



**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ
ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
ДОМ ДЕТСКОГО ТВОРЧЕСТВА «НА 9-ОЙ ЛИНИИ»
ВАСИЛЕОСТРОВСКОГО РАЙОНА САНКТ-ПЕТЕРБУРГА**

Программа принята

на педагогическом совете

протокол № 3

от «30» мая 2025 г

Утверждена

Директор ГБУ ДО

ЛЛТ «На 9-ой линии»

Приказ № 74

от «16» июня 2025 г

_____ И.В.Петерсон

**ДОПОЛНИТЕЛЬНАЯ
ОБЩЕРАЗВИВАЮЩАЯ ПРОГРАММА
«Программирование. Подготовка к IT-профессии»**

Возраст обучающихся: 9 - 17 лет

Срок освоения: 2 года

Разработчик:
Лубченков Леонид Константинович,
педагог дополнительного образования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Программа имеет **техническую** направленность.

Адресат программы: школьники 9-17 лет, заинтересованные в программировании и информационных технологиях.

Актуальность программы:

Современное общество переживает активную стадию цифровой трансформации.

Современное общество невозможно представить без информационных технологий, которые играют ключевую роль в профессиональном и повседневном мире. Программирование, будучи основой ИТ, стало важнейшим навыком для специалистов, что обусловило высокий интерес к данному направлению среди молодежи. Однако школьного курса информатики часто оказывается недостаточно для освоения необходимых навыков, и потому важной педагогической задачей является предоставление возможности школьникам изучить программирование на глубоком уровне.

Все больше сфер жизни людей становятся зависимыми от информационных технологий и электроники. Одной из составляющих информационной компетентности является владение языком программирования, вследствие чего встает вопрос о выборе языка программирования, который отвечает современным требованиям к написанию программ, служит основой для дальнейшего развития и совершенствования навыков программирования и удобен в освоении подростками.

Программа «Программирование. Подготовка к ИТ-профессии» предназначена для учащихся школ возраста 9-17 лет и реализуется в двухгодичном формате, охватывая как базовые, так и продвинутые аспекты программирования и проектной деятельности. Программа включает два уровня освоения, которые дают ученикам базовые и продвинутые навыки программирования, адаптированные для их возрастной категории и уровня подготовки.

Уровень освоения – **базовый**

Объем и срок освоения: 2 года 1 год -144 часа, 2 год 144 часа - 288 часов

Цели и задачи программы

Цель: Развить у школьников навыки программирования, алгоритмического и проектного мышления, необходимые для успешного освоения ИТ-профессий в будущем и выполнения реальных задач в условиях быстро меняющихся технологий.

Задачи программы

Обучающие задачи:

- Дать базовые и углубленные знания в области алгоритмов и информатики.
- Обучить основам и принципам объектно-ориентированного программирования.
- Освоить программирование на Python и научить разрабатывать базовые веб-страницы и приложения.
- Познакомить с игровым движком Godot для создания интерактивных приложений.
- Научить планированию и реализации проектов с использованием профессиональных инструментов, включая базы данных и системы контроля версий.

Развивающие задачи:

- Развить алгоритмическое и проектное мышление.
- Углубить навыки командной работы и управления проектами.
- Развить навыки поиска информации, самообучения и критического мышления.
- Закрепить умение творчески подходить к решению технических задач.

Воспитательные задачи:

- Воспитывать трудолюбие, ответственность за результат и готовность к саморазвитию.

- Развить уверенность в собственных силах и способность достигать профессиональных целей.
- Формировать активную личностную позицию и умение работать в команде.

Язык реализации: государственный язык Российской Федерации (русский язык).

Форма обучения: очная.

Условия набора в коллектив: в группу принимаются все желающие, по заявлению родителей, без предварительного отбора. Распределение в группы осуществляется по итогам собеседования. Обучающиеся принимаются на курс для начинающих или на продвинутый уровень по итогам собеседования.

Количество обучающихся в группе: списочный состав обучающихся в группах формируется по норме наполняемости: не менее 10 человек.

Формы организации занятий: программой предусмотрены аудиторные занятия.

Занятия по программе проходят в разных **формах**:

- практическое занятие,
- защита проектов
- соревнований.
- ролевая игра;
- познавательная игра;
- творческое моделирование;
- викторина и др.

Форма организации деятельности детей:

- фронтальная (изучение теоретического материала с демонстрацией образцов, готовых конструкций, презентаций, объяснение нового материала),
- индивидуально-групповая (самостоятельная работа с раздаточным материалом, работа в малых группах, парах над созданием творческого проекта).

Планируемые результаты

1. Предметные результаты:

- Дать базовые и углубленные знания в области алгоритмов и информатики.
- Обучить основам и принципам объектно-ориентированного программирования.
- Освоить программирование на Python и научить разрабатывать базовые веб-страницы и приложения.
- Познакомить с игровым движком Godot для создания интерактивных приложений.
- Научить планированию и реализации проектов с использованием профессиональных инструментов, включая базы данных и системы контроля версий.

2. Личностные результаты:

- Реализация творческого потенциала;
- Удовлетворение потребности в самовыражении;
- Повышение культуры межличностного общения;
- Постановка личных и профессиональных целей;
- Развитое логическое и алгоритмическое мышление.

3. Метапредметные результаты:

- Умение находить достоверные источники информации;
- Умение быстро самообучаться;
- Навык планирования работы и самоорганизации;
- Умение работать в команде и сформированные лидерские качества.

- Умение программирования, планирования и командной работы.

Условия реализации: Программа рассчитана на 2 года и проводится очно и дистанционно с использованием интернет-ресурсов.

Отличительные особенности программы

- **Практическая направленность:** программа предусматривает большое количество практических занятий и проектных работ, что позволяет учащимся лучше усваивать материал.
- **Гибкость и личностно-ориентированный подход:** обучение проводится в двух типах групп (для начинающих и продолжающих), а сложность проектов и темп обучения подбираются индивидуально.
- **Поддержка профессионального роста:** освоение основ ООП, алгоритмов и работы с современными инструментами программирования формирует у школьников уверенность в своих силах и готовность к дальнейшему обучению в IT-специальностях.

Материально-техническое оснащение

Для реализации данной программы требуется компьютерный класс, оснащенный следующим оборудованием:

- компьютеры (рабочие станции), объединенные в локальную сеть и подключенные к ресурсам Интернет;
- Источник бесперебойного питания.
- Оборудование для подключения к ресурсам Интернет (выделенный канал подключения, модем).

Рабочие станции должны иметь следующую конфигурацию:

	Минимальная	Рекомендуемая
Процессор	Intel линейки Core семейства i3 или AMD линейки Ryzen семейства 3	Intel линейки Core семейства i5 или AMD линейки Ryzen семейства 5
Оперативная память	4Гб	8Гб
Жесткий диск	200Гб	320Гб
Видеокарта	256Мб встроенная	512Мб не встроенная
Сетевая карта	Пропускная способность 100Мбит	Пропускная способность 100Мбит
Монитор	Диагональ 17 дюймов	Диагональ 17 дюймов

Требуемое программное обеспечение:

Операционное

- MS Windows 8.1 / MS Windows 10 / MS Windows 11 / Linux Ubuntu

Прикладное

- Веб-браузер
- интерпретатор Python версии ≥ 3.10
- PyCharm IDE
- Git

Специальное

- Терминал
- Архиватор

Кадровое обеспечение программы: педагог дополнительного образования

УЧЕБНЫЙ ПЛАН 1-ОГО ГОДА ОБУЧЕНИЯ ГРУППЫ НАЧИНАЮЩИХ (144 Ч.)

№ п/ п	Наименование раздела, темы	Количество часов			Форма контроля	Самостоятельная работа с использованием дистанционных образовательных технологий
		Всего	Теория	Практика		
1	Введение в программирование	10	6	4	Проверка конспекта.	Курс логика на hexlet.io: https://ru.hexlet.io/courses/logic
2	Основы программирования на Python	58	27	31	Готовая программа Бесполезные факты. Готовая программа Отгадай число. Готовая программа Анаграммы. Тестирование.	Задачи с сайта: https://informatics.mccme.ru/
3	Продвинутые программы на Python	36	16	20	Готовая программа Крестики-нолики. Готовая программа Викторина. Тестирование.	Задачи с сайта: https://informatics.mccme.ru/
4	Разработка графических интерфейсов	40	17	23	Готовая программа Сумасшедший сказочник. Готовая программа Паника в пиццерии. Готовая программа Прерванный полет.	Курс по библиотеке Pygame: https://younglinux.info/pygame/ Курс Введение в Git: https://ru.hexlet.io/courses/intro_to_git
Всего		144	66	78		

УЧЕБНЫЙ ПЛАН 2-ОГО ГОДА ОБУЧЕНИЯ ГРУППЫ НАЧИНАЮЩИХ (144 Ч.)

№ п/ п	Наименование раздела, темы	Количество часов			Форма контроля	Самостоятельная работа с использованием дистанционных образовательных технологий
		Всего	Теория	Практика		
1	Объектно-ориентированное программирование	24	10	14	1) Выполнение задания «класс “Библиотека”» 2) Выполнение задания текстовая RPG-игра	Изучение материалов на сайте ddpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
2	Классические алгоритмы на Python	24	13	11	1) Выполнение задания «Сравнение алгоритмов» 2) Выполнение задания «Минимальный набор монет»	Изучение материалов на сайте ddpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
3	Разработка игр на движке Godot	48	20	28	1) Выполнение задания «Платформер» 2) Выполнение задания «Лабиринт»	Изучение материалов на сайте ddpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
4	Индивидуальные проекты	48	6	42	1) Составление технической документации 2) Итоговая презентация проекта	
Всего		144	49	95		

Утверждена

Директор ГБУ ДО
ЛЛТ «На 9-ой линии»

Приказ № 74
от «16» июня 2025 г
И.В.Петерсон

**КАЛЕНДАРНЫЙ ГРАФИК РЕАЛИЗАЦИИ
ДООП «ПРОГРАММИРОВАНИЕ. ПОДГОТОВКА К ИТ-ПРОФЕССИИ»
НА 2025-2026 УЧ. ГОД**

Год обучения	Дата начала обучения по программе	Дата окончания обучения по программе	Всего учебных недель	Всего учебных день	Количество учебных часов	Режим занятия
1-й год 1 группа	05.09.2025	23.05.2026	36	72	144	2 раза в неделю по 2 часа
2-й год 1 группа	05.09.2025	23.05.2026	36	72	144	2 раза в неделю по 2 часа

Рабочая программа
К ОБЩЕРАЗВИВАЮЩЕЙ ПРОГРАММЕ
«Программирование. Подготовка к IT-профессии»
год обучения
Задачи 1 года обучения

Обучающие задачи:

- Дать базовые и углубленные знания в области алгоритмов и информатики.
- Обучить основам и принципам объектно-ориентированного программирования.
- Освоить программирование на Python и научить разрабатывать базовые веб-страницы и приложения.

Развивающие задачи:

- Развить алгоритмическое и проектное мышление.
- Углубить навыки командной работы и управления проектами.
- Развить навыки поиска информации, самообучения и критического мышления.
- Закрепить умение творчески подходить к решению технических задач.

Воспитательные задачи:

- Воспитывать трудолюбие, ответственность за результат и готовность к саморазвитию.
- Развить уверенность в собственных силах и способность достигать профессиональных целей.
- Формировать активную личностную позицию и умение работать в команде.

СОДЕРЖАНИЕ ОБУЧЕНИЯ 1-ОГО ГОДА ГРУППЫ ДЛЯ НАЧИНАЮЩИХ

Программа первого года обучения направлена на освоение базовых принципов программирования, изучение языка Python и знакомство с разработкой графических интерфейсов. Учащиеся шаг за шагом изучат фундаментальные концепции программирования, научатся писать простые программы и создадут свои первые проекты.

1. Введение в программирование

Знакомство с основами алгоритмического мышления и ключевыми концепциями программирования, такими как логические операции, циклы и ветвления. Этот модуль позволяет заложить прочный фундамент для дальнейшего изучения программирования.

Теория: План курса. Что такое программирование. Программирование как работа. Сферы программирования. Понятие алгоритма. Что такое компьютер, его особенности. Характеристики и виды языков программирования. Краткая история развития языков программирования. Устройство компьютера. Виды программ. Сферы программирования. Виды языков программирования. Языки для изучения программирования. Языки для работы. Что такое машинный код. Что такое система счисления. Представление чисел в двоичной системе. Степени двойки. Перевод из десятичной системы в двоичную. Арифметический операции в двоичной системе. Что такое логика. Таблицы истинности. Простые операции И, ИЛИ, НЕ. Сложные операции исключающее ИЛИ, импликация. Основные правила преобразования логических выражений.

Практика: Вводное тестирование. Демонстрация кода на разных языках программирования. Самостоятельное изучение информации по понравившемуся направлению программирования и языку, который в нем используется. Перевод чисел из десятичной системы в двоичную. Перевод чисел из двоичной системы в десятичную. Сложение и вычитание двоичных чисел в столбик. Вывод таблиц истинности исключающего или и импликации. Вывод таблиц истинности логических выражений. Преобразование логических выражений.

Формы контроля:

Проверка конспекта.

2. Основы программирования на Python

Изучение базовых возможностей Python, включая переменные, типы данных, функции и структуры данных. Учащиеся научатся разрабатывать программы с использованием циклов, условий и обработки данных, что позволит им решить широкий спектр задач.

Теория: Что такое Python. Особенности Python. Инструменты для работы с Python. Что такое IDLE. Режимы программирования. Печать строк. Комментарии. Описание игры Game Over. Строки. Кавычки внутри строк. Вывод на экран нескольких значений. Задание завершающей строки при выводе. Тройные кавычки. Звук системного динамика. Escape-последовательности, сцепление строк. Символ продолжения строки. Повторение строк. Числовые типы данных. Целочисленное деление. Остаток от деления. Математические операторы. Что такое переменные и литералы. Создание переменных. Использование переменных. Имена переменных. Функция input. Применение функции input. Создание новых строк с помощью строковых методов. Обнаружение логических ошибок. Устранение логических ошибок. Преобразование строк в целые числа. Составные операторы присвоения. Знакомство с программой Рантье. Знакомство с программой Бесплезные факты. Импорт модуля random. Применение функции randint(). Применение функции randrange(). Как работает конструкция if. Создание условий Операторы сравнения. Создание блоков кода с помощью отступов. Создание собственных условных конструкций. Как работают условия else. Как работают условия elif. Как работает цикл while. Инициализация управляющей переменной. Проверка значения управляющей переменной. Изменение значения управляющей переменной. Борьба с бесконечными циклами. Трассировка программы. Условия, которые могут становиться ложными. Значения как условия. Истинные и ложные значения. Выход из цикла с помощью команды break. Команда continue и возврат к началу цикла. Как пользоваться командами break и continue. Намеренное создание бесконечных циклов. Что такое составные условия. Логический оператор not. Логический оператор and. Логический оператор or. Как планируют программы. Алгоритмы на псевдокоде. Пошаговая доработка алгоритма. Знакомство с игрой Отгадай число. Работа цикла for. Применение цикла for. Создание цикла for. Счет с помощью цикла for. Счет по возрастанию. Счет по числам, кратным пяти. Счет по убыванию. Применение функции len(). Применение оператора in. Индексация строк. Позиции с положительными номерами. Позиции с отрицательными номерами. Случайный элемент строки. Что такое неизменяемость строк. Создание констант. Создание новых строк из существующих. Значение None. Срезы строк. Создание срезов. Сокращения в записи срезов. Что такое кортеж. Создание кортежей. Создание пустого кортежа. Кортеж как условие. Создание непустого кортежа. Вывод элементов кортежа на экран. Перебор элементов кортежа. Применение функции len() к кортежам. Применение оператора in к кортежам. Индексация кортежей. Срезы кортежей. Изменяемость кортежей. Сцепление кортежей. Знакомство с игрой Анаграммы. Как сформировать анаграмму. Что такое списки. Создание списка. Применение функции len() к спискам. Применение оператора in к спискам. Индексация списков. Срезы списков. Сцепление списков. Изменяемость списков. Присвоение нового значения элементу, выбранному по индексу. Присвоение новых значений срезу списка. Удаление элемента списка. Удаление среза списка. Когда использовать кортежи, а когда – списки. Списочные методы. Знакомство с программой Рекорды. Что такое вложенные последовательности. Создаем вложенные последовательности. Доступ к вложенным элементам. Распаковка последовательности. Знакомство с программой Рекорды 2.0. Распределенные ссылки. Что такое словари. Создание словарей. Доступ к значениям в словаре. Поиск значения. Добавление пары «ключ - значение». Замена пары «ключ - значение». Удаление пары «ключ - значение». Обработка ошибочного выбора. Особенности словарей. Знакомство с игрой Виселица.

