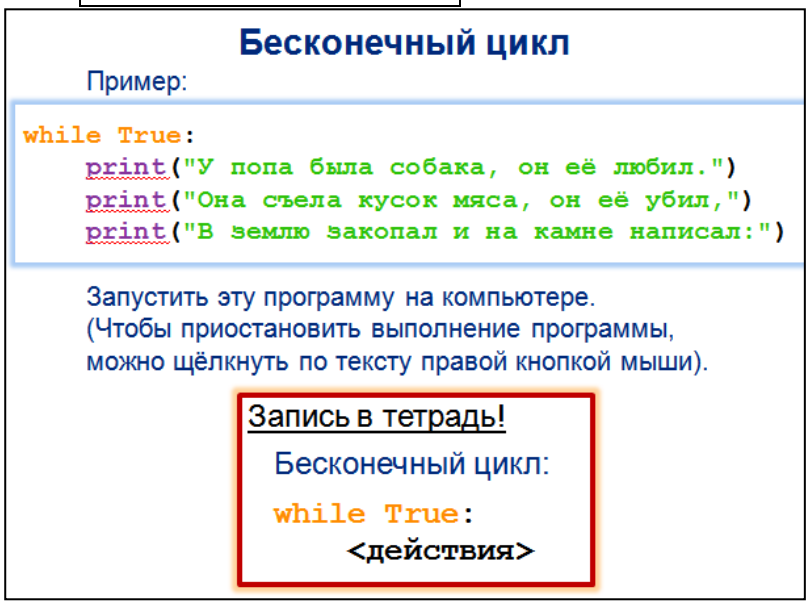
```
while True:
    print("У попа была собака, он её любил.")
    print("Она съела кусок мяса, он её убил,")
    print("В землю закопал и на камне написал:")
```

Пишем и запускаем эту программу на компьютере, чтобы увидеть бесконечный цикл в действии.

Запись в тетрадь:

Бесконечный цикл:

```
while True:
    <действия>
```



Досрочный выход из цикла

break

Пример:

```
import random
while True:
   a=random.randint(1,10)
   print(a)
   if a==7:
      break
```

Пишем и запускаем эту программу на компьютере, чтобы увидеть оператор break в действии.

Что происходит в программе?

Досрочный выход из цикла

break

Пример:

```
import random
while True:
    a=random.randint(1,10)
    print(a)
    if a==7:
        break
```

Запустить программу на компьютере.

4. Работа на компьютерах (20 мин)	УУД: - Личностные - Регулятивные - Познавательные - Знаково-символические - Коммуникативные

Учащиеся пишут программы на компьютерах под руководством учителя.

Задачи:

1) С помощью цикла while вывести любую фразу 7 раз.
2) Написать программу, которая получает два целых числа A и B (0 < A < B) и выводит все натуральные числа в интервале от A до B.
3) С помощью бесконечного цикла реализовать игру «Купи слона». Программа должна выводить фразу «Купи слона!», получать ответ пользователя и выводить: «Все говорят “фраза, введённая пользователем”. А ты купи слона!» Затем программа получает новый ответ от пользователя и так до бесконечности.
4) С помощью бесконечного цикла и оператора break реализовать игру «Угадай число». Программа генерирует случайное число в диапазоне от 1 до 7 и говорит пользователю: «Угадай число от 1 до 7!». В бесконечном цикле программа считывает ответы пользователя и подсказывает ему «больше!» или

«меньше!», а в случае правильного ответа цикл завершается, и выводятся сердечные поздравления с победой.

5) Доработать программу «Угадай число» так, чтобы она не заканчивалась, то есть, чтобы после угадывания одного числа, начиналось угадывание следующего. Увеличить диапазон с 7 до 15 и ввести ограничение на количество попыток (например, 3 попытки). Если пользователь не справился за 3 попытки, вывести «GAME OVER».

Задачи:

1) С помощью цикла while вывести любую фразу 7 раз.
2) Написать программу, которая получает два целых числа A и B (0 < A < B) и выводит все натуральные числа в интервале от A до B.
3) С помощью бесконечного цикла реализовать игру «Купи слона». Программа должна выводить фразу «Купи слона!», получать ответ пользователя и выводить: «Все говорят "фраза, введённая пользователем". А ты купи слона!» Затем программа получает новый ответ от пользователя и так до бесконечности.

