

Индивидуальный предприниматель Селендеева О.Н.

УТВЕРЖДАЮ

Индивидуальный предприниматель

_____/Селендеева О.Н.//

**ДОПОЛНИТЕЛЬНАЯ ОБЩЕОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА
ДОПОЛНИТЕЛЬНАЯ ОБЩЕРАЗВИВАЮЩАЯ ПРОГРАММА
«ПРОГРАММИРОВАНИЕ ИГР НА PYTHON»**

Москва, 2023

Оглавление

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА	3
1.1. Общая характеристика программы	3
1.2. Цели и задачи программы	4
1.3. Планируемые результаты обучения	5
2. УЧЕБНЫЙ ПЛАН ПРОГРАММЫ	6
3. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК	8
4. РАБОЧАЯ ПРОГРАММА	1
5. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ	8
5.1. Контроль знаний, умений и навыков	8
5.2. Критерии оценивания освоения программы при проведении различных форм контроля:	9
6. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ (УЧЕБНИКИ, РАЗДАТОЧНЫЕ МАТЕРИАЛЫ, МЕТОДИЧЕСКИЕ УКАЗАНИЯ, ПЛАКАТЫ, СЛАЙДЫ, ИНТЕРНЕТ-РЕСУРСЫ)	10
6.1. Информационные и учебно-методические условия реализации программы	10
6.2. Рекомендованная литература для обучающихся	10
6.3. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", рекомендованных для освоения программы	11
7. ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИЕ УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ	12
7.1. Материально-техническая и ресурсная база	12
7.2. Кадровое обеспечение программы	12
ПРИЛОЖЕНИЯ	13
Приложение 1. Примерные вопросы для промежуточного тестирования	13
Приложение 2. Примерные задания для оценки качества освоения учебного материала	19

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

1.1. Общая характеристика программы

Данный документ описывает комплекс основных характеристик образования (объем, содержание, планируемые результаты) и организационно-педагогических условий, который представлен в виде учебного плана, календарного учебного графика, рабочих программ учебных предметов, модулей, иных компонентов, а также оценочных и методических материалов общеобразовательной общеразвивающей программы "Программирование игр на Python".

В ходе дополнительной общеобразовательной общеразвивающей программы "Программирование игр на Python" обучающиеся научатся самостоятельно придумывать игры и реализовывать их на языке программирования Python, овладеют навыками создания сюжета, дизайна и механик компьютерных игр, научатся создавать персонажей, смогут использовать подходы, принципы и технологии создания современных текстовых, двумерных и трёхмерных игр с использованием языка программирования Python и библиотеки pygame.

Выдача обучающимся документов о дополнительном образовании (сертификат установленного образца) осуществляется при условии успешного прохождения итоговой аттестации.

Программа разработана на основе следующих **нормативных документов**:

- Федеральный закон от 29 декабря 2012 г. №273-ФЗ «Об образовании в Российской Федерации»;
- Приказ Министерства просвещения РФ от 9 ноября 2018 г. N 196 "Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам"
- Постановление Главного государственного санитарного врача РФ от 28.09.2020 № 28 «Об утверждении санитарных правил СП 2.4.3648-20 "Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи" (вместе с "СП 2.4.3648-20. Санитарные правила...")» (Зарегистрировано в Минюсте России 18.12.2020 № 61573).

Направленность (профиль) программы: техническая.

Актуальность программы, соответствие государственному, социальному заказу/запросам.

В современном мире умение программировать является **ценным и востребованным навыком**, который все больше и больше ценится на рынке, особенно сегодня, в условиях ограниченных ресурсов и повышенных расходов. В результате прохождения обучения обучающиеся сформируют базу знаний в области программирования на языке Python, активизируют навыки использования полученных знаний и умений в практической деятельности, разовьют логическое мышление, проявят интерес к программированию, раскроют свои способности в сфере разработки игр. Обучающимся будут созданы оптимальные условия для всестороннего удовлетворения потребностей и развития их индивидуальных склонностей и способностей, появится мотивация личности к познанию и

техническому творчеству. Полученные на данной программе навыки помогут сделать первые шаги в таких востребованных профессиях, как программист, геймдизайнер, разработчик игр.

Отличительные особенности программы: по окончании обучения учащиеся с помощью полученных на курсе знаний и навыков создадут несколько собственных 2D и 3D-игр, создадут портфолио из них. На занятиях ребенок погружается в мир IT, осваивает азы программирования через игру. Обучение проходит в обычном редакторе для языка программирования Python, и начинается с самых базовых понятий, поэтому курс подходит для ребят, не имеющих опыта в программировании.