Практика: Установка Python в Windows. Установка Python в других операционных системах. Программа Hello, World! Работа с IDLE. Написание игры Game Over. Написание программы Game Over 2.0. Написание программы Воображаемые благодарности. Написание программы Текстовые задачи. Написание программы Привет. Написание программы Персональный привет. Написание программы Манипуляции с цитатой. Исправление

программы Рантье. Добавление функционала к программе Рантье. Начальные комментарии. Получение пользовательского ввода. Вывод name на экран в верхнем и нижнем регистре. Пятикратный вывод имени. Подсчет количества секунд. Вычисление значений moon_weight и sun_weight. Ожидание выхода. Написание программы Кости. Написание программы Пароль. Написание программы Открыто/Закрыто. Написание программы Компьютерный датчик настроения. Написание программы Симулятор трехлетнего ребенка. Написание программы Проигранное сражения. Написание программы Метрдотель. Написание программы Привередливая считалка. Написание программы Эксклюзивная сеть. Задачи на составление алгоритмов на псевдокоде. План программы. Начальный блок комментариев. Импорт модуля. Объяснение правил Установка начальных значений. Цикл отгадывания. Поздравления победителю. Ожидание выхода. Написание программы Слово по буквам. Написание программы Считалка. Написание программы Анализатор текста. Написание программы Случайные буквы. Написание программы Только согласные. Написание программы Резчик пиццы. Написание программы Арсенал героя. Написание программы Арсенал героя 2.0. Создание пустой строки для анаграммы. Настройка цикла. Выбор случайной позиции в слове. Новая версия jump1e. Новая версия word. Программа приветствует игрока. Получение пользовательского ввода. Поздравление с правильно отгаданным словом. Конец игры. Написание программы Арсенал героя 3.0. Написание программы рекорды. Отображение меню. Выход из программы. Отображение списка рекордов. Добавление рекорда. Удаление рекорда. Сортировка списка рекордов. Обработка ошибочного выбора. Ожидаем пользователя. Написание программы Рекорды 2.0. Вывод результатов, содержащихся во вложенных кортежах. Добавление результата как вложенного кортежа. Обработка ошибочного выбора. Ожидаем пользователя. Написание программы Переводчик с гикского на русский. Создание констант. Инициализация переменных. Создание основного цикла. Получение ответа игрока. Проверка наличия буквы в слове. Завершение игры. Тестирование на знание основ Python.

Формы контроля:

Готовая программа Бесплезные факты.

Готовая программа Отгадай число.

Готовая программа Анаграммы.

Тестирование.

3. Продвинутые программы на Python

Углубление знаний Python с акцентом на создание сложных программ. Учащиеся изучат работу с файлами, обработку исключений и основы модульного проектирования, что откроет возможности для реализации более функциональных приложений.

Теория: Что такое функция. Объявление функции. Документирование функции. Вызов нестандартной функции. Что такое абстракция. Передача данных с помощью параметров. Возврат значений функциями. Что такое инкапсуляция. Функции, которые и, принимают и возвращают значения. Что такое повторное использование кода. Позиционные параметры и позиционные аргументы. Позиционные параметры и именованные аргументы. Значения параметров по умолчанию. Что такое области видимости. Чтение глобальной переменной внутри функции. Затенение глобальной переменной внутри функции. Изменение глобальной переменной внутри функции. Когда использовать глобальные переменные и константы. Знакомство с игрой Крестики-нолики. План игры Крестики-нолики. Открытие и закрытие файла. Чтение текстового файла. Посимвольное чтение строки. Чтение всех строк файла в список. Перебор строк файла. запись строк в файл. запись списка строк в файл. Консервация данных и запись в файл. Чтение и расконсервация данных из файла. Полка для хранения консервированных данных. Извлечение консервированных данных через интерфейс полки. Применение конструкций try/except. Типы исключений. Обработка нескольких типов исключений. Аргумент исключения. Добавление блока else. Знакомство с игрой Викторина. Как организованы данные в текстовом файле. Основы объектно-ориентированного подхода. Объявление класса. Объявление метода. Создание объекта. Вызов метода. Создание

конструктора. Создание нескольких объектов. Инициализация атрибутов. Доступ к атрибутам. Вывод объекта на экран. Создание атрибута класса. Доступ к атрибуту класса. Создание статического метода. Вызов статического метода. Что такое инкапсуляция объектов. Создание закрытых атрибутов. Доступ к закрытым атрибутам. Создание закрытых методов. Доступ к закрытым методам. Соблюдаем приватность. В каких случаях нужны закрытые атрибуты и методы. План доработки программы Моя зверюшка. Создание свойств. Доступ к свойствам. Отправка сообщения. Прием сообщения. Сочетание объектов. Знакомство с программой Карты. Создание новых классов с помощью наследования. Создание базового класса. Наследование от базового класса. Расширение производного класса. Применение производного класса. Создание базового класса. Переопределение методов базового класса. Вызов методов базового класса. Применение производного класса. Что такое полиморфизм. Создание модулей. Импорт модулей. Применение импортированных функций и классов. Знакомство с игрой Блек-джек. Продумывание системы классов.

Практика: Написание программы Инструкция. Написание программы Принимай – возвращай. Написание программы День рождения. Написание программы Доступ отовсюду. Функция `display_instruct()`. Функция `ask_yes_no()`. Функция `ask_number()`. Функция `pieces()`. Функция `new_board()`. Функция `display_board()`. Функция `legal_moves()`. Функция `winner()`. Функция `human_move()`. Функция `computer_move()`. Функция `next_turn()`. Функция `congrat_winner()`. Функция `main()`. Запуск программы. Написание программы Прочитаем. Написание программы Запишем. Написание программы Законсервируем. Написание программы Обработаем. Функция `open_file()`. Функция `next_line()`. Функция `next_block()`. Функция `welcome()`. Настройка игры. Задание вопроса. Получение ответа. Проверка ответа. Переход к следующему вопросу. Завершение игры. Запуск функции `main()`. Написание программы Моя зверюшка. Добавление конструктора к программе. Написание программы Зверюшка с атрибутом. Написание программы Классово верная зверюшка. Написание программы Закрытая зверюшка. Написание программы Зверюшка со свойствами. Класс `Critter`. Создание зверюшки. Создание меню. Запуск программы. Написание программы Гибель пришельца. Написание программы Карты. Создание класса `Card`. Создание класса `Hand`. Применение объектов-карт. Сочетание объектов-карт в объекте `Hand`. Написание программы Карты 2.0. Написание программы Карты 3.0. Написание программы Простая игра. Модуль `cards`. Написание псевдокода для основного цикла игры. Импорт модулей `cards` и `games`. Класс `VJ_Card`. Класс `VJ_Deck`. Класс `VJ_Hand`. Класс `VJ_Player`. Класс `VJ_Dealer`. Класс `VJ_Game`. Функция `main()`. Тестирование на знание продвинутых возможностей Python.

Формы контроля:

Готовая программа Крестики-нолики.

Готовая программа Викторина.

Тестирование.

4. Разработка графических интерфейсов

Ознакомление с библиотекой `Pygame` и основами создания графических интерфейсов. Учащиеся научатся проектировать интерактивные приложения, используя графику, события и анимации, что позволит создавать собственные игры и визуально насыщенные проекты.

Теория: Что такое GUI. Что такое событийно-ориентированное программирование. Базовое окно. Знакомство с программой Простейший GUI. Создание рамки. Создание метки. Запуск событийного цикла базового окна. Создание кнопок. Запуск событийного цикла базового окна с кнопками. Импорт модуля `tkinter`. Объявление класса `Application`. Объявление метода-конструктора. Объявление метода, создающего элементы управления. Создание объекта класса `Application`. Связывание обработчика с событием. Создание обработчика события. Обертка программы. Размещение элементов управления с помощью менеджера `Grid`. Создание текстового поля. Создание текстовой области. Текстовые элементы: извлечение и вставка данных. Обертка программы. Ссылка только на родительский объект элемента управления. Создание флажков. Получение статуса флажка. Создание

переключателя. Доступ к значениям в переключателе. Обертка программы. Знакомство с программой сумасшедший сказочник. Знакомство с пакетами. Создание графического окна. Импорт модуля games. Инициализация графического экрана. Запуск основного цикла. Загрузка изображения для фоновой картинка. Установка фона. Что такое система координат графики. Отображение спрайта. загрузка изображения для спрайта. Создание спрайта. Добавление спрайта на экран. Отображение текста. Импорт модуля color. Создание объекта Text. Добавление объекта Text на экран. Создание объекта Message. Ширина и высота графического экрана. Добавление объекта Message на экран. Создание подвижных спрайтов. Настройка скорости движения спрайта. Учет границ экрана. Создание подкласса Sprite. Переопределение метода update().Обертка программы. Чтение координат указателя. Настройка видимости указателя. Перенаправление ввода в графическое окно. Регистрация столкновений. Обработка столкновений. Обертка программы. Знакомство с игрой Паника в пиццерии. Регистрация нажатий. Обертка программы. Вращение спрайта. Применение свойства angle у спрайтов. Подбор изображений. Создание списка изображений. Создание анимированного объекта. Работа со звуком. Работа с музыкой. Обертка программы. Функциональность игры. Классы игры. Медиаресурсы. Знакомство с программами Прерванный полет 1.0 – 3.0.Знакомство с программами Прерванный полет 4.0 – 5.0.Знакомство с программами Прерванный полет 6.0 – 7.0.Знакомство с программой Прерванный полет 8.0.Пути дальнейшего развития. Источники информации для изучения Python. Напутственные слова.

Практика: Импорт модуля tkinter. Создание базового окна. Изменение вида базового окна. Запуск событийного цикла базового окна. Написание программы Это я, метка. Написание программы Бесплезные кнопки. Написание программы Бесплезные кнопки 2.0.Написание программы Счетчик щелчков. Написание программы Долгожитель. Написание программы Киноман. Дополнение программы Киноман. Импорт модуля tkinter. Метод-конструктор класса Application. Метод create_widgets() класса Application. Метод tell_story() класса Application. Основная часть программы. Написание программы Новое графическое окно. Написание программы фоновая картинка. Написание программы Спрайт-пицца. Написание программы Ничего себе результат. Написание программы Победа. Написание программы Летающая птица. Написание программы скачущая птица. Написание программы Подвижная сковородка. Написание программы Ускользящая птица. Класс Pan. Класс Pizza. Класс Chef. Функция main().Написание программы Читаю с клавиатуры. Написание программы Крутящийся спрайт. Написание программы Взрыв. Написание программы Звук и музыка. Класс Asteroid. Функция main().Класс Ship. Инстанцирование класса Ship. Импорт модуля math. Добавление переменной и константы в класс Ship. Изменение метода upclate() объекта Ship. Написание программ Прерванный полет 1.0 – 3.0.Изменение метода upclate() объекта Ship. Класс Missile. Добавление константы в класс Ship. Создание метода-конструктора в классе Ship. Изменение метода update() объекта Ship. Написание программ Прерванный полет 4.0 – 5.0.Изменение метода upclate() объекта Missile. Добавление метода die() объекту Missile. Изменение метода update() объекта Ship. Добавление метода die() объекту Ship. Добавление константы в класс Asteroid. Добавление метода die() объекту Asteroid. Класс Wrapper. Класс Collider. Изменение класса Asteroid. Изменение класса Ship. Изменение класса Missile. Класс Explosion. Написание программ Прерванный полет 6.0 – 7.0.Импорт модуля color. Класс Game. Добавление переменной и константы в класс Asteroid. Изменение метода-конструктора в классе Asteroid. Изменение метода die() объекта Asteroid. Добавление константы в класс Ship. Изменение метода-конструктора в классе Ship. Изменение метода update() объекта Ship. Добавление метода die() объекту Ship. Функция main(). Написание программы Прерванный полет 8.0.Планирование собственного проекта.

Формы контроля:

Готовая программа Сумасшедший сказочник.

Готовая программа Паника в пиццерии.

Готовая программа Прерванный полет.

**КАЛЕНДАРНО-ТЕМАТИЧЕСКОЕ ПЛАНИРОВАНИЕ ДЛЯ НАЧИНАЮЩИХ
1 ГОД ОБУЧЕНИЯ**

Группа №1

(пятница, суббота)

№ занятия	Наименование раздела, темы (теория и практика)	Содержание (теоретическая и практическая часть)	Дата проведения занятия		Количество часов			Формы контроля усвоения материала	Самостоятельная работа с использованием дистанционных образовательных технологий
			по плану	фактически	Теория	Практика	Всего		
Модуль 1 Введение в программирование					6	4	10		
1.	Вводное тестирование. Знакомство с курсом. Знакомство с предметом программирования. Сферы программирования.	Теория: План курса. Что такое программирование. Программирование как работа. Сферы программирования. Практика: Вводное тестирование	05.09		1	1	2	Вводное тестирование	Занятие в группе ВК: https://vk.com/progbaza Тестирование в Google Forms
2.	Алгоритм, компьютер и язык программирования.	Теория: Понятие алгоритма. Что такое компьютер, его особенности. Характеристики и виды языков программирования. Краткая история развития языков программирования. Практика: Демонстрация кода на разных языках программирования	06.09		1,5	0,5	2		Занятие в группе ВК: https://vk.com/progbaza Курс логика на hexlet.io. Занятие 1: https://ru.hexlet.io/courses/logic/lessons/logic_basics/theory_unit
3.	Языки программирования. Сферы программирования.	Теория: Устройство компьютера. Виды программ. Сферы программирования. Виды языков программирования. Языки для изучения	12.09		1,5	0,5	2		Занятие в группе ВК: https://vk.com/progbaza Курс логика на hexlet.io. Занятие 2:

		программирования. Языки для работы. Практика: Самостоятельное изучение информации по понравившемуся направлению программирования и языку, который в нем используется.						https://ru.hexlet.io/courses/logic/lessons/logic_function_representation/theory_unit
4.	Машинный код. Двоичная система счисления.	Теория: Что такое машинный код. Что такое система счисления. Представление чисел в двоичной системе. Степени двойки. Перевод из десятичной системы в двоичную. Арифметические операции в двоичной системе. Практика: Перевод чисел из десятичной системы в двоичную. Перевод чисел из двоичной системы в десятичную. Сложение и вычитание двоичных чисел в столбик.	13.09		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Курс логика на hexlet.io. Занятие 3: https://ru.hexlet.io/courses/logic/lessons/logic_toughest_problem/theory_unit
5.	Логика. Логические операции. Преобразование логических выражений.	Теория: Что такое логика. Таблицы истинности. Простые операции И, ИЛИ, НЕ. Сложные операции исключающее ИЛИ, импликация. Основные правила преобразования логических выражений. Практика: Вывод таблиц истинности исключающего или и импликации. Вывод таблиц	19.09		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Курс логика на hexlet.io. Занятие 4: https://ru.hexlet.io/courses/logic/lessons/logic_predicates_quantifiers/theory_unit

		истинности логических выражений. Преобразование логических выражений.							
Модуль 2 Основы программирования на Python					27	31	58		
6.	Знакомство с Python. Установка Python. Первая программа.	Теория: Что такое Python. Особенности Python. Инструменты для работы с Python. Практика: Установка Python в Windows. Установка Python в других операционных системах. Программа Hello, World!	20.09		1	1	2	Проверка конспекта по Введению в программирование	Занятие в группе ВК: https://vk.com/progbaza Курс логика на hexlet.io. Занятие 5: https://ru.hexlet.io/courses/logic/lessons/logic_theories/theory_unit
7.	Знакомство с IDLE.	Теория: Что такое IDLE. Режимы программирования. Печать строк. Комментарии. Описание игры GameOver. Практика: Работа с IDLE. Написание игры GameOver.	26.09		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Курс логика на hexlet.io. Занятие 6: https://ru.hexlet.io/courses/logic/lessons/logic_paradoxes/theory_unit
8.	Строки и кавычки	Теория: Строки. Кавычки внутри строк. Вывод на экран нескольких значений. Задание завершающей строки при выводе. Тройные кавычки. Звук системного динамика Практика: Написание программы	27.09		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Эксперименты со онлайн средой выполнения Python: https://repl.it/new/python3