Задачи:

4) С помощью бесконечного цикла и оператора break реализовать игру «Угадай число». Программа генерирует случайное число в диапазоне от 1 до 7 и говорит пользователю: «Угадай число от 1 до 7!». В бесконечном цикле программа считывает ответы пользователя и подсказывает ему «больше!» или «меньше!», а в случае правильного ответа цикл завершается, и выводятся сердечные поздравления с победой.
5) Доработать программу «Угадай число» так, чтобы она не заканчивалась, то есть, чтобы после угадывания одного числа, начиналось угадывание следующего. Увеличить диапазон с 7 до 15 и ввести ограничение на количество попыток (например, 3 попытки). Если пользователь не справился за 3 попытки, вывести «GAME OVER».

Выставление оценок за работу на уроке.

Дополнительные материалы:

Программа «Угадай число» вариант-1:

```
import random
   a=random.randint(1,7)
   print("Угадай число от 1 до 7")
   while True:
      b=int(input())
      if b==a:
```

```
        print("Верно!")
        break
    elif b<a:
        print("Больше!")
    else:
        print("Меньше!")
```

Программа «Угадай число» вариант-2:

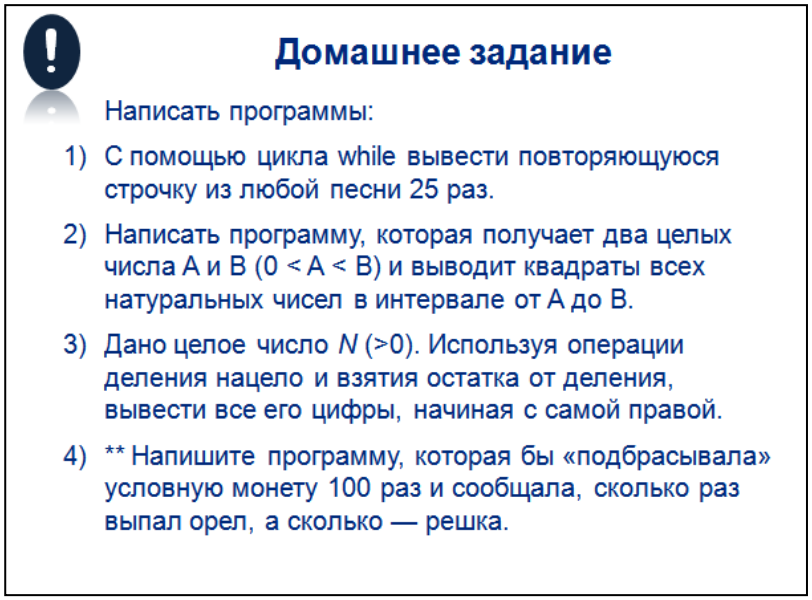
```
import random
while True:
  a=random.randint(1,15)
  print("Угадай число от 1 до 15")
  k=0
  while True:
    k=k+1
    if k>3:
      print("GAME OVER")
      break
    b=int(input())
    if b==a:
      print("Пусть другие говорят:")
      print("Главное – участие.")
      print("Лишь победа дарит нам")
      print("Наслажденье, счастье!")
      print("Вас с победой поздравляем!")
      print("От души мы вам желаем")
      print("Без намёков и прикрас")
      print("Побеждать ещё 100 раз!")
      break
    elif b<a:
      print("Больше!")
    else:
      print("Меньше!")
```

5. Домашнее задание (3 мин)	УУД: - Личностные - Коммуникативные

Написать программы:

1) С помощью цикла while вывести повторяющуюся строчку из любой песни 25 раз.
2) Написать программу, которая получает два целых числа A и B (0 < A < B) и выводит квадраты всех натуральных чисел в интервале от A до B.
3) Дано целое число *N* (>0). Используя операции деления нацело и взятия остатка от деления, вывести все его цифры, начиная с самой правой.

4) ** Напишите программу, которая бы «подбрасывала» условную монету 100 раз и сообщала, сколько раз выпал орел, а сколько — решка.