Срок обучения: программа реализуется в объеме 72 академических часа, 36 недель (9 месяцев).

Режим занятий: 2-4 академических часа в неделю

Продолжительность академического часа – 45 минут.

Занятия начинаются не ранее 9.00 часов утра и заканчиваются не позднее 20.00 часов. Для обучающихся в возрасте 16-18 лет допускается окончание занятий в 21.00 часов.

Продолжительность занятий в учебные дни - не более 3-х академических часов в день, в выходные и каникулярные дни - не более 4 академических часов в день. После 30-45 минут теоретических занятий организуется перерыв длительностью не менее 10 мин.

Адресат программы и примерный портрет слушателя курсов: программа разработана для учащихся от 11 до 14 лет, которых интересуют современные технологии создания компьютерных игр. К освоению дополнительной общеобразовательной программы – дополнительной общеразвивающей программы допускаются: лица без предъявления требований к уровню образования.

По завершении реализации программы, как правило, проводится анкетирование обучающихся с целью изучения мнения по вопросу эффективности и информативности проведенного обучения, уровню организации учебного процесса, удовлетворенности учебно-методическим материалом, работниками образовательной организации проводится анализ высказанных предложений и пожеланий.

1.2. Цели и задачи программы

Цель программы – получение обучающимися знаний и навыков по разработке программного обеспечения, получение визуального представления о программировании игр, знакомство с фундаментальными структурами языков программирования, получение навыков создания сюжета, дизайна и механик игры, создание собственных текстовых, 2D или 3D игр на Python.

Задачи программы:

1. Сформировать системные знания в области компьютерных технологий и разработки игр и программ.
2. Предоставить специализированные практические навыки от основ программирования до применения подходов и технологий создания современных игр.
3. Научить создавать собственные игры на языке Python, с использованием библиотеки pygame.
4. Дать представление о профессии разработчика игр.

5. Сформировать навыки работы над проектами, развивая творческий подход, а также навыки тайм-менеджмента, решения задач и принятия решений.
6. Научить публиковать свои игры в интернет.
7. Сформировать интерес к увлечению программированием и раскрытию своих способностей в сфере IT-технологий.

1.3. Планируемые результаты обучения

По итогам освоения дополнительной общеобразовательной программы - дополнительной общеразвивающей программы "Программирование игр на Python" обучающиеся должны будут овладеть следующими знаниями, умениями и навыками:

Знать:

- основы работы с языком Python и его библиотекой pygame
- базовый синтаксис языка программирования Python
- правила и технику работы с переменными и условиями в коде программы
- использованием алгоритмов, 2D и 3D графикой, пользовательским интерфейсом
- принципы и технологию создания современных текстовых, двумерных и трёхмерных игр.

Уметь:

- самостоятельно работать над созданием игры на языке Python и применять библиотеку pygame
- использовать в работе принципы написания кода и создания алгоритмов
- применять творческий подход к реализации задач в игре
- создавать игровых и неигровых персонажей
- добавлять в игру модели объектов, работать с текстурами
- создавать активные компоненты игры, добавлять пользовательский интерфейс
- настраивать диалоги с персонажами
- писать скрипты на языке Python и настраивать их взаимодействие
- создавать свои игры, включая подготовку, сборку и тестирование
- работать на результат
- публиковать созданные игры в интернете
- создавать и презентовать свои проекты

Владеть навыками в области:

- планирования, алгоритмического и пространственного мышления
- создания сюжета, дизайна и механик игры,
- реализации концепции уровня игры
- автоматизации и алгоритмизации игровых процессов
- синхронизации игровых действий и звуков
- разработки компьютерных игр
- поиска ошибок и улучшения написанного кода
- тайм-менеджмента, постановки и решения задач и принятия решений
- презентации проектов и портфолио

2. УЧЕБНЫЙ ПЛАН ПРОГРАММЫ

В процессе преподавания курса "Программирование игр на Python" используются как классические методы обучения (лекции), так и различные виды практической работы обучающихся по заданию преподавателя, которые направлены на развитие навыков создания простых 2D и 3D игр, креативных качеств и на поощрение интеллектуальных инициатив учащихся.

Формы организации образовательного процесса (индивидуальные, групповые и т.д.) и другие виды занятий по программе определяются содержанием программы. Образовательная деятельность обучающихся предусматривает следующие виды учебных занятий и учебных работ: лекции, практические и семинарские занятия, круглые столы, мастер-классы, тренинги, семинары по обмену опытом, проведение открытых занятий, консультации и другие виды учебных занятий и методической работы, определенные учебным планом.