		GameOver 2.0.						
9.	Escape-последовательности и операции со строками	Теория: Escape-последовательности, сцепление строк. Символ продолжения строки. Повторение строк. Практика: Написание программы Воображаемые благодарности.	03.10		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Гипотенуза: https://informatics.mcmc.ru/mod/statements/view.php?id=2296#1
10.	Работа с числами	Теория: Числовые типы данных. Целочисленное деление. Остаток от деления. Математические операторы. Практика: Написание программы Текстовые задачи.	04.10		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Следующее и предыдущее: https://informatics.mcmc.ru/mod/statements/view.php?id=2296&chapterid=2937#1
11.	Переменные	Теория: Что такое переменные и литералы. Создание переменных. Использование переменных. Имена переменных. Практика: Написание программы Привет.	10.10		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Дележ яблок 1: https://informatics.mcmc.ru/mod/statements/view.php?id=2296&chapterid=2938#1
12.	Пользовательский ввод и строковые методы	Теория: Функция input. Применение функции input. Создание новых строк с помощью строковых методов. Практика:	11.10		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Дележ яблок 2: https://informatics.mcmc.ru/mod/statements/view.php?id=2296&chapterid=2939#1

		Написание программы Персональный привет. Написание программы Манипуляции с цитатой.							cme.ru/mod/statements/view.php?id=2296&chapterid=2939#1
13.	Выбор типа данных и конвертация значений	Теория: Обнаружение логических ошибок. Устранение логических ошибок. Преобразование строк в целые числа. Составные операторы присвоения. Знакомство с программой Рантье. Практика: Исправление программы Рантье. Добавление функционала к программе Рантье.	17.10		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача МКАД: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2940#1
14.	Написание программы Бесполезные факты	Теория: Знакомство с программой Бесполезные факты Практика: Начальные комментарии. Получение пользовательского ввода. Вывод патена экран в верхнем и нижнем регистре. Пятикратный вывод имени. Подсчет количества секунд. Вычисление значений moon_weight и sun_weight. Ожидание выхода.	18.10		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Задача Последняя цифра: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2941#1
15.	Генерирование случайных чисел	Теория: Импорт модуля random. Применение функции randint().Применение функции randrange(). Практика: Написание программы Кости	24.10		1	1	2	Готовая программа Бесполезные факты	Занятие в группе ВК: https://vk.com/progbaza Задача Число десятков двузначного числа:

								https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2942#1
16.	Условные конструкции с if	Теория: Как работает конструкция if. Создание условий Операторы сравнения. Создание блоков кода с помощью отступов. Создание собственных условных конструкций. Практика: Написание программы Пароль	25.10		1	1	2	Занятие в группе ВК: https://vk.com/progba za Задача Число десятков: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2943#1
17.	Конструкции if с условием else и выражения elif	Теория: Как работают условия else. Как работают условия elif. Практика: Написание программы Открыто/Закрыто. Написание программы Компьютерный датчик настройки.	31.10		1	1	2	Занятие в группе ВК: https://vk.com/progba za Задача Сумма цифр: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2944#1
18.	Создание циклов с использованием while	Теория: Как работает цикл while. Инициализация управляющей переменной. Проверка значения управляющей переменной. Изменение значения управляющей переменной. Практика: Написание программы Симулятор трехлетнего	01.11		1	1	2	Занятие в группе ВК: https://vk.com/progba za Задача Следующее четное: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2945#1

		ребенка.						
19.	Бесконечные циклы	Теория: Борьба с бесконечными циклами. Трассировка программы. Условия, которые могут становиться ложными. Значения как условия. Истинные и ложные значения. Практика: Написание программы Проигранное сражения. Написание программы Метрдетель.	07.11		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Электронные часы 1: https://informatics.mcsme.ru/mod/statements/view.php?id=2296&chapterid=2947#1
20.	Команды break и continue	Теория: Выход из цикла с помощью команды break. Команда continue и возврат к началу цикла. Как пользоваться командами break и continue. Намеренное создание бесконечных циклов. Практика: Написание программы Привередливая считалка.	08.11		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Электронные часы 2: https://informatics.mcsme.ru/mod/statements/view.php?id=2296&chapterid=2948#1
21.	Составные условия	Теория: Что такое составные условия. Логический оператор not. Логический оператор and. Логический оператор or. Практика: Написание программы Эксклюзивная сеть.	14.11		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Обмен значений: https://informatics.mcsme.ru/mod/statements/view.php?id=2296&chapterid=2949#1

22.	Планирование программ	Теория: Как планируют программы. Алгоритмы на псевдокоде. Пошаговая доработка алгоритма. Практика: Задачи на составление алгоритмов на псевдокоде.	15.11		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Изучение руководства по стилю кода PEP 8: https://pep8.ru/doc/pep8/
23.	Написание игры Отгадай число	Теория: Знакомство с игрой Отгадай число. Практика: План программы Начальный блок комментариев. Импорт модуля. Объяснение правил Установка начальных значений. Цикл отгадывания. Поздравления победителю. Ожидание выхода.	21.11		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Задача Конец занятия: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2950#1
24.	Циклы for	Теория: Работа цикла for. Применение цикла for. Создание цикла for. Счет с помощью цикла for. Счет по возрастанию. Счет по числам, кратным пяти. Счет по убыванию. Практика: Написание программы Слово по буквам. Написание программы Считалка.	22.11		1	1	2	Готовая программа Отгадай число	Занятие в группе ВК: https://vk.com/progbaza Задача Стоимость покупки: https://informatics.mccme.ru/mod/statements/view.php?id=2296&chapterid=2951#1
25.	Операторы и функции для работы с последовательностями: применение к строкам	Теория: Применение функции len(). Применение оператора in. Индексация строк. Позиции с	28.11		1	1	2		Занятие в группе ВК: https://vk.com/progbaza

		положительными номерами. Позиции с отрицательными номерами. Случайный элемент строки. Практика: Написание программы Анализатор текста. Написание программы Случайные буквы.						Задача Разность времен: https://informatics.mcmce.ru/mod/statements/view.php?id=2296&chapterid=2952#1
26.	Конструирование новых строк	Теория: Что такое неизменяемость строк. Создание констант. Создание новых строк из существующих. Значение None. Срезы строк. Создание срезов. Сокращения в записи срезов. Практика: Написание программы Только согласные. Написание программы Резчик пиццы.	29.11		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Автопробег: https://informatics.mcmce.ru/mod/statements/view.php?id=2296&chapterid=2953#1
27.	Создание кортежей	Теория: Что такое кортеж. Создание кортежей. Создание пустого кортежа. Кортеж как условие. Создание непустого кортежа. Вывод элементов кортежа на экран. Перебор элементов кортежа. Практика: Написание программы Арсенал героя	05.12		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Дележ яблок 3: https://informatics.mcmce.ru/mod/statements/view.php?id=2296&chapterid=2954#1
28.	Использование кортежей	Теория: Применение функции len() к кортежам. Применение оператора in к кортежам. Индексация кортежей. Срезы кортежей. Неизменяемость	06.12		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Улитка: https://informatics.mcmce.ru/mod/statements/view.php?id=2296&chapterid=2955#1

		кортежей. Сцепление кортежей. Практика: Написание программы Арсенал героя 2.0.							cme.ru/mod/statements/view.php?id=2296&chapterid=2955#1
29.	Написание игры Анаграммы	Теория: Знакомство с игрой Анаграммы. Как сформировать анаграмму. Практика: Создание пустой строки для анаграммы. Настройка цикла. Выбор случайной позиции в слове. Новая версия jumble. Новая версия word. Программа приветствует игрока. Получение пользовательского ввода. Поздравление с правильно отгаданным словом. Конец игры.	12.12		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Задача Четные числа: https://informatics.mccme.ru/mod/statements/view.php?id=280#1
30.	Использование списков	Теория: Что такое списки. Создание списка. Применение функции len() к спискам. Применение оператора in к спискам. Индексация списков. Срезы списков. Сцепление списков. Изменяемость списков. Присвоение нового значения элементу, выбранному по индексу. Присвоение новых значений срезу списка. Удаление элемента списка. Удаление среза списка. Практика: Написание программы Арсенал героя 3.0.	13.12		1	1	2	Готовая программа Анаграммы	Занятие в группе ВК: https://vk.com/progbaza Задача Остаток: https://informatics.mccme.ru/mod/statements/view.php?id=280&chapterid=334#1

31.	Применение списочных методов	<p>Теория: Когда использовать кортежи, а когда – списки. Списочные методы. Знакомство с программой Рекорды.</p> <p>Практика: Написание программы рекорды. Отображение меню. Выход из программы. Отображение списка рекордов. Добавление рекорда. Удаление рекорда. Сортировка списка рекордов. Обработка ошибочного выбора. Ожидаем пользователя.</p>	19.12		1	1	2		<p>Занятие в группе ВК: https://vk.com/progba za</p> <p>Задача Квадраты: https://informatics.mcsme.ru/mod/statements/view.php?id=280&chapterid=335#1</p>
32.	Вложенные последовательности	<p>Теория: Что такое вложенные последовательности. Создаем вложенные последовательности. Доступ к вложенным элементам. Распаковка последовательности. Знакомство с программой Рекорды 2.0.</p> <p>Практика: Написание программы Рекорды 2.0. Вывод результатов, содержащихся во вложенных кортежах. Добавление результата как вложенного кортежа. Обработка ошибочного выбора. Ожидаем пользователя.</p>	20.12		1	1	2		<p>Занятие в группе ВК: https://vk.com/progba za</p> <p>Задача Минимальный делитель: https://informatics.mcsme.ru/mod/statements/view.php?id=280&chapterid=339#1</p>

33.	Использование словарей.	Теория: Распределенные ссылки. Что такое словари. Создание словарей. Доступ к значениям в словаре. Поиск значения. Добавление пары «ключ - значение». Замена пары «ключ - значение». Удаление пары «ключ - значение». Обработка ошибочного выбора. Особенности словарей. Практика: Написание программы Переводчик с гикского на русский.	26.12		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Делитель числа: https://informatics.mccme.ru/mod/statements/view.php?id=280&chapterid=340#1
34.	Написание игры Виселица	Теория: Знакомство с игрой Виселица. Практика: Создание констант. Инициализация переменных. Создание основного цикла. Получение ответа игрока. Проверка наличия буквы в слове. Завершение игры. Тестирование на знание основ Python.	27.12		0,5	1,5	2	Тестирование на знание основ Python.	Занятие в группе ВК: https://vk.com/progbaza Тестирование в Google Forms
Модуль 3Продвинутые программы на Python					16	20	36		
35.	Создание функций	Теория: Что такое функция. Объявление функции. Документирование функции. Вызов нестандартной функции. Что такое абстракция. Практика: Написание программы Инструкция.	09.01		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача НОД: https://informatics.mccme.ru/mod/statements/view.php?id=268#1

36.	Параметры и возвращаемые значения	Теория: Передача данных с помощью параметров. Возврат значений функциями. Что такое инкапсуляция. Функции, которые и, принимают и возвращают значения. Что такое повторное использование кода. Практика: Написание программы Принимай – возвращай.	10.01		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Площадь комнаты: https://informatics.mccme.ru/mod/statements/view.php?id=268&chapterid=200#1
37.	Именованные аргументы и значения параметров по умолчанию	Теория: Позиционные параметры и позиционные аргументы. Позиционные параметры и именованные аргументы. Значения параметров по умолчанию. Практика: Написание программы День рождения.	16.01		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Фишки: https://informatics.mccme.ru/mod/statements/view.php?id=268&chapterid=1414#1
38.	Использование глобальных переменных и констант	Теория: Что такое области видимости. Чтение глобальной переменной внутри функции. Затенение глобальной переменной внутри функции. Изменение глобальной переменной внутри функции. Когда использовать глобальные переменные и константы. Практика: Написание программы Доступ отовсюду.	17.01		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Спиралька: https://informatics.mccme.ru/mod/statements/view.php?id=268&chapterid=1470#1
39.	Написание игры	Теория:	23.01		0,5	1,5	2		Занятие в группе ВК:

	Крестики-нолики	<p>Знакомство с игрой Крестики-нолики. План игры Крестики-нолики.</p> <p>Практика:</p> <p>Функция display_instruct(). Функция ask_yes_no(). Функция ask_number(). Функция pieces(). Функция new_board(). Функция display_board(). Функция legal_moves(). Функция winner(). Функция human_move(). Функция computer_move(). Функция next_turn(). Функция congrat_winner(). Функция main().</p> <p>Запуск программы.</p>						<p>https://vk.com/progbaza</p> <p>Задача Ханойские башни:</p> <p>https://informatics.mcsme.ru/mod/statements/view.php?id=2550#1</p>
40.	Чтение текстового файла и запись в него	<p>Теория:</p> <p>Открытие и закрытие файла. Чтение текстового файла. Посимвольное чтение строки. Чтение всех строк файла в список. Перебор строк файла. запись строк в файл. запись списка строк в файл.</p> <p>Практика:</p> <p>Написание программы</p> <p>Прочитаем. Написание программы</p> <p>Запишем.</p>	24.01	1	1	2	<p>Готовая программа Крестики-нолики</p>	<p>Занятие в группе ВК:</p> <p>https://vk.com/progbaza</p> <p>Задача Ремонт в Ханое:</p> <p>https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3051#1</p>

41.	Хранение структурированных данных в файлах	Теория: Консервация данных и запись в файл. Чтение и расконсервация данных из файла. Полка для хранения консервированных данных. Извлечение консервированных данных через интерфейс полки. Практика: Написание программы Законсервируем.	30.01		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Циклические башни: https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3052#1
42.	Обработка исключений	Теория: Применение конструкций try/except. Типы исключений. Обработка нескольких типов исключений. Аргумент исключения. Добавление блока else. Практика: Написание программы Обработкаем.	31.01		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Несправедливые башни: https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3053#1
43.	Написание игры Викторина	Теория: Знакомство с игрой Викторина. Как организованы данные в текстовом файле. Практика: Функцияopen_file(). Функцияnext_line(). Функцияnext_block(). Функцияwelcome(). Настройка игры. Задание вопроса. Получение ответа. Проверка ответа. Переход к следующему	06.02		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Задача Сортирующие башни: https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3054#1

		вопросу Завершение игры. Запуск функции main().							
44.	Создание классов, методов и объектов.	Теория: Основы объектно-ориентированного подхода. Объявление класса. Объявление метода. Создание объекта. Вызов метода. Создание конструктора. Создание нескольких объектов. Практика: Написание программы Моя зверюшка. Добавление конструктора к программе.	07.02		1	1	2	Готовая программа Викторина	Занятие в группе ВК: https://vk.com/progbaza Задача Обменные башни: https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3055#1
45.	Применение атрибутов	Теория: Инициализация атрибутов. Доступ к атрибутам. Вывод объекта на экран. Создание атрибута класса. Доступ к атрибуту класса. Создание статического метода. Вызов статического метода. Практика: Написание программы Зверюшка с атрибутом. Написание программы Классово верная зверюшка.	13.02		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача Ханойская сортировка: https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3283#1
46.	Инкапсуляция объектов	Теория: Что такое инкапсуляция объектов. Создание закрытых атрибутов. Доступ к закрытым атрибутам. Создание закрытых методов. Доступ к закрытым методам. Соблюдаем	14.02		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Задача N-е число Фибоначчи: https://informatics.mcsme.ru/mod/statements/view.php?id=2550&chapterid=3283#1

		<p>приватность. В каких случаях нужны закрытые атрибуты и методы.</p> <p>Практика: Написание программы Закрытая зверюшка.</p>							<p>cme.ru/mod/statements/view.php?id=253#1</p>
47.	Доработка программы Моя зверюшка	<p>Теория: План доработки программы Моя зверюшка. Создание свойств. Доступ к свойствам.</p> <p>Практика: Написание программы Зверюшка со свойствами. Класс Critter. Создание зверюшки. Создание меню. Запуск программы.</p>	20.02		0,5	1,5	2		<p>Занятие в группе ВК: https://vk.com/progba za</p> <p>Задача НОД (рекурсивный вариант): https://informatics.mcm.ru/mod/statements/view.php?id=253&chapterid=154#1</p>
48.	Сочетание объектов	<p>Теория: Отправка сообщения. Прием сообщения. Сочетание объектов. Знакомство с программой Карты.</p> <p>Практика: Написание программы Гибель пришельца. Написание программы Карты. Создание класса Card. Создание класса Hand. Применение объектов-карт. Сочетание объектов-карт в объекте Hand.</p>	21.02		1	1	2		<p>Занятие в группе ВК: https://vk.com/progba za</p> <p>Задача Генератор: https://informatics.mcm.ru/mod/statements/view.php?id=253&chapterid=155#1</p>
49.	Наследование	<p>Теория: Создание новых классов с помощью наследования. Создание базового класса. Наследование от базового класса. Расширение</p>	27.02		1	1	2		<p>Занятие в группе ВК: https://vk.com/progba za</p> <p>Задача Без массивов: https://informatics.mcm.ru/mod/statements/view.php?id=253&chapterid=156#1</p>