Сценарий занятия 6
Класс: 10-11
Тема урока: Цикл с параметром FOR.
Цель урока: организовать учебную деятельность обучающихся по формированию понятия цикла с параметром For в языке программирования Python, умения реализовывать циклическую алгоритмическую конструкция на языке программирования Python
Задачи:
обучающие: познакомить учащихся с циклом с параметром и научить применять его при составлении программ на языке программирования Python.
развивающие: развитие мыслительной деятельности, речи, алгоритмического стиля мышления.
воспитательные: воспитание эмоционально-положительной направленности на практическую деятельность, интереса к информатике, личной ответственности за результаты своей работы.
Тип урока: усвоение новых знаний.
Вид урока: урок-презентация, урок-лабораторное занятие.
Ресурсы: компьютеры, проектор, среда программирования Python IDLE, презентация.
Межпредметные связи: математика.
Основные понятия: цикл с параметром.
Ход урока (дидактическая структура урока)

1. Организационный этап (2 мин)	УУД: - Личностные - Коммуникативные
Приветствие, перекличка.	

2. Проверка и разбор домашнего задания (10 мин)	УУД: - Личностные - Познавательные - Регулятивные - Знаково-символические - Коммуникативные
Заданное на прошлом уроке домашнее задание: Написать программы: 5) С помощью цикла while вывести повторяющуюся строчку из любой песни 25 раз. 6) Написать программу, которая получает два целых числа A и B (0 < A < B) и выводит квадраты всех натуральных чисел в интервале от A до B. 7) Дано целое число *N* (>0). Используя операции деления нацело и взятия остатка от деления, вывести все его цифры, начиная с самой правой. 8) ** Напишите программу, которая бы «подбрасывала» условную монету 100 раз и сообщала, сколько раз выпал орел, а сколько — решка.	

3. Теоретическая часть (10 мин)	УУД: - Личностные - Познавательные - Знаково-символические - Коммуникативные

На прошлом уроке мы изучили цикл с предусловием WHILE. Сегодня мы изучим ещё один цикл – цикл с параметром FOR.

В отличие от цикла WHILE в цикле FOR можно указать точное количество повторений, нужное программисту. Посмотрим на пример цикла FOR:

```
for i in range(5):
    print("Привет!")
```

Напечатаем и запустим эту программу на своих компьютерах. В результате работы программы слово «Привет!» будет напечатано 5 раз. Мы указали диапазон – число «5». Range – диапазон в переводе с английского. При этом переменная i по мере выполнения цикла будет принимать значения 0, 1, 2, 3, 4. Можно это проверить:

```
for i in range(5):
    print(i)
```

В результате работы программы выведется столбец чисел:

0
1
2
3
4

Пример цикла FOR:

```
for i in range(5):
    print("Привет!")
```

Переменная i принимает значения 0, 1, 2, 3, 4:

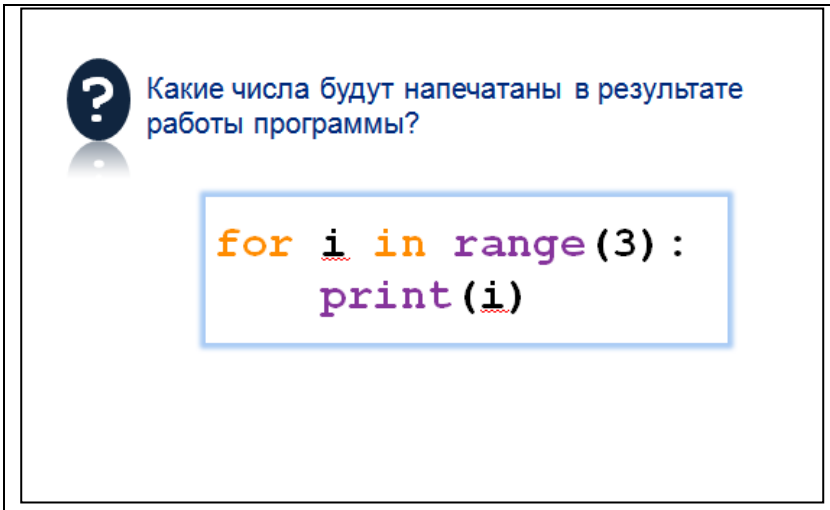
```
for i in range(5):
    print(i)
```

for – «для»
in range – «в диапазоне» в переводе с английского

Какие числа будут напечатаны в результате работы следующей программы?

```
for i in range(3):
    print(i)
```