		производного класса. Применение производного класса. Практика: Написание программы Карты 2.0.						cme.ru/mod/statements/view.php?id=253&chapterid=156#1
50.	Переопределение унаследованных методов	Теория: Создание базового класса. Переопределение методов базового класса. Вызов методов базового класса. Применение производного класса. Практика: Написание программы Карты 3.0.	28.02		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Когда корень не равен корню: https://informatics.mcmce.ru/mod/statements/view.php?id=2307#1
51.	Полиморфизм	Теория: Что такое полиморфизм. Создание модулей. Импорт модулей. Применение импортированных функций и классов. Практика: Написание программы Простая игра.	06.03		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Задача Соревнования картингистов: https://informatics.mcmce.ru/mod/statements/view.php?id=2042&chapterid=1922#1
52.	Написание игры Блек-джек	Теория: Знакомство с игрой Блек-джек. Продумывание системы классов. Практика: Модуль cards. Написание	07.03		0,5	1,5	2	Тестирование на знание продвинутых возможностей Python. Занятие в группе ВК: https://vk.com/progbaza Тестирование в Google Forms

		псевдокода для основного цикла игры. Импорт модулей cards и games. Класс BJ_Card. Класс BJ_Deck. Класс BJ_Hand. Класс BJ_Player. Класс BJ_Dealer. Класс BJ_Game. Функция main(). Тестирование на знание продвинутых возможностей Python.							
Модуль 4 Разработка графических интерфейсов					17	16.0 3	40		
53.	GUI в подробностях	Теория: Что такое GUI. Что такое событийно-ориентированное программирование. Базовое окно. Знакомство с программой Простейший GUI. Практика: Импорт модуля tkinter. Создание базового окна. Изменение вида базового окна. Запуск событийного цикла базового окна.	13.03		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 1: https://younglinux.info/pygame/pygame
54.	Применение меток и кнопок	Теория: Создание рамки. Создание метки. Запуск событийного цикла базового окна. Создание кнопок. Запуск событийного цикла базового окна с кнопками. Практика: Написание программы Это я, метка. Написание программы Беспольные кнопки.	14.03		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 2: https://younglinux.info/pygame/framework
55.	Создание GUI с помощью класса	Теория: Импорт модуля tkinter.	20.03		1	1	2		Занятие в группе ВК: https://vk.com/progba

		Объявление класса Application. Объявление метода-конструктора. Объявление метода, создающего элементы управления. Создание объекта класса Application Практика: Написание программы Бесполезные кнопки 2.0.							za Просмотр курса по библиотеке Pygame. Занятие 3: https://younglinux.info/pygame/draw
56.	Связывание элементов управления с обработчиками событий	Теория: Связывание обработчика с событием. Создание обработчика события. Обертка программы. Практика: Написание программы Счетчик щелчков	21.03		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 4: https://younglinux.info/pygame/key
57.	Текстовые поля и области. Менеджер размещения Grid	Теория: Размещение элементов управления с помощью менеджера Grid. Создание текстового поля. Создание текстовой области. Текстовые элементы: извлечение и вставка данных. Обертка программы. Практика: Написание программы Долгожитель.	27.03		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 5: https://younglinux.info/pygame/mouse
58.	Применение флажков и переключателей	Теория: Ссылка только на родительский объект элемента управления. Создание флажков. Получение статуса	28.03		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по

		флажка. Создание переключателя. Доступ к значениям в переключателе. Обертка программы. Практика: Написание программы Киноман. Дополнение программы Киноман.						библиотеке Pygame. Занятие 6: https://younglinux.info/pygame/surface
59.	Написание программы Сумасшедший сказочник	Теория: Знакомство с программой сумасшедший сказочник. Практика: Импорт модуля tkinter. Метод-конструкторклассаApplication. Метод create_widgets() класса Application. Метод tell_story() класса Application. Основная часть программы.	03.04		0,5	1,5	2	Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 7: https://younglinux.info/pygame/rect
60.	Знакомство с пакетами pygame и livewires	Теория: Знакомство с пакетами. Создание графического окна. Импорт модуля games. Инициализация графического экрана. Запуск основного цикла. Загрузка изображения для фоновой картинка. Установка фона. Практика: Написание программы Новое графическое окно. Написание программы фоновая картинка.	04.04		1	1	2	Готовая программа Сумасшедший сказочник Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 8: https://younglinux.info/pygame/font
61.	Система координат графики	Теория: Что такое система координат графики. Отображение спрайта. загрузка изображения для спрайта. Создание спрайта. Добавление спрайта	10.04		1	1	2	Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame.

		на экран. Практика: Написание программы Спрайт-пицца							Занятие 9: https://younglinux.info/pygame/image
62.	Текст и сообщения	Теория: Отображение текста. Импорт модуля color. Создание объекта Text. Добавление объекта Text на экран. Создание объекта Message. Ширина и высота графического экрана. Добавление объекта Message на экран. Практика: Написание программы Ничего себе результат. Написание программы Победа.	11.04		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 10: https://younglinux.info/pygame/sprite
63.	Подвижные спрайты	Теория: Создание подвижных спрайтов. Настройка скорости движения спрайта. Учет границ экрана. Создание подкласса Sprite. Переопределение метода update(). Обертка программы. Практика: Написание программы Летящая птица. Написание программы скачущая птица.	17.04		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Просмотр курса по библиотеке Pygame. Занятие 11: https://younglinux.info/pygame/mixer
64.	Обработка ввода с помощью мыши	Теория: Чтение координат указателя. Настройка видимости указателя. Перенаправление ввода в графическое окно. Регистрация столкновений. Обработка столкновений.	18.04		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Начало работы с Git: https://githowto.com/ru

		Обертка программы. Практика: Написание программы Подвижная сковорода. Написание программы Ускользящая птица.							
65.	Написание игры Паника в пиццерии	Теория: Знакомство с игрой Паника в пиццерии Практика: Класс Pan. Класс Pizza. Класс Chef. Функция main().	24.04		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Курс Введение в Git. Занятие 1: https://ru.hexlet.io/courses/intro_to_git/lessons/intro/theory_unit
66.	Вращение спрайта	Теория: Регистрация нажатий. Обертка программы. Вращение спрайта. Применение свойства angle у спрайтов. Практика: Написание программы Читаю с клавиатуры. Написание программы Крутящийся спрайт.	25.04		1	1	2	Готовая программа Паника в пиццерии	Занятие в группе ВК: https://vk.com/progbaza Курс Введение в Git. Занятие 2: https://ru.hexlet.io/courses/intro_to_git/lessons/setup/theory_unit
67.	Создание анимации	Теория: Подбор изображений. Создание списка изображений. Создание анимированного объекта. Работа со звуком. Работа с музыкой. Обертка программы. Практика: Написание программы Взрыв. Написание программы Звук и музыка.	02.05		1	1	2		Занятие в группе ВК: https://vk.com/progbaza Курс Введение в Git. Занятие 3: https://ru.hexlet.io/courses/intro_to_git/lessons/workflow/theory_unit

68.	Разработка игры Прерванный полет	Теория: Функциональность игры. Классы игры. Медиаресурсы. Знакомство с программами Прерванный полет 1.0 – 3.0. Практика: Класс Asteroid. Функция main().Класс Ship. Инстанцирование класса Ship. Импорт модуля math. Добавление переменной и константы в класс Ship. Изменение метода upclate() объекта Ship. Написание программ Прерванный полет 1.0 – 3.0.	08.05		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Курс Введение в Git. Занятие 4: https://ru.hexlet.io/courses/intro_to_git/lessons/github/theory_unit
69.	Добавление функций стрельбы	Теория: Знакомство с программами Прерванный полет 4.0 – 5.0. Практика: Изменение метода upclate() объекта Ship.Класс Missile. Добавление константы в класс Ship. Создание метода- конструктора в классе Ship. Изменение метода update() объекта Ship. Написание программ Прерванный полет 4.0 – 5.0.	15.05		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Курс Введение в Git. Занятие 5: https://ru.hexlet.io/courses/intro_to_git/lessons/working-directory/theory_unit
70.	Обработка столкновений и добавление взрывов	Теория: Знакомство с программами Прерванный полет 6.0 – 7.0. Практика: Изменение метода upclate() объекта Missile. Добавление метода die() объекту Missile. Изменение метода update()	16.05		0,5	1,5	2		Занятие в группе ВК: https://vk.com/progbaza Курс Введение в Git. Занятие 6: https://ru.hexlet.io/courses/intro_to_git/less

		<p>объекта Ship. Добавление метода die() объекту Ship. Добавление константы в класс Asteroid. Добавление метода die() объекту Asteroid. Класс Wrapper. Класс Collider. Изменение класса Asteroid. Изменение класса Ship. Изменение класса Missile. Класс Explosion. Написание программ Прерванный полет 6.0 – 7.0.</p>							<p>ons/changes/theory_unit</p>
71.	Уровни, ведение счета, музыкальная тема	<p>Теория: Знакомство с программой Прерванный полет 8.0. Практика: Импорт модуля color. Класс Game. Добавление переменной и константы в класс Asteroid. Изменение метода-конструктора в классе Asteroid. Изменение метода die() объекта Asteroid. Добавление константы в класс Ship. Изменение метода-конструктора в классе Ship. Изменение метода update() объекта Ship. Добавление метода die() объекту Ship. Функция main(). Написание программы Прерванный полет 8.0.</p>	22.05		0,5	1,5	2		<p>Занятие в группе ВК: https://vk.com/progbaza Загрузка проектов с курса на Github: https://github.com/</p>
72.	Развитие программирования в	<p>Теория: Пути дальнейшего развития. Источники информации для изучения Python. Напутственные слова.</p>	23.05		1	1	2	<p>Готовая программа Прерванный полет</p>	<p>Занятие в группе ВК: https://vk.com/progbaza Лекция Языки</p>

		Практика: Планирование собственного проекта.							программирования: критерии выбора: https://www.youtube.com/watch?v=T70qJndjYi0&ab_channel=IT%D0%A8%D0%9A%D0%9E%D0%9B%D0%90SAMSUNG
	Всего				66	78	144		

Рабочая программа
К ОБЩЕОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЕ
«Программирование. Подготовка к IT-профессии»
2 год обучения
Задачи 2 года обучения

Обучающие задачи:

- Дать базовые и углубленные знания в области алгоритмов и информатики.
- Обучить основам и принципам объектно-ориентированного программирования.
- Освоить программирование на Python и научить разрабатывать базовые веб-страницы и приложения.
- Познакомить с игровым движком Godot для создания интерактивных приложений.
- Научить планированию и реализации проектов с использованием профессиональных инструментов, включая базы данных и системы контроля версий.

Развивающие задачи:

- Развить алгоритмическое и проектное мышление.
- Углубить навыки командной работы и управления проектами.
- Развить навыки поиска информации, самообучения и критического мышления.
- Закрепить умение творчески подходить к решению технических задач.

Воспитательные задачи:

- Воспитывать трудолюбие, ответственность за результат и готовность к саморазвитию.
- Развить уверенность в собственных силах и способность достигать профессиональных целей.
- Формировать активную личностную позицию и умение работать в команде.

СОДЕРЖАНИЕ ОБУЧЕНИЯ 2-ОГО ГОДА

Программа второго года обучения направлена на углубление и систематизацию знаний и навыков, полученных ранее, с особым акцентом на развитии проектных компетенций и освоении новых областей программирования.

Она охватывает такие ключевые направления, как объектно-ориентированное программирование (ООП), классические алгоритмы, разработка игр на Godot, а также создание и защита индивидуальных проектов. Учебный план рассчитан на развитие навыков, необходимых для создания собственных проектов.

1. Объектно-ориентированное программирование (ООП)

В этом блоке ученики углубленно изучат основы и принципы ООП, что включает понятия классов и объектов, атрибутов и методов. Практическая часть включает проектирование простых классов и объектов, создание текстовых квестов и симуляций. Особое внимание уделяется таким темам, как конструкторы (`__init__()`), атрибуты и методы, наследование классов. В рамках проекта дети создадут классы для текстовых игр и симуляторов, что позволит им закрепить принципы ООП на практике и научиться организовывать код для многофункциональных приложений.

Теория: Обсуждение содержания курса. Пример из жизни: объекты и классы (сравнение с реальными вещами — машина, телефон). Что такое класс в программировании. Как создавать объекты на Python. Что такое атрибуты (свойства) и методы (действия) объектов. Примеры из жизни: телефон звонит, машина едет. Как создавать объекты с помощью конструктора `__init__()`. Значение свойств по умолчанию. Создание класса Student, его атрибутов (имя, возраст, класс) и методов (например, изменение класса или получение информации о студенте). Планирование проекта "Монополия". Описание классов: Cell (клетка поля), Player (игрок), Dice (кубик). Доступ к текущим значениям атрибутов, изменение значений атрибутов. Как спроектировать текстовый квест с использованием ООП. Описание классов Location (место) и Choice (выбор). Хранение ссылки на объект стороннего класса в качестве значения атрибута. Основы физического моделирования с использованием ООП. Описание класса Pendulum, который моделирует движение маятника. Как моделировать движение объектов в пространстве. Описание классов Planet (планета) и Rocket (ракета). Что такое наследование. Пример: класс Animal и классы-наследники Dog и Cat.

Практика: прохождение учениками теста для определения остаточных знаний с предыдущего года обучения. Создание простого класса Car с атрибутами (цвет, марка). Создание объектов на

основе класса. Создание класса Book, который получает название и автора при создании объекта. Создание простого приложения, где можно добавлять, удалять и изменять данные учеников. Создание основных классов для игры, написание методов для генерации случайного числа для кубика и движения игрока по клеткам. Добавление функционала в игру: покупка недвижимости, просмотр баланса и списка приобретенной ранее недвижимости. Создание простого квеста с несколькими локациями и выбором действий для игрока. Создание класса Achievement, описывающего достижение в квесте, привязка достижений к действиям в квесте, хранение достижений в списке. Написание простого симулятора движения маятника с использованием ООП. Написание программы, в которой ракета движется вокруг планеты или взлетает с её поверхности. Создание класса Animal с общими методами, а затем наследование для классов Dog и Cat с уникальными особенностями.

Формы контроля:

Выполнение задания «Класс “Библиотека”»

Выполнение задания «Текстовая RPG-игра»

2. Классические алгоритмы на Python

В данном разделе учащиеся разберут базовые и продвинутые алгоритмы, включая сортировку, поиск и жадные алгоритмы. Будут подробно рассмотрены алгоритмы, такие как линейный и бинарный поиск, пузырьковая и выборочная сортировки, а также алгоритмы рекурсии и нахождения НОД. Практические занятия включают написание программ для решения классических задач, таких как задача о сдаче или задача о ранце. Это позволяет ученикам развить навыки алгоритмического мышления и научиться эффективно решать задачи с оптимальной производительностью.

Теория: Определение алгоритма как последовательности шагов для выполнения задачи. Примеры алгоритмов в повседневной жизни. Важность алгоритмов в программировании. История развития алгоритмов. Алгоритмы в эпоху компьютеров. Что такое линейные алгоритмы и их основные характеристики. Алгоритмы, которые выполняются последовательно. Что такое поиск в алгоритмах. Линейный поиск и его применение. Введение в сортировку данных. Простой алгоритм сортировки выбором: как он работает и его особенности. Алгоритм пузырьковой сортировки. Сравнение пузырьковой сортировки с сортировкой выбором. Алгоритм бинарного поиска. Как бинарный поиск отличается от линейного и в каких случаях его использовать. Что такое рекурсия. Примеры рекурсивных алгоритмов в жизни (например, русские матрёшки). Как работает рекурсия в программировании. Что такое наибольший общий делитель (НОД) и как его можно найти. Описание алгоритма Евклида. Что такое жадные алгоритмы. Пример жадного алгоритма: задача о сдаче (наименьшее количество монет для данной суммы). Классические примеры применения жадных алгоритмов – задача о ранце и задача выбора заявок. Что такое поиск в ширину и для чего он используется. Пример использования для поиска кратчайшего пути. Обзор изученных алгоритмов: сортировка, поиск, рекурсия, жадные алгоритмы.

Практика: Написание алгоритмов для последовательного выполнения действий, например, нахождение максимального и минимального значения в списке. Написание программы для поиска элемента в списке с использованием линейного поиска. Написание программы для сортировки списка с использованием сортировки выбором. Реализация пузырьковой сортировки на Python. Сортировка списка по возрастанию и убыванию. Написание программы для поиска элемента в отсортированном списке с использованием бинарного поиска. Написание рекурсивного алгоритма для вычисления факториала числа и решения других простых задач. Реализация алгоритма Евклида на Python для нахождения НОД двух чисел. Написание программы для решения задачи о сдаче с использованием жадного алгоритма. Написание программы для поиска пути в лабиринте с использованием алгоритма "Поиск в ширину". Решение задач на применение различных алгоритмов. Создание программы, которая использует несколько алгоритмов для выполнения различных задач (например, сортировка списка с последующим поиском).

Формы контроля:

Выполнение задания «Сравнение алгоритмов»

Выполнение задания «Минимальный набор монет»

3. Разработка игр на движке Godot

Учащиеся познакомятся с возможностями игрового движка Godot, включая основы работы с 3D-сценами и узлами, освоят принципы программирования на GDScript и научатся использовать такие элементы, как таймеры, сигналы, камеры, физические тела и анимации. Практическая работа включает создание сцен с узлами, настройку освещения, управление интерфейсом и анимациями, разработку управления персонажем и его взаимодействия с объектами. Этот блок позволит детям получить опыт создания простых 3D-игр и работы с физическими и интерактивными элементами в игровом пространстве.

Теория: Что такое Godot, его возможности, использование в разработке игр. Основные компоненты интерфейса. Понятие сцены и узлов в Godot. Как они связаны между собой. Виды узлов. Как работают трансформации в 3D: перемещение, вращение и масштабирование объектов. Принцип работы координатных систем. Как работает камера в 3D. Виды камер и их настройка для различных перспектив. Введение в GDScript. Переменные, типы данных, условия и циклы. Функции, классы и методы в GDScript. Организация кода в игре. Что такое сигналы в Godot и как они помогают взаимодействовать между объектами и сценами. Как подключать и отправлять сигналы между узлами. Введение в структуры данных GDScript: массивы и словари. Примеры их использования для хранения данных, таких как инвентарь или характеристики персонажей. Работа с таймерами в GDScript. Как создавать задержки выполнения действий и как это используется в игровых механиках (например, создание атак, паузы или задержки перед действиями). Введение в ресурсы (Resource) в Godot. Как ресурсы помогают сохранять данные и настройки объектов вне основной сцены. Как создавать объекты на сцене динамически через скрипты. Использование метода `instance()` для клонирования объектов. Введение в физику в 3D. Типы тел: статические, кинематические, динамические. Обработка столкновений. Как работает гравитация в 3D и как моделировать движение объектов. Основы создания анимации в 3D. Использование Animation Player для создания анимации. Создание и настройка элементов интерфейса (жизни, очки, кнопки) для 3D игр. Взаимодействие между UI и 3D миром. Как создавать игровые уровни в 3D. Понятие модульности и повторного использования объектов. Виды освещения в Godot: точечный, направленный свет, окружение. Настройка теней. Скриптинг для управления персонажем. Реализация взаимодействий с окружением. Как обрабатывать столкновения в 3D играх, используя кинематическое тело. Углубленное изучение физики в 3D. Как работать с физическими взаимодействиями объектов. Основы создания ИИ для персонажей в играх. Патрулирование, преследование игрока. Как создавать пути и траектории в 3D пространстве. Использование узла Path для задания пути движения объектов и узла PathFollow3D для управления их движением по траектории. Как работают эффекты частиц в 3D. Настройка и оптимизация частиц. Как оптимизировать 3D игру для повышения производительности. Управление ресурсами и памятью.

Практика: Установка Godot, знакомство с интерфейсом и создание первого проекта. Создание простой сцены с несколькими узлами (например, персонаж и предметы). Написание скриптов для перемещения объектов в 3D пространстве (например, вращение куба или перемещение шара). Добавление камеры на сцену, настройка её положения. Скрипт для динамического управления камерой, которая следует за игроком. Написание простых скриптов для управления объектами на сцене (например, изменение цвета или размера объекта через код). Написание скриптов для управления игровыми элементами, создание пользовательских функций. Написание скриптов для отправки сигналов при определённых событиях (например, персонаж подбирает предмет, враг обнаружил игрока). Подключение сигналов для взаимодействия между разными объектами. Написание скриптов для управления инвентарём персонажа с использованием массива. Работа со словарём для хранения атрибутов персонажей или предметов (например, сила, здоровье, урон). Создание таймеров в игре для выполнения действий с задержкой (например, атака врага через 3 секунды после его появления). Создание пользовательских ресурсов для хранения данных (например, характеристик персонажей, оружия или инвентаря). Работа с ресурсами через скрипты. Написание кода для генерации врагов, препятствий или других объектов в процессе игры. Пример: динамическое создание платформ в платформере или генерация астероидов в космической игре. Добавление физики на сцену. Использование узлов RigidBody и CollisionShape для создания объектов, которые взаимодействуют друг с другом (например, падение объектов). Написание

скриптов для имитации гравитации и движения объектов. Пример: падение объектов под действием гравитации или движение шара по наклонной плоскости. Создание анимации вращения, перемещения или изменения масштаба объекта. Управление анимацией через скрипты. Добавление интерфейса в игру, вывод очков и жизней на экран. Настройка кнопок для управления игрой. Создание простого уровня с препятствиями и различными объектами. Добавление света на сцену, настройка теней для реалистичной визуализации. Изменение освещения в зависимости от событий на сцене. Написание скриптов для перемещения персонажа в 3D, прыжков и взаимодействия с объектами. Написание скриптов для обработки столкновений персонажа с объектами, препятствиями. Написание сложных взаимодействий объектов, таких как столкновения с силой или катапультирование объектов. Написание скрипта для врага, который двигается по сцене, преследует игрока и реагирует на его действия. Создание пути для движения объекта (например, врага или транспорта) по заданной траектории в 3D пространстве. Написание скриптов для контроля скорости и положения объекта на пути. Создание эффекта частиц для визуализации огня, дыма, взрывов и других эффектов. Оптимизация готового проекта (например, уменьшение нагрузки на процессор за счёт правильной настройки объектов и скриптов).

Формы контроля:

Выполнение задания «Платформер»

Выполнение задания «Лабиринт»

4. Индивидуальные проекты

В заключительной части курса учащиеся разработают собственные проекты, выбирая между играми, приложениями или веб-проектами. Работы над проектом включают постановку целей по принципу SMART, анализ аналогов, планирование структуры, разработку базового функционала и реализацию сложных функций. Ученики будут применять такие технологии, как базы данных, мультимедиа и API, проводить тестирование и оптимизацию, а также научатся работать с Git. В конце обучения каждый учащийся защитит свой проект перед аудиторией, что позволит продемонстрировать полученные навыки и достижения.

Теория: Что такое индивидуальный проект и его значение в обучении. Примеры успешных проектов в разных направлениях (игры, веб-сайты, приложения). Принципы SMART (Specific, Measurable, Achievable, Relevant, Time-bound) для постановки целей. Цели для игр, сайтов и приложений.

Исследование аналогичных проектов. Анализ требований и технологий, которые помогут в реализации проекта. Как спланировать структуру проекта (модульный подход). Разработка основных модулей. Важность прототипов для игры, сайта или приложения. Как выбрать правильные инструменты для проекта. Обзор движков, фреймворков и библиотек для разных направлений. Как построить минимально жизнеспособный продукт (MVP). Определение базовых задач.

Принципы проектирования удобного интерфейса для игр, сайтов и приложений. Как добавить интерактивность и дополнительные функции в проект. Почему важно тестирование на каждом этапе. Основные виды тестирования. Основные принципы оптимизации для игр, сайтов и приложений.

Введение в базы данных. Как сохранять и загружать данные в проекте. Основы работы с сетями в проекте. Примеры применения сетевых функций. Что такое системы контроля версий и как они помогают в разработке. Как мультимедиа (графика, звук) улучшает пользовательский опыт. Основы работы с изображениями, звуками и видео в проектах. Что такое система достижений и статистики, как её можно применять для мотивации и улучшения опыта пользователя. Что такое API и как оно используется для добавления внешних данных и функций в проект. Основы работы с внешними сервисами и их использование для улучшения проекта. Примеры авторизации через соцсети. Основы рефакторинга: зачем упрощать и улучшать структуру кода, как это влияет на поддерживаемость проекта. Примеры распространённых методов рефакторинга. Принципы UX-тестирования, его роль в улучшении проекта, методы проведения тестов с реальными пользователями. Как создать финальный проектный документ, описывающий весь процесс разработки. Как подготовить проект к выпуску на разных платформах. Особенности экспорта для ПК, мобильных устройств. Принципы подготовки презентации. Презентация как важный этап завершения проекта. Навыки эффективного представления своего продукта.

Практика: Обсуждение идей и направлений. Каждый ученик выбирает проект: игра, сайт или приложение. Написание цели проекта и ключевых задач. Исследование аналогов проектов.

Разработка схемы проекта. Создание базового прототипа. Подготовка необходимых инструментов. Реализация базового функционала. Разработка интерфейса. Добавление интерактивных элементов. Тестирование. Оптимизация. Работа с базой данных. Реализация сетевых функций. Работа с Git. Интеграция мультимедиа. Добавление статистики в проект. Работа с API. Настройка интеграции. Рефакторинг кода. Проведение UX-тестирования. Подготовка проекта к релизу. Написание плана релиза и списка изменений. Экспорт проекта в готовую игру или приложение. Настройка параметров экспорта для различных платформ. Подготовка презентации проекта для демонстрации: создание слайдов, описание функционала и ключевых моментов разработки. Проведение презентации перед аудиторией.

Формы контроля:

Составление технической документации

Итоговая презентация проекта

**КАЛЕНДАРНО-ТЕМАТИЧЕСКОЕ ПЛАНИРОВАНИЕ
2 ГОД ОБУЧЕНИЯ**

№ занятия	Наименование раздела, темы (теория и практика)	Содержание (теоретическая и Практика)	Дата проведения занятия		Количество часов			Формы контроля усвоения материала	Самостоятельная работа с использованием дистанционных образовательных технологий
			по плану	фактически	Теория	Практика	Всего		
	Модуль 1 Объектно-ориентированное программирование				10	14	24		
1	Вводное тестирование	Теория: обсуждение содержания курса Практика: прохождение учениками теста для определения остаточных знаний с предыдущего года обучения	05.09		0,5	1,5	2		

2	Повторение основных понятий ООП.	Теория: Пример из жизни: объекты и классы (сравнение с реальными вещами — машина, телефон). Что такое класс в программировании. Как создавать объекты на Python. Практика: Создание простого класса Car с атрибутами (цвет, марка). Создание объектов на основе класса.	06.09		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
3	Атрибуты и методы	Теория: Что такое атрибуты (свойства) и методы (действия) объектов. Примеры из жизни: телефон звонит, машина едет. Практика: Создание методов в классе Car, таких как <code>start_engine()</code> и <code>stop_engine()</code> .	12.09		1,5	0,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
4	Конструкторы объектов	Теория: Как создавать объекты с помощью конструктора <code>__init__()</code> . Значение свойств по умолчанию. Практика: Создание класса Book, который получает название и автора при создании объекта.	13.09		1	1	2	Выполнение задания «класс «Библиотека»»	Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
5	Пример: База данных учеников	Теория: Создание класса Student, его атрибутов (имя, возраст, класс) и методов (например, изменение класса или получение информации о студенте). Практика: Создание простого приложения, где можно добавлять, удалять и изменять данные учеников.	19.09		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП

6	Пример: Игра "Монополия "	Теория: Планирование проекта "Монополия". Описание классов: Cell (клетка поля), Player (игрок), Dice (кубик). Практика: Создание основных классов для игры, написание методов для генерации случайного числа для кубика и движения игрока по клеткам.	20.09		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
7	Продолжение разработки игры «Монополия»	Теория: Доступ к текущим значениям атрибутов, изменение значений атрибутов Практика: Добавление функционала в игру: покупка недвижимости, просмотр баланса и списка приобретенной ранее недвижимости	26.09		0	2	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
8	Пример: Текстовый квест на базе ООП	Теория: Как спроектировать текстовый квест с использованием ООП. Описание классов Location (место) и Choice (выбор). Практика: Создание простого квеста с несколькими локациями и выбором действий для игрока.	27.09		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
9	Продолжение разработки текстового квеста	Теория: Хранение ссылки на объект стороннего класса в качестве значения атрибута Практика: Создание класса Achievement, описывающего достижение в квесте, привязка достижений к действиям в квесте, хранение достижений в списке	03.10		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП

10	Пример: Моделирование движения маятника	Теория: Основы физического моделирования с использованием ООП. Описание класса Pendulum, который моделирует движение маятника. Практика: Написание простого симулятора движения маятника с использованием ООП.	04.10		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
11	Пример: Моделирование движения ракеты	Теория: Как моделировать движение объектов в пространстве. Описание классов Planet (планета) и Rocket (ракета). Практика: Написание программы, в которой ракета движется вокруг планеты или взлетает с её поверхности.	10.10		0,5	1,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
12	Наследование классов (основы)	Теория: Что такое наследование. Пример: класс Animal и классы-наследники Dog и Cat. Практика: Создание класса Animal с общими методами, а затем наследование для классов Dog и Cat с уникальными особенностями.	11.10		0,5	1,5	2	Выполнение задания текстовая RPG-игра	Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела ООП
2 Классические алгоритмы на Python					13	11	24		
13	Что такое алгоритмы и их важность	Теория: Определение алгоритма как последовательности шагов для выполнения задачи. Примеры алгоритмов в повседневной жизни. Важность алгоритмов в программировании. История развития алгоритмов. Алгоритмы в эпоху компьютеров.	17.10		2	0	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
14	Линейные алгоритмы	Теория: Что такое линейные алгоритмы и их основные характеристики. Алгоритмы, которые выполняются последовательно.	18.10		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение

		Практика: Написание алгоритмов для последовательного выполнения действий, например, нахождение максимального и минимального значения в списке.							домашнего задания из раздела Алгоритмы
15	Алгоритмы с циклами	Теория: Что такое поиск в алгоритмах. Линейный поиск и его применение. Практика: Написание программы для поиска элемента в списке с использованием линейного поиска.	24.10		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
16	Сортировка (сортировка выбором)	Теория: Введение в сортировку данных. Простой алгоритм сортировки выбором: как он работает и его особенности. Практика: Написание программы для сортировки списка с использованием сортировки выбором.	25.10		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
17	Сортировка (пузырьковая сортировка)	Теория: Алгоритм пузырьковой сортировки. Сравнение пузырьковой сортировки с сортировкой выбором. Практика: Реализация пузырьковой сортировки на Python. Сортировка списка по возрастанию и убыванию.	31.10		1	1	2	Выполнение задания «Сравнение алгоритмов»	Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
18	Бинарный поиск	Теория: Алгоритм бинарного поиска. Как бинарный поиск отличается от линейного и в каких случаях его использовать. Практика: Написание программы для поиска элемента в отсортированном списке с использованием бинарного поиска.	01.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы

19	Рекурсия (основы)	Теория: Что такое рекурсия. Примеры рекурсивных алгоритмов в жизни (например, русские матрёшки). Как работает рекурсия в программировании. Практика: Написание рекурсивного алгоритма для вычисления факториала числа и решения других простых задач.	07.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
20	Алгоритм Евклида для нахождения НОД	Теория: Что такое наибольший общий делитель (НОД) и как его можно найти. Описание алгоритма Евклида. Практика: Реализация алгоритма Евклида на Python для нахождения НОД двух чисел.	08.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
21	Жадные алгоритмы	Теория: Что такое жадные алгоритмы. Пример жадного алгоритма: задача о сдаче (наименьшее количество монет для данной суммы). Практика: Написание программы для решения задачи о сдаче с использованием жадного алгоритма.	14.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
22	Некоторые известные жадные алгоритмы	Теория: Классические примеры применения жадных алгоритмов – задача о ранце и задача выбора заявок Практика: написание кода для решения описанных задач	15.11		1	1	2	Выполнение задания «Минимальный набор монет»	Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
23	Алгоритмы поиска пути (алгоритм "Поиск в	Теория: Что такое поиск в ширину и для чего он используется. Пример использования для поиска кратчайшего пути.	21.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и