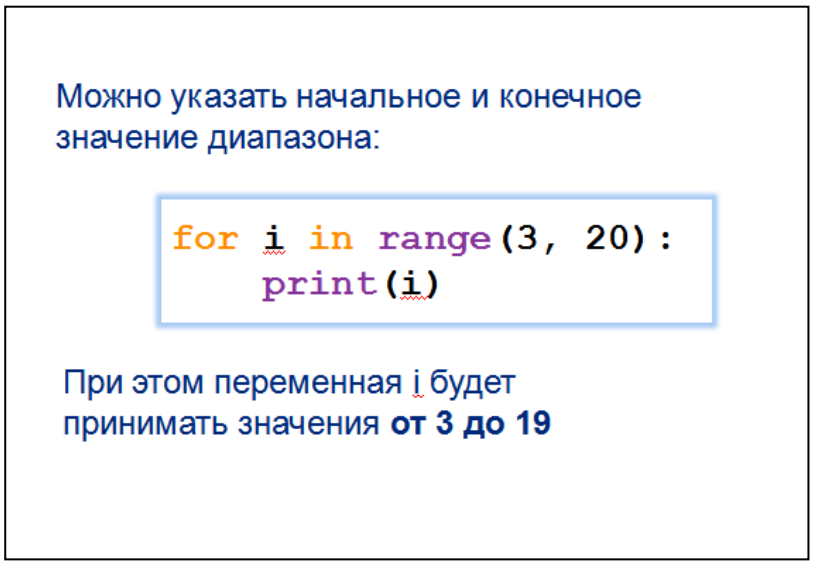
(Ответ: 0, 1, 2)



Можно указать начальное и конечное значение диапазона:

```
for i in range(3, 20):
    print(i)
```

При этом переменная i будет принимать значения от 3 до 19. Проверяем работу программы на компьютере.



Запись в тетрадь:

for – «для»
in range – «в диапазоне» в переводе с английского
Общая форма записи:

```
for <переменная-счётчик> in range(<диапазон>):
    <действия>
```

Пример 1:

```
for i in range(5):
    print("Привет!")
```

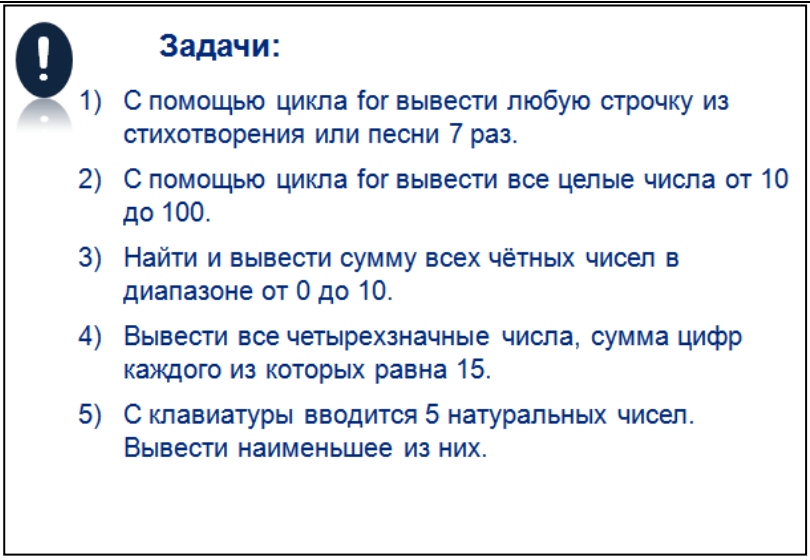
Пример 2 (с начальным и конечным значением):

```
for i in range(3, 20):
    print(i)
```

4. Работа на компьютерах (20 мин)	УУД: - Личностные - Регулятивные - Познавательные - Знаково-символические - Коммуникативные

Учащиеся пишут программы на компьютерах под руководством учителя.
Задачи:

1) С помощью цикла for вывести любую строчку из стихотворения или песни 7 раз.
2) С помощью цикла for вывести все целые числа от 10 до 100.
3) Найти и вывести сумму всех чётных чисел в диапазоне от 0 до 10.
4) Вывести все четырехзначные числа, сумма цифр каждого из которых равна 15.
5) С клавиатуры вводится 5 натуральных чисел. Вывести наименьшее из них.



Выставление оценок за работу на уроке.

5. Домашнее задание (3 мин)	УУД: - Личностные - Коммуникативные

Написать программы:

1) С помощью цикла for вывести все чётные числа от 10 до 20.
2) С помощью цикла for вывести 10 первых степеней двойки.
3) С клавиатуры вводится 7 натуральных чисел. Вывести наибольшее из них.