	ширину")	Практика: Написание программы для поиска пути в лабиринте с использованием алгоритма "Поиск в ширину".							выполнение домашнего задания из раздела Алгоритмы
24	Практическое задание: решение задач с применением алгоритмов	Теория: Обзор изученных алгоритмов: сортировка, поиск, рекурсия, жадные алгоритмы. Практика: Решение задач на применение различных алгоритмов. Создание программы, которая использует несколько алгоритмов для выполнения различных задач (например, сортировка списка с последующим поиском).	22.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Алгоритмы
3 Разработка игр на движке Godot					20	28	48		
25	Введение в Godot	Теория: Что такое Godot, его возможности, использование в разработке игр. Основные компоненты интерфейса. Практика: Установка Godot, знакомство с интерфейсом и создание первого проекта.	28.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
26	Сцены и узлы	Теория: Понятие сцены и узлов в Godot. Как они связаны между собой. Виды узлов. Практика: Создание простой сцены с несколькими узлами (например, персонаж и предметы).	29.11		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
27	Основы 3D трансформаций	Теория: Как работают трансформации в 3D: перемещение, вращение и масштабирование объектов. Принцип работы координатных систем. Практика: Написание скриптов для перемещения объектов в 3D	05.12		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев

		пространстве (например, вращение куба или перемещение шара).							
28	Работа с камерами и управление перспективой	Теория: Как работает камера в 3D. Виды камер и их настройка для различных перспектив. Практика: Добавление камеры на сцену, настройка её положения. Скрипт для динамического управления камерой, которая следует за игроком.	06.12		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
29	Основы скриптинга на GDScript (часть 1)	Теория: Введение в GDScript. Переменные, типы данных, условия и циклы. Практика: Написание простых скриптов для управления объектами на сцене (например, изменение цвета или размера объекта через код).	12.12		0,5	1,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
30	Основы скриптинга на GDScript (часть 2)	Теория: Функции, классы и методы в GDScript. Организация кода в игре. Практика: Написание скриптов для управления игровыми элементами, создание пользовательских функций.	13.12		1	1	2	Выполнение задания «Платформер»	Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
31	Сигналы и их использование в скриптах	Теория: Что такое сигналы в Godot и как они помогают взаимодействовать между объектами и сценами. Как подключать и отправлять сигналы между узлами. Практика: Написание скриптов для отправки сигналов при определённых событиях (например, персонаж подбирает предмет, враг обнаружил игрока). Подключение сигналов для взаимодействия между	19.12		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев

		разными объектами.							
32	Работа с массивами и словарями в GDScript	Теория: Введение в структуры данных GDScript: массивы и словари. Примеры их использования для хранения данных, таких как инвентарь или характеристики персонажей. Практика: Написание скриптов для управления инвентарём персонажа с использованием массива. Работа со словарём для хранения атрибутов персонажей или предметов (например, сила, здоровье, урон).	20.12		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
33	Таймеры и задержки в скриптах	Теория: Работа с таймерами в GDScript. Как создавать задержки выполнения действий и как это используется в игровых механиках (например, создание атак, паузы или задержки перед действиями). Практика: Создание таймеров в игре для выполнения действий с задержкой (например, атака врага через 3 секунды после его появления).	26.12		0,5	1,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
34	Создание и использование ресурсов (Resource)	Теория: Введение в ресурсы (Resource) в Godot. Как ресурсы помогают сохранять данные и настройки объектов вне основной сцены. Практика: Создание пользовательских ресурсов для хранения данных (например, характеристик персонажей, оружия или инвентаря). Работа с ресурсами через скрипты.	27.12						Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
35	Динамическая генерация	Теория: Как создавать объекты на сцене динамически через скрипты.	09.01		1	1	2		Изучение материалов на сайте

	объектов в игре	Использование метода instance() для клонирования объектов. Практика: Написание кода для генерации врагов, препятствий или других объектов в процессе игры. Пример: динамическое создание платформ в платформере или генерация астероидов в космической игре.							ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
36	Физика в 3D: столкновения	Теория: Введение в физику в 3D. Типы тел: статические, кинематические, динамические. Обработка столкновений. Практика: Добавление физики на сцену. Использование узлов RigidBody и CollisionShape для создания объектов, которые взаимодействуют друг с другом (например, падение объектов).	10.01		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
37	Моделирование гравитации и движения объектов	Теория: Как работает гравитация в 3D и как моделировать движение объектов. Практика: Написание скриптов для имитации гравитации и движения объектов. Пример: падение объектов под действием гравитации или движение шара по наклонной плоскости.	16.01		0,5	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
38	Анимация объектов в 3D	Теория: Основы создания анимации в 3D. Использование AnimationPlayer для создания анимаций. Практика: Создание анимации вращения, перемещения или изменения масштаба объекта. Управление анимацией через скрипты.	17.01		0,5	1,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев

39	Пользовательский интерфейс (UI) в 3D играх	Теория: Создание и настройка элементов интерфейса (жизни, очки, кнопки) для 3D игр. Взаимодействие между UI и 3D миром. Практика: Добавление интерфейса в игру, вывод очков и жизней на экран. Настройка кнопок для управления игрой.	23.01		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
40	Построение уровней	Теория: Как создавать игровые уровни в 3D. Понятие модульности и повторного использования объектов. Практика: Создание простого уровня с препятствиями и различными объектами.	24.01		0,5	1,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
41	Работа с освещением и тенями в 3D	Теория: Виды освещения в Godot: точечный, направленный свет, окружение. Настройка теней. Практика: Добавление света на сцену, настройка теней для реалистичной визуализации. Изменение освещения в зависимости от событий на сцене.	30.01		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
42	Управление персонажем	Теория: Скриптинг для управления персонажем. Реализация взаимодействий с окружением. Практика: Написание скриптов для перемещения персонажа в 3D, прыжков и взаимодействия с объектами.	31.01		0,5	1,5	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
43	Обработка столкновений для 3D персонажа	Теория: Как обрабатывать столкновения в 3D играх, используя кинематическое тело. Практика: Написание скриптов для обработки столкновений персонажа с объектами, препятствиями.	06.02		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания

									из раздела Геймдев
44	Моделирование физики в 3D (продвинутый уровень)	Теория: Углубленное изучение физики в 3D. Как работать с физическими взаимодействиями объектов. Практика: Написание сложных взаимодействий объектов, таких как столкновения с силой или катапультирование объектов.	07.02		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
45	Создание искусственного интеллекта для врагов	Теория: Основы создания ИИ для персонажей в играх. Патрулирование, преследование игрока. Практика: Написание скрипта для врага, который двигается по сцене, преследует игрока и реагирует на его действия.	13.02		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
46	Пути и траектории (Path and PathFollow3D)	Теория: Как создавать пути и траектории в 3D пространстве. Использование узла Path для задания пути движения объектов и узла PathFollow3D для управления их движением по траектории. Практика: Создание пути для движения объекта (например, врага или транспорта) по заданной траектории в 3D пространстве. Написание скриптов для контроля скорости и положения объекта на пути.	14.02		1	1	2	Выполнение задания «Лабиринт»	Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
47	Создание эффектов частиц (например, огня или дыма)	Теория: Как работают эффекты частиц в 3D. Настройка и оптимизация частиц. Практика: Создание эффекта частиц для визуализации огня, дыма, взрывов и других эффектов.	20.02		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев

48	Оптимизация игры	Теория: Как оптимизировать 3D игру для повышения производительности. Управление ресурсами и памятью. Практика: Оптимизация готового проекта (например, уменьшение нагрузки на процессор за счёт правильной настройки объектов и скриптов).	21.02		1	1	2		Изучение материалов на сайте ddtpython.pythonanywhere.com и выполнение домашнего задания из раздела Геймдев
4 Индивидуальные проекты					6	42	48		
49	Введение	Теория: Что такое индивидуальный проект и его значение в обучении. Примеры успешных проектов в разных направлениях (игры, веб-сайты, приложения). Практика: Обсуждение идей и направлений. Каждый ученик выбирает проект: игра, сайт или приложение.	27.02		0,25	1,75	2		
50	Постановка целей проекта	Теория: Принципы SMART (Specific, Measurable, Achievable, Relevant, Time-bound) для постановки целей. Цели для игр, сайтов и приложений. Практика: Написание цели проекта и ключевых задач.	28.02		0,25	1,75	2		
51	Анализ и исследование для проекта	Теория: Исследование аналогичных проектов. Анализ требований и технологий, которые помогут в реализации проекта. Практика: Исследование аналогов проектов: <ul style="list-style-type: none"> • Игра: анализ похожих игр, механик, интерфейсов. • Сайт: изучение современных дизайнов и функционала 	06.03		0,25	1,75	2		

		<p>сайтов.</p> <ul style="list-style-type: none"> • Приложение: исследование приложений, связанных с выбранной темой (например, приложения для учёта задач или бюджета). 							
52	Планирование структуры проекта	<p>Теория: Как спланировать структуру проекта (модульный подход). Разработка основных модулей. Практика: Разработка схемы проекта:</p> <ul style="list-style-type: none"> • Игра: структура уровней, персонажи, игровые механики. • Сайт: макет страниц, функционал и структура сайта. • Приложение: схема взаимодействий между модулями и интерфейсом. 	07.03		0,25	1,75	2		
53	Создание прототипов	<p>Теория: Важность прототипов для игры, сайта или приложения. Практика: Создание базового прототипа:</p> <ul style="list-style-type: none"> • Игра: прототип уровня с базовой механикой. • Сайт: прототип главной страницы с навигацией. • Приложение: набросок пользовательского интерфейса. 	13.03		0,25	1,75	2		
54	Выбор инструментов	Теория: Как выбрать правильные инструменты для проекта. Обзор	14.03		0,25	1,75	2		

	для разработки	<p>движков, фреймворков и библиотек для разных направлений.</p> <p>Практика: Подготовка необходимых инструментов:</p> <ul style="list-style-type: none"> • Игра: установка Godot или Pygame. • Сайт: формирование базового проекта на фреймворке(Django, Flask). • Выбор библиотеке для работы с графической оболочкой приложения (PyQT, Tkinter) 							
55	Начало разработки: базовый функционал	<p>Теория: Как построить минимально жизнеспособный продукт (MVP). Определение базовых задач.</p> <p>Практика: Реализация базового функционала:</p> <ul style="list-style-type: none"> • Игра: движение персонажа, взаимодействие с объектами. • Сайт: создание базовых страниц и навигации. • Приложение: реализация основного интерфейса и работы с данными. 	20.03		0,25	1,75	2		
56	Работа с интерфейсом пользователя	<p>Теория: Принципы проектирования удобного интерфейса для игр, сайтов и приложений.</p> <p>Практика: Разработка интерфейса:</p> <ul style="list-style-type: none"> • Игра: создание меню и интерфейса игры (очки, здоровье). • Сайт: создание контактных форм, меню навигации. • Приложение: интерфейс для 	21.03		0,25	1,75	2		

		ввода и отображения данных.							
57	Добавление интерактивности и функционала	<p>Теория: Как добавить интерактивность и дополнительные функции в проект.</p> <p>Практика: Добавление интерактивных элементов:</p> <ul style="list-style-type: none"> • Игра: добавление элементов взаимодействия (враги, бонусы). • Сайт: добавление форм обратной связи или интерактивных элементов. • Приложение: добавление логики обработки данных (например, расчёты или фильтры). 	27.03		0,25	1,75	2		
58	Обработка ошибок и тестирование	<p>Теория: Почему важно тестирование на каждом этапе. Основные виды тестирования.</p> <p>Практика: Тестирование:</p> <ul style="list-style-type: none"> • Игра: проверка взаимодействия объектов и физики. • Сайт: проверка ссылок, форм, работы на разных устройствах. • Приложение: тестирование правильности обработки данных и интерфейсов. 	28.03		0,25	1,75	2		
59	Оптимизация проекта	<p>Теория: Основные принципы оптимизации для игр, сайтов и приложений.</p> <p>Практика: Оптимизация:</p> <ul style="list-style-type: none"> • Игра: уменьшение нагрузки 	03.04		0,25	1,75	2		

		<p>на ресурсы, оптимизация графики.</p> <ul style="list-style-type: none"> • Сайт: ускорение загрузки страниц, сжатие изображений и кода. • Приложение: оптимизация работы с данными и повышение производительности. 							
60	Использование базы данных	<p>Теория: Введение в базы данных. Как сохранять и загружать данные в проекте.</p> <p>Практика: Работа с базой данных:</p> <ul style="list-style-type: none"> • Игра: создание системы сохранений. • Сайт: подключение базы данных для хранения пользователей и контента. • Приложение: сохранение данных пользователя в базе данных (например, SQLite). 	04.04		0,25	1,75	2		
61	Интеграция сетевых функций	<p>Теория: Основы работы с сетями в проекте. Примеры применения сетевых функций.</p> <p>Практика: Реализация сетевых функций:</p> <ul style="list-style-type: none"> • Игра: добавление мультиплеера. • Сайт: подключение к API или создание чата. • Приложение: синхронизация данных с сервером. 	10.04		0,25	1,75	2		
62	Использование контроля	<p>Теория: Что такое системы контроля версий и как они помогают в</p>	11.04		0,25	1,75	2		

	версий (Git)	<p>разработке.</p> <p>Практика: Работа с Git:</p> <ul style="list-style-type: none"> • Создание репозитория для проекта. • Совместная работа над кодом. • Управление изменениями и слияниями. 							
63	Работа с мультимедиа (графика, звук)	<p>Теория: Как мультимедиа (графика, звук) улучшает пользовательский опыт. Основы работы с изображениями, звуками и видео в проектах.</p> <p>Практика: Интеграция мультимедиа:</p> <ul style="list-style-type: none"> • Игра: добавление звуковых эффектов, музыки, фона. • Сайт: внедрение мультимедийных элементов (видео, аудио) на страницы. • Приложение: добавление звуковых оповещений и визуальных элементов (например, изображений). 	17.04		0,25	1,75	2		
64	Создание системы достижений или статистики для проекта	<p>Теория: Что такое система достижений и статистики, как её можно применять для мотивации и улучшения опыта пользователя.</p> <p>Практика: Добавление статистики в проект:</p> <ul style="list-style-type: none"> • Игра: создание системы достижений за выполненные уровни, собранные бонусы и т.д. • Сайт: ведение статистики посещаемости страниц или 	18.04		0,25	1,75	2		

		<p>количества оставленных комментариев.</p> <ul style="list-style-type: none"> • Приложение: учёт выполненных задач, прогресс в обучении, количество внесённых данных. 							
65	Работа с API для расширения функционала	<p>Теория: Что такое API и как оно используется для добавления внешних данных и функций в проект.</p> <p>Практика: Работа с API:</p> <ul style="list-style-type: none"> • Игра: подключение игрового API для рейтингов или мультиплеерной функциональности. • Сайт: добавление данных из внешних сервисов (например, погода, новости). • Приложение: интеграция с API для получения дополнительных данных (например, курсы валют или расписание мероприятий). 	24.04		0,25	1,75	2		
66	Интеграция с внешними сервисами (например, авторизация через социальные сети)	<p>Теория: Основы работы с внешними сервисами и их использование для улучшения проекта. Примеры авторизации через соцсети.</p> <p>Практика: Настройка интеграции:</p> <ul style="list-style-type: none"> • Игра: добавление сервиса, позволяющего игроку делиться результатами в социальных сетях. • Сайт: добавление авторизации через Google 	25.04		0,25	1,75	2		

		<p>или Facebook.</p> <ul style="list-style-type: none"> • Приложение: интеграция с внешним сервисом для обмена данными или авторизации пользователя. 							
67	Рефакторинг кода и улучшение структуры	<p>Теория: Основы рефакторинга: зачем упрощать и улучшать структуру кода, как это влияет на поддерживаемость проекта. Примеры распространённых методов рефакторинга.</p> <p>Практика: Рефакторинг кода:</p> <ul style="list-style-type: none"> • Игра: Упрощение сложных игровых функций, улучшение структуры скриптов. • Сайт: Рефакторинг кода страниц, упрощение CSS и JavaScript. • Приложение: Перенос повторяющегося кода в отдельные функции и оптимизация структуры проекта. 	02.05						
68	Тестирование удобства использования (UX-тестирование)	<p>Теория: Принципы UX-тестирования, его роль в улучшении проекта, методы проведения тестов с реальными пользователями.</p> <p>Практика: Проведение UX-тестирования:</p> <ul style="list-style-type: none"> • Игра: Анализ удобства управления и интерфейса. • Сайт: Тестирование удобства навигации и расположения элементов. 	08.05		0,25	1,75	2		

		<ul style="list-style-type: none"> Приложение: Проверка логичности интерфейса и удобства выполнения задач. 							
69	Проектная документация и планирование релиза	<p>Теория: Как создать финальный проектный документ, описывающий весь процесс разработки.</p> <p>Практика: Подготовка проекта к релизу. Написание плана релиза и списка изменений.</p>	15.05		0,25	1,75	2	Составление технической документации	
70	Подготовка проекта к выпуску (экспорт)	<p>Теория: Как подготовить проект к выпуску на разных платформах. Особенности экспорта для ПК, мобильных устройств.</p> <p>Практика: Экспорт проекта в готовую игру или приложение. Настройка параметров экспорта для различных платформ.</p>	16.05		0,25	1,75	2		
71	Презентация проекта (подготовка к защите)	<p>Теория: Принципы подготовки презентации.</p> <p>Практика: Подготовка презентации проекта для демонстрации: создание слайдов, описание функционала и ключевых моментов разработки.</p>	22.05		0,25	1,75	2		
72	Презентация и защита проекта	<p>Теория: Презентация как важный этап завершения проекта. Навыки эффективного представления своего продукта.</p> <p>Практика: Проведение презентации перед аудиторией:</p> <ul style="list-style-type: none"> Демонстрация работы проекта. Описание процесса разработки и ключевых решений. 	23.05		0,25	1,75	2	Итоговая презентация проекта	
	Всего				49	95	144		

МЕТОДИЧЕСКИЕ И ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

№ п\п	Тема программы	Форма занятий	Приемы и методы организации и проведения занятия	Дидактический материал, техническое оснащение занятий	Формы подведения итогов
1.	Введение в программирование	Занятие-лекция, занятие-беседа, занятие самостоятельных работ	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Инструкции по правилам безопасности и правилам поведения в ГБУ ДО ДДТ «На 9-ой линии». Программа. Персональный компьютер с доступом в интернет	Устная форма проверки знаний
2.	Основы программирования на Python	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ, занятие-игра	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code	Письменная форма проверки знаний
3.	Продвинутое программирование на Python	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ, семинар	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code	Письменная форма проверки знаний
4.	Разработка графических интерфейсов	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code	Зачёт
5,	Веб-программирование с Django	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual	Письменная форма проверки знаний

				Studio Code	
6.	Объектно-ориентированное программирование	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code	Письменная форма проверки знаний
7.	Классические алгоритмы на Python	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ, занятие-лабораторная работа, видеозанятие	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code	Письменная форма проверки знаний
8.	Разработка игр на движке Godot	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ, видеозанятие	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code	Тестирование разработанных игр
9.	Индивидуальные проекты	Семинар, занятие-инсценировка, занятие-беседа, занятие-диспут	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code.	Презентация итогового проекта учащимися
10.	Основы искусственного интеллекта и машинного обучения	Занятие-лекция, занятие-беседа, занятие самостоятельных работ, занятие практических работ, занятие-лабораторная работа, исследовательски й занятие	Словесный. Наглядный (иллюстрация , демонстрация). Практически й	Проектор, персональный компьютер с доступом в интернет, интегрированная среда разработки Pycharm или Visual Studio Code. Веб-камера, штатив	Оценка качества разработанной модели и обсуждение перспектив развития искусственного интеллекта и его влияние на общество

Оценочные результаты

Для отслеживания результативности образовательной деятельности по программе проводятся: входной и итоговый контроли.

Цель – проследить динамику развития и рост мастерства учащихся.

Входной контроль (первое занятие) проводится с целью выявления мотивации выбора творческого объединения и устойчивости интереса учащихся (*анкета для учащихся «Чем я хочу поделиться с миром?» (Приложение 1)*).

Итоговый контроль (проводится в конце обучения по программе). Формы контроля: презентация творческого проекта.

СИСТЕМА ОЦЕНИВАНИЯ ЗАДАЧ ПО ПРОГРАММИРОВАНИЮ

	Критерии	Оценка
1.	Код не сдан	0
2.	Код сдан, но не запускается	1
3.	Код запускается, но имеет логическую ошибку, отчего вывод неверный	2
4.	Вывод программы верный, но в программе содержится более трех стилистических ошибок	3
5.	Вывод программы верный, но в программе содержится от 1 до 3 стилистических ошибок	4
6.	Вывод программы верный, и код написан без стилистических ошибок	5

СИСТЕМА НАЧИСЛЕНИЯ БАЛЛОВ ПО РАЗДЕЛАМ ПРОГРАММЫ ДЛЯ НАЧИНАЮЩИХ

Задание	Баллы	
	Мин.	Макс.
Модуль 1. Объектно-ориентированное программирование		
Выполнение задания «класс “Библиотека”»	0	5
Выполнение задания «текстовая RPG-игра»	0	5
Модуль 2. Классические алгоритмы на Python		
Выполнение задания «Сравнение алгоритмов»	0	5
Выполнение задания «Минимальный набор монет»	0	5
Модуль 3. Разработка игр на движке Godot		
Выполнение задания «Платформер»	0	5
Выполнение задания «Лабиринт»	0	5
Модуль 4. Индивидуальные проекты		
Составление технической документации	0	5
Итоговая презентация проекта	0	5
ИТОГО	0	40

ИНФОРМАЦИОННЫЕ ИСТОЧНИКИ

Литература для обучающихся:

1. Доусон М. Програмируем на Python. - СПб.: Питер, 2014. - 416 с.
2. Лутц, М. Изучаем Python, том 1, 5-изд.: Пер.сангл.— СПб.: ООО “Диалектика”, 2019. — 832с.
3. Бхаргава, А. Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих. СПб.: Питер, 2018. — 288 с.

Литература для педагога:

1. Шень А. Программирование: теоремы и задачи. 6-е изд., дополненное. М.: МЦНМО, 2017. - 320 с.
2. Голицына О.Л., Партыка Т.Л., Попов И.И. Языки программирования. Учебное пособие. — 2-е изд., перераб. и доп. — М.: Форум, 2010. — 400 с.
3. Цветкова М. С., Великович Л. С. Информатика и ИКТ. М.: Академия,2012. – 352 с.
4. Андреева Е. В. Математические основы информатики. Элективный курс: Учебное пособие. М.: БИНОМ. Лаборатория знаний, 2005 – 328 с.
5. Верещагин Н. К., Шень А. Лекции по математической логике и теории алгоритмов. Часть 2. Языки и исчисления. — 4-е изд., испр. — М.: МЦНМО, 2012. — 240 с

Интернет источники:

1. Ddtpython – официальный сайт курса со всеми необходимыми материалами для обучения <https://ddtpython.pythonanywhere.com/>
2. Google Teachable Machine – инструмент для обучения школьников основам искусственного интеллекта <https://teachablemachine.withgoogle.com/>
3. Codeforces – платформа для олимпиадного программирования, основанная коллективом разработчиков из университета ИТМО <https://codeforces.com/>
4. GeeksForGeeks – портал с образовательными статьями по различным IT-направлениям <https://www.geeksforgeeks.org/>

ЗАДАНИЯ ДЛЯ ТЕКУЩЕГО И ПРОМЕЖУТОЧНОГО КОНТРОЛЯ (2й год обучения)

класс “Библиотека”

Разработать класс “Библиотека”, который хранит данные о книгах и позволяет добавлять, удалять и искать книги по названию и автору. Создать классы “Книга” и “Автор” с атрибутами и методами, например, для отображения информации о книге и авторе. Реализовать взаимодействие между объектами через методы класса “Библиотека”.

текстовая RPG-игра

Создать текстовую RPG-игру, где игрок управляет персонажем, обладающим разными навыками (например, воин или маг). Разработать классы “Персонаж” и подклассы “Воин” и “Маг” с разными атрибутами (здоровье, атака и т.д.) и методами для выполнения действий (например, “атака”, “лечение”).

Сравнение алгоритмов

Реализовать алгоритмы сортировки (пузырьковую и сортировку слиянием) для массива чисел и провести тестирование производительности для массивов разного размера (например, 100, 1000, 10,000 элементов). Вывести результаты и сравнить скорость работы алгоритмов путем построения графиков.

Минимальный набор монет

Написать программу, решающую задачу о сдаче с минимальным количеством монет. Реализовать алгоритм жадного подхода, используя массив с номиналами монет. Например, для суммы 67 рублей и набора монет {50, 10, 5, 1} программа должна определить минимальное количество монет, необходимых для получения заданной суммы.

Платформер

Создать игру в жанре платформера, где игрок управляет персонажем, который должен преодолевать препятствия и собирать предметы. Реализовать управление персонажем (движение, прыжок) и добавить базовые элементы игры, такие как таймер для отслеживания времени прохождения уровня и счетчик очков за собранные предметы.

Лабиринт

Создать 2D-игру "Лабиринт", в которой игрок управляет персонажем, старающимся найти выход из лабиринта.

Задачи проекта:

- Разработать простую карту лабиринта с препятствиями, используя узлы TileMap и узлы для статичных объектов.
- Настроить управление персонажем с помощью клавиш (W, A, S, D или стрелки) для перемещения по лабиринту.
- Добавить элемент "выход" из лабиринта и реализовать простую проверку на достижение выхода.

Дополнительные элементы (при желании):

- Добавить таймер для отсчета времени, за которое игрок должен найти выход.
- Включить простую анимацию для передвижения персонажа и эффект при достижении выхода (например, изменение цвета или вспышка экрана).

Алгебра логики

Создать модель, способную выполнять логические операции (например, AND, OR, NOT, XOR) на основе обучающих данных.

- Подготовить набор данных, содержащий логические выражения и их ожидаемые результаты.
- Обучить модель на этом наборе данных.
- Проверить модель, предоставив на вход новые логические выражения и проанализировав результаты ее работы.
- Сохранить код модели и добавьте примеры предсказаний, которые демонстрируют ее работу.

Рекомендательная система

Задание 1: Создать простую рекомендательную систему на базе модели k-Nearest Neighbors (k-NN) для рекомендаций фильмов или книг. Для этого:

- Загрузить один из общедоступных наборов данных, например, MovieLens (содержит рейтинги фильмов) или Book-Crossing (содержит рейтинги книг).

- Настроить модель k-NN для рекомендации объектов на основе схожести предпочтений пользователей.
- Подключить данные и настроить модель так, чтобы она рекомендовала похожие фильмы/книги, основываясь на предпочтениях выбранного пользователя.
- Сохранить результаты работы модели, включая несколько рекомендаций для конкретных пользователей.

Задание 2: Подготовить отчет по работе системы. В отчете:

- Описать процесс настройки модели k-NN и загрузки данных.
- Привести примеры рекомендованных системой фильмов/книг для нескольких пользователей и оценить качество этих рекомендаций.
- Добавить краткие предложения по улучшению модели, например, настройку параметра k (число ближайших соседей) для повышения точности рекомендаций.

ЗАДАНИЯ ДЛЯ ТЕКУЩЕГО И ПРОМЕЖУТОЧНОГО КОНТРОЛЯ (1й год обучения) ВВОДНОЕ ТЕСТИРОВАНИЕ

1. Напишите степени числа 2 от нулевой до десятой
2. Какие существуют операционные системы?
3. Напишите таблицу истинности для данных логических операций

Таблица истинности:

Логические переменные		Логические операции					
		отрицание	конъюнкция	дизъюнкция	исключающая дизъюнкция	импликация	эквиваленция
<i>A</i>	<i>B</i>	$\neg A$	$A \& B$	$A \vee B$	$A \oplus B$	$A \rightarrow B$	$A \leftrightarrow B$
0	0						
0	1						
1	0						
1	1						

4. Напишите программу, которая меняет местами две переменные целочисленного типа.
5. Напиши программу, на вход которой подается целое число n от 0 до 23, которое означает текущее время в часах. Программа должна выводить "Доброе утро", если введено больше 7 и до 12 включительно, "Добрый день", если больше 12 и до 18 включительно, "Доброй ночи", если больше 18 или до 7 включительно.
6. Напиши сортировку пузырьком на любом языке программирования. Массив можно задать в самой программе
7. Что такое факториал?
8. Напиши программу, на которая считает факториал поданного на вход числа n
9. Представь, что перед вами на столе лежат яблоко, груша и банан в таком порядке. Сколькими способами их можно переставить?
10. Сколькими способами можно выбрать: а) один фрукт, б) два фрукта, в) хотя бы один фрукт?

ТЕСТИРОВАНИЕ НА ЗНАНИЕ ОСНОВ ПРОГРАММИРОВАНИЯ НА PYTHON ДЛЯ НАЧИНАЮЩИХ

1. Что выведет программа:

```
nameList = ['Harsh', 'Pratik', 'Bob', 'Dhruv']
print nameList[1][-1]
```

Ответ: k

2. Какая из следующих функций переводит строку в число с плавающей точкой?

- int(x [,base])
- long(x [,base])

- float(x)

- str(x)

Ответ: float(x)

3. Что выведет программа:

```
for i in range(2):  
    print i
```

```
for i in range(4,6):
```

```
    print i
```

Ответ:

0

1

4

5

4. Что выведет этот код:

```
print 9//2
```

- 4.5

- 4.0

- 4

- Ошибку

Ответ: 4

5. Что выведет этот код:

```
i = 0
```

```
while i < 3:
```

```
    print i
```

```
    i++
```

```
    print i+1
```

- 0 2 1 3 2 4

- 0 1 2 3 4 5

- Ошибку

- 1 0 2 4 3 5

Ответ: Ошибку

Задача 1. Напишите программу, которая принимает на вход трехзначное число и выводит сумму его цифр.

Задача 2. Даны значения двух моментов времени, принадлежащих одним и тем же суткам: часы, минуты и секунды для каждого из моментов времени. Известно, что второй момент времени наступил не раньше первого. Определите, сколько секунд прошло между двумя моментами времени. Программа на вход получает три целых числа — часы, минуты, секунды, задающие первый момент времени и три целых числа, задающих второй момент времени.

ТЕСТИРОВАНИЕ НА ЗНАНИЕ ПРОДВИНУТЫХ ПРОГРАММ НА PYTHON

1. Что выведет следующий код:

```
def REVERSE(L):
```

```
    L.reverse()
```

```
    return(L)
```

```
def YKNJS(L):
```

```
    List = []
```

```
    List.extend(REVERSE(L))
```

```
    print(List)
```

```
L = [1, 3.1, 5.31, 7.531]
```

```
YKNJS(L)
```

- [1, 3.1, 5.31, 7.531]

- [7.531, 5.31, 3.1, 1]
- IndexError
- AttributeError: 'NoneType' object has no attribute 'REVERSE'

Ответ: [7.531, 5.31, 3.1, 1]

2. Что выведет следующий код:

```
from math import sqrt
L1 = [x**2 for x in range(10)].pop()
L1 += 19
print(sqrt(L1), end = " ")
L1 = [x**2 for x in reversed(range(10))].pop()
L1 += 16
print(int(sqrt(L1)))
```

10.0 4.0

4.3588 4

10.0 4

10.0 0

Ответ: 10.0 4

3. Что выведет следующий код:

```
T = (1, 2, 3, 4, 5, 6, 7, 8)
print(T[T.index(5)], end = " ")
print(T[T[T[6]-3]-6])
```

- 4 0

- 5 8

- 5 IndexError

- 4 1

Ответ: 5 8

4. Что выведет следующий код:

```
dictionary = {'GFG' : 'geeksforgeeks.org',
             'google' : 'google.com',
             'facebook' : 'facebook.com'
            }
del dictionary['google'];
for key, values in dictionary.items():
    print(key)
dictionary.clear();
for key, values in dictionary.items():
    print(key)
del dictionary;
for key, values in dictionary.items():
    print(key)
```

- Runtime error

- GFG

facebook

- facebook

GFG

Ответ: facebook

GFG

5. Что выведет следующий код:

```
L1 = [1, 2, 3, 4]
```

```
L2 = L1
```

```
L3 = L1.copy()
```

```
L4 = L1
```

```
L1[0] = [5]
print(L1, L2, L3, L4)
```

```
- [5, 2, 3, 4] [5, 2, 3, 4] [1, 2, 3, 4] [1, 2, 3, 4]
- [[5], 2, 3, 4] [[5], 2, 3, 4] [[5], 2, 3, 4] [1, 2, 3, 4]
- [5, 2, 3, 4] [5, 2, 3, 4] [5, 2, 3, 4] [1, 2, 3, 4]
- [[5], 2, 3, 4] [[5], 2, 3, 4] [1, 2, 3, 4] [1, 2, 3, 4]
Ответ: [[5], 2, 3, 4] [[5], 2, 3, 4] [1, 2, 3, 4] [1, 2, 3, 4]
```

Задача 1. Задачу можно решить с помощью циклов, однако, рекомендуем попробовать решить их с помощью рекурсивных функций. Известно, что любой цикл можно заменить рекурсией. Дана строка, содержащая только десятичные цифры. Найти и вывести наибольшую цифру. **Входные данные:** Вводится строка ненулевой длины. Известно также, что длина строки не превышает 1000 знаков и строка содержит только десятичные цифры. **Выходные данные:** Выведите максимальную цифру, которая встречается во введенной строке.

Задача 2. Даны два числа. Найти их наибольший общий делитель. **Входные данные:** Вводятся два натуральных числа, не превышающих 109. **Выходные данные:** Выведите НОД введенных чисел.

ГОТОВАЯ ПРОГРАММА БЕСПОЛЕЗНЫЕ ФАКТЫ

Из пользовательского ввода программа получает информацию об имени, возрасте пользователя и массе его тела. На основе этих нехитрых данных программа способна сгенерировать несколько забавных, но совершенно бесполезных фактов о пользователе: например, сколько тот будет весить на Луне.

ГОТОВАЯ ПРОГРАММА ОТГАДАЙ ЧИСЛО

Компьютер выбирает случайное число от 1 до 100, а игрок должен его отгадать за минимальное количество попыток. При каждой очередной попытке компьютер говорит игроку, как соотносится предложенный вариант с действительно загаданным числом: первое больше, второе больше или они совпадают. Как только игрок отгадывает число, игра заканчивается.

ГОТОВАЯ ПРОГРАММА АНАГРАММЫ

Эта игра - воссоздание типичной головоломки на перестановку букв, одной из тех, какие в прошлом веке можно было встретить в воскресных газетах (тех самых, которые публика читала, пока не появился Интернет). Компьютер случайным образом выбирает из группы слов одно, переставляет его буквы тоже в случайном порядке и предъявляет игроку. Задача человека - восстановить исходное слово.

ГОТОВАЯ ПРОГРАММА КРЕСТИКИ-НОЛИКИ

Данный проект покажет, как, применяя на практике некоторые базовые идеи искусственного интеллекта (ИИ), создать виртуального противника. Компьютер будет противостоять игроку-человеку в захватывающем интеллектуальном шоу Крестики-нолики. Пользователю суждено столкнуться с грозным (хотя и не совсем безупречным) соперником, который к тому же очень самоуверен.

ГОТОВАЯ ПРОГРАММА ВИКТОРИНА

Игра Викторина проверяет знания игрока, задавая ему вопросы на выбор ответа из нескольких вариантов. Каждая игра представляет собой - «эпизод», относительно единый по тематике. Сильная сторона программы в том, что вопросы игры не - «защиты» в код, а хранятся в отдельном файле. Содержимое этого игрового файла легко изменить. Более того, с помощью простейшего текстового редактора, например, Блокнота в Windows, вы можете создавать собственные эпизоды (игры) на любые темы - от зоологии до японских мультфильмов.

ГОТОВАЯ ПРОГРАММА СУМАСШЕДШИЙ СКАЗОЧНИК

Программа Сумасшедший сказочник просит пользователя помочь в составлении назидательного рассказа. Пользователь должен ввести имя человека, существительное во множественном числе и инфинитив глагола. Предоставлена также возможность указать одно или несколько прилагательных и выбрать одну часть тела. Получив все эти данные, программа создает рассказ. Сумасшедший сказочник взаимодействует с пользователем через GUI.

ГОТОВАЯ ПРОГРАММА ПАНИКА В ПИЦЦЕРИИ

По сюжету игры в одной пиццерии случилась неприятность: шеф-повара до последнего градуса ярости довел наплыв привередливых посетителей, и он взобрался на крышу здания и стал оттуда разбрасывать готовую пиццу. Конечно, нельзя допустить такого перевода продуктов: пицца должна

быть спасена. С помощью мыши игрок может передвигать по экрану глубокую сковородку и с ее помощью ловить падающую пиццу. С каждой следующей пойманной пиццей количество очков на счету у игрока увеличивается, но как только пицца касается земли, игра заканчивается.

ГОТОВАЯ ПРОГРАММА ПРЕРВАННЫЙ ПОЛЕТ

Проект - упрощенная версия классической аркады Астероиды. Пользователь, играя в Прерванный полет, управляет космическим кораблем, который движется сквозь пояс смертоносных каменных астероидов. Корабль может вращаться и линейно ускоряться, а кроме того, он вооружен ракетами. Ракетный удар по астероиду разрушает его. Впрочем, это не снимает разом всех проблем, потому что астероиды крупного и среднего размера, взрываясь, дробятся на две части. Как только игрок расправится со всеми астероидами, на корабль налетит новая, еще более мощная волна этих космических убийц. Счет игрока увеличивается с каждым разрушенным астероидом, но как только звездолет столкнется с одним из них, игре наступит конец.

ТЕСТИРОВАНИЕ НА ЗНАНИЕ ОСНОВ ПРОГРАММИРОВАНИЯ НА PYTHON ДЛЯ ПРОДОЛЖАЮЩИХ

1. Что выведет следующая программа?

```
x = ['ab', 'cd']
for i in x:
    i.upper()
print(x)
```

Ответ: ['ab', 'cd']

2. Что выведет следующая программа?

```
for i in [1, 2, 3, 4][::-1]:
    print(i)
```

Ответ:

```
4
3
2
1
```

3. Что выведет следующая программа?

```
print(not(4>3))
print(not(5&5))
```

- False, False
- None, None
- True, True
- True, False

Ответ: False, False

4. Что выведет следующая программа?

```
data = [2, 3, 9]
temp = [[x for x in data] for x in range(3)]
print(temp)
```

- [[[2, 3, 9]], [[2, 3, 9]], [[2, 3, 9]]]
- [[2, 3, 9], [2, 3, 9], [2, 3, 9]]
- [[[2, 3, 9]], [[2, 3, 9]]]

Ответ: [[[2, 3, 9]], [[2, 3, 9]], [[2, 3, 9]]]

5. Что выведет следующая программа?

```
i = 1
while True:
    if i % 3 == 0:
        break
    print(i)
    i += 1
```

- 1 2 3

- 1 2

- Syntax Error

Ответ: Syntax Error

Задача 1. Вклад в банке составляет x рублей. Ежегодно он увеличивается на p процентов, после чего дробная часть копеек отбрасывается. Каждый год сумма вклада становится больше. Определите, через сколько лет вклад составит не менее y рублей.

Входные данные: Программа получает на вход три натуральных числа: x , p , y . **Выходные данные:** Программа должна вывести одно целое число.

Задача 2. Последовательность Фибоначчи определяется так:

$$\varphi_0 = 0, \varphi_1 = 1, \dots, \varphi_n = \varphi_{n-1} + \varphi_{n-2}.$$

По данному числу n определите n -е число Фибоначчи φ_n .

Входные данные: Вводится натуральное число n . **Выходные данные:** Выведите ответ на задачу.

ТЕСТИРОВАНИЕ НА ЗНАНИЕ ПРОДВИНУТЫХ ВОЗМОЖНОСТЕЙ PYTHON ДЛЯ ПРОДОЛЖАЮЩИХ

1. Что выведет следующая программа?

```
class A(object):
    val = 1

class B(A):
    pass

class C(A):
    pass

print (A.val, B.val, C.val)
B.val = 2
print (A.val, B.val, C.val)
A.val = 3
print (A.val, B.val, C.val)
```

Ответ:

1 1 1

1 2 1

3 2 3

2. Что выведет следующая программа?

```
check1 = ['Learn', 'Quiz', 'Practice', 'Contribute']
check2 = check1
check3 = check1[:]
```

```
check2[0] = 'Code'
```

```
check3[1] = 'Mcq'
```

```
count = 0
```

```
for c in (check1, check2, check3):
```

```
    if c[0] == 'Code':
```

```
        count += 1
```

```
if c[1] == 'Mcq':  
    count += 10
```

```
print (count)
```

Ответ:

12

3. Что выведет следующая программа?

```
data = 50
```

```
try:
```

```
    data = data/10
```

```
except ZeroDivisionError:
```

```
    print('Cannot divide by 0 ', end = "")
```

```
finally:
```

```
    print('GeeksforGeeks ', end = "")
```

```
else:
```

```
    print('Division successful ', end = "")
```

- Runtime error
- Cannot divide by 0 GeeksforGeeks
- GeeksforGeeks Division successful
- GeeksforGeeks

Ответ: Runtime error

4. Что выведет следующая программа?

```
D = {1 : [1, 2, 3], 2: (4, 6, 8)}
```

```
D[1].append(4)
```

```
print(D[1], end = " ")
```

```
L = [D[2]]
```

```
L.append(10)
```

```
D[2] = tuple(L)
```

```
print(D[2])
```

- [1, 2, 3, 4] [4, 6, 8, 10]
- [1, 2, 3, 4] ((4, 6, 8), 10)
- '[1, 2, 3, 4] TypeError: tuples are immutable
- [1, 2, 3, 4] (4, 6, 8, 10)

Ответ: [1, 2, 3, 4] ((4, 6, 8), 10)

5. Что выведет следующая программа?

```
y = 8
```

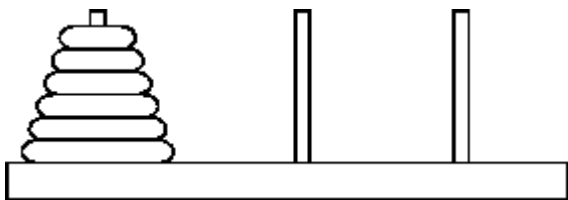
```
z = lambda x : x * y
```

```
print (z(6))
```

- 48
- 14
- 64

Ответ: 48

Задача 1. Головоломка “Ханойские башни” состоит из трех стержней, пронумерованных числами 1, 2, 3. На стержень 1 надета пирамидка из n дисков различного диаметра в порядке возрастания диаметра. Диски можно перекладывать с одного стержня на другой по одному, при этом диск нельзя класть на диск меньшего диаметра. Необходимо переложить всю пирамидку со стержня 1 на стержень 3 за минимальное число перекладываний.



Напишите программу, которая решает головоломку; для данного числа дисков n печатает последовательность переключений в формате $a\ b\ c$, где a — номер перекладываемого диска, b — номер стержня с которого снимается данный диск, c — номер стержня на который надевается данный диск.

Например, строка $1\ 2\ 3$ означает перемещение диска номер 1 со стержня 2 на стержень 3. В одной строке печатается одна команда. Диски пронумерованы числами от 1 до n в порядке возрастания диаметров.

Входные данные: Вводится натуральное число n . **Выходные данные:** Программа должна вывести минимальный (по количеству произведенных операций) способ перекладывания пирамидки из данного числа дисков.

Задача 2. Постановлением ЮНЕСКО оригинал Ханойской башни был подвергнут реставрации. В связи с этим во время пользования головоломкой нельзя было перекладывать кольца с первого стержня сразу на третий и наоборот.

Решите головоломку (переложите все кольца с первого стержня на третий) с учетом этих ограничений. Вам не нужно находить минимальное решение, но количество совершенных перемещений не должно быть больше 200000, при условии, что количество дисков не превосходит 10.

Каждое перемещение задается тремя числами: номер кольца, исходный стержень, конечный стержень.

Входные данные: Вводится натуральное число n . **Выходные данные:** Выведите ответ на задачу.

ГОТОВАЯ ПРОГРАММА ХУДОЖНИК

Известный художник решил написать новый шедевр. После многих дней усердной работы он захотел исследовать свое творение. Художник вспомнил, что картина писалась следующим образом: сначала был взят белый холст, имеющий форму прямоугольника шириной w и высотой h . Затем художник нарисовал на этом холсте n прямоугольников со сторонами, параллельными сторонам холста и вершинами, расположенными в целочисленных координатах. Помогите художнику определить площадь незакрашенной части холста.

Входные данные

Первая строка входа содержит два натуральных числа w и h ($1 \leq w, h \leq 100$). Во второй строке записано целое число n ($0 \leq n \leq 5000$) — количество прямоугольников. Следующие n строк содержат информацию о всех прямоугольниках. Каждая строка описывает один прямоугольник в виде четырех чисел x_1, y_1, x_2, y_2 , где (x_1, y_1) и (x_2, y_2) — координаты левого верхнего и правого нижнего угла прямоугольника соответственно. Пример:

```
5 5
2
1 1 3 3
2 2 4 4
```

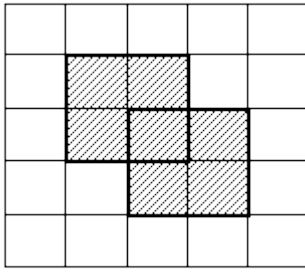
Выходные данные

Выведите одно целое число — площадь незакрашенной части холста. Пример:

```
18
```

Пояснение к первому примеру:

(0,0)



(5,5)

ГОТОВАЯ ПРОГРАММА ПОСТУЛАТ БЕРТРАНА

Постулат Бертрانا (теорема Бертрана-Чебышева, теорема Чебышева) гласит, что для любого $n > 1$ найдется простое число p в интервале $n < p < 2n$. Такая гипотеза была выдвинута в 1845 году французским математиком Джозефом Бертраном (проверившим ее до $n=3000000$) и доказана в 1850 году Пафнутием Чебышевым. Раманужан в 1920 году нашел более простое доказательство, а Эрдеш в 1932 – еще более простое.

Ваша задача состоит в том, чтобы решить несколько более общую задачу – а именно по числу n найти количество простых чисел p из интервала $n < p < 2n$. Напомним, что число называется простым, если оно делится только само на себя и на единицу.

Входные данные

Вход содержит целое число n ($2 \leq n \leq 50000$). Пример:

239

Выходные данные

Выведите одно число – ответ на задачу. Пример:

39

ГОТОВАЯ ПРОГРАММА КРЕСТИКИ-НОЛИКИ

Данный проект покажет, как, применяя на практике некоторые базовые идеи искусственного интеллекта (ИИ), создать виртуального противника. Компьютер будет противостоять игроку-человеку в захватывающем интеллектуальном шоу Крестики-нолики. Пользователю суждено столкнуться с грозным (хотя и не совсем безупречным) соперником, который к тому же очень самоуверен.

ГОТОВАЯ ПРОГРАММА БЛЕК-ДЖЕК

Проект представляет собой упрощенную версию карточной игры - Блек-джек. Игровой процесс идет так: участники получают карты, с которыми связаны определенные числовые значения - очки, и каждый участник стремится набрать 21 очко, но не больше. Количество очков, соответствующих карте с цифрой, равно ее номиналу; валет, дама и король идут за 10 очков, а туз - за 1 или 11 (в зависимости от того, как выгоднее для игрока). Компьютер сдает карты и играет против нескольких игроков, от одного до семи. В начале каждого раунда компьютер передает каждому из участников, в том числе и себе, по две карты. Игрокам видны карты друг друга, и даже автоматически подсчитывается текущая сумма очков на руках у каждого. Впрочем, из двух карт, которые дилер сдал себе, одна до поры до времени лежит рубашкой вверх.

Затем каждому игроку по очереди предоставляется возможность тянуть дополнительные карты. Игрок может брать их из перемешанной колоды до тех пор, пока ему угодно и пока сумма очков на руках у него не превысила 21. При превышении, которое называется перебором, участник проигрывает. Если все перебрали, то компьютер выводит свою вторую карту и начинает новый раунд. Если же один или несколько участников остались в игре, то раунд еще не закончен. Дилер открывает свою вторую карту и, по общему правилу Блек-джека, тянет дополнительные карты для себя до тех пор, пока сумма его очков не будет равна 17 или больше. Если дилер, в нашем случае - компьютер, совершает перебор, то победу одерживают все участники, оставшиеся в игре. Если нет, то сумма очков каждого из участников сравнивается с очками, которые набрал компьютер. Набравший больше очков (человек или компьютер) побеждает. При одинаковой сумме очков объявляется ничья между компьютером и одним или несколькими участниками.

ГОТОВАЯ ПРОГРАММА ЛАБИРИНТ

Используя материалы по библиотеке ругаме, создайте игру Лабиринт. Желательный размер лабиринта – 50 x 50 клеток (можно выбрать свой размер, но не менее 20x20), пользователь может

перемещаться вверх, вниз, вправо и влево. Каждый шаг смещает пользователя на одну клетку, если не мешает стена. Вход и выход из лабиринта генерируются случайно при каждом запуске игры. При прохождении лабиринта пользователю выдается информация о затраченном количестве шагов, времени прохождения, предложение завершить игру и начать заново.

Далее напишите программу, которая будет автоматически передвигать пользователя и искать выход из лабиринта. При прохождении лабиринта выдается информация о количестве шагов. (Желательно придумать свой менее эффективный алгоритм, чем правило правой руки).

ГОТОВАЯ ПРОГРАММА ПАРСЕР ДЛЯ СОЦИАЛЬНЫХ СЕТЕЙ

Используя документацию API социальной сети ВК и материалы занятий по библиотекам Python, написать веб-приложение с интерфейсом для выбора пользователей из социальной сети. Приложение должно иметь две функции:

- Вывод списка подписчиков группы.
- Вывод списка общих подписчиков двух групп.

Списки id пользователей должны сохраняться в текстовые файлы. Каждый id должен располагаться на отдельной строке