№	Наименование дисциплины/раздела/темы	Количество академических часов				Форма аттестации /контроля
		Всего	в т.ч. аудиторных		СРС	
			теория	практич. занятия		
1	Знакомство с Python	2	1	1	0	Практическое задание
2	Операторы в Python	2	0,5	1	0,5	Практическое задание
3	Условные операторы	2	0,5	0,5	1	Практическое задание
4	Работа со строками	2	0,5	1	0,5	Практическое задание. Контрольные вопросы
5	Циклы в Python	2	0,5	0,5	1	Практическое задание
6	Массивы в Python	2	1	1	0	Практическое задание
7	Словари и множества в Python	2	1	1	0	Практическое задание
8	Функции в Python	2	1	0,5	0,5	Практическое задание. Контрольные вопросы
9	Классические алгоритмы	2	1	1	0	Практическое задание
10	Работа с текстовыми файлами в Python	2	0,5	0,5	1	Практическое задание
11	Классы в Python	2	0,5	0,5	1	Практическое задание
12	Основы ООП в Python	2	1	0,5	0,5	Практическое задание. Контрольные вопросы.
13	Основы Pygame	2	0,5	0,5	1	Практическое задание
14	Создание игры-бродилки.	2	1	1	0	Практическое

	Часть 1					задание
15	Создание игры-бродилки. Часть 2	2	1	1	0	Практическое задание
16	Создание игры-бродилки. Часть 3	2	0,5	0,5	1	Практическое задание. Контрольные вопросы
17	Создание игры Flappy Bird. Часть 1	2	1	1	0	Практическое задание
18	Создание игры Flappy Bird. Часть 2	2	0,5	0,5	1	Практическое задание
19	Создание игры Flappy Bird. Часть 3	2	0,5	0,5	1	Практическое задание
20	Компиляция и публикация игр	2	1	0,5	0,5	Практическое задание. Контрольные вопросы.
21	Создание игры в жанре Платформер. Часть 1	2	0,5	0,5	1	Практическое задание
22	Создание игры в жанре Платформер. Часть 2	2	1	1	0	Практическое задание
23	Создание игры в жанре Платформер. Часть 3	2	1	1	0	Практическое задание
24	Создание игры в жанре Платформер. Часть 4	2	0,5	0,5	1	Практическое задание. Контрольные вопросы
25	Создание игры в жанре Tower Defence. Часть 1	2	1	0,5	0,5	Практическое задание
26	Создание игры в жанре Tower Defence. Часть 2	2	0,5	0,5	1	Практическое задание
27	Создание игры в жанре Tower Defence. Часть 3	2	0,5	0,5	1	Практическое задание
28	Создание игры в жанре Tower Defence. Часть 4	2	0	1	1	Практическое задание. Контрольные вопросы
29	Создание игры в жанре 2D Shooter. Часть 1	2	1	1	0	Практическое задание
30	Создание игры в жанре 2D Shooter. Часть 2	2	0,5	0,5	1	Практическое задание
31	Создание игры в жанре 2D Shooter. Часть 3	2	0,5	0,5	1	Практическое задание
32	Создание игры в жанре 2D Shooter. Часть 4.	2	0,5	0,5	1	Практическое задание. Контрольные вопросы.
33	Создание 3D-игры в Pygame. Часть 1	2	1	0,5	0,5	Практическое задание
34	Создание 3D-игры в Pygame. Часть 2	2	0,5	0,5	1	Практическое задание
35	Подготовка к защите проектов	2	0,5	0,5	1	Практическое задание
36	Защита проекта. Итоговая аттестация-презентация	2	0,5	0,5	1	Практическое задание. Контрольные вопросы. Защита проекта
	ИТОГО	72	25	24,5	22,5	

3. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Календарный учебный график формируется при осуществлении обучения в течение всего календарного года. По мере набора групп слушателей по программе составляется календарный график, учитывающий объемы лекций, практики, самоподготовки, другие формы организации занятий.

Темы / недели	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	Итого часов
1. Знакомство с Python	2																																				2
2. Операторы в Python		2																																			2
3. Условные операторы			2																																		2
4. Работа со строками				1,5																																	1,5
Промежуточный контроль				0,5																																	0,5
5. Циклы в Python					2																																2
6. Массивы в Python						2																															2
7. Словари и множества в Python							2																														2
8. Функции в Python								1,5																													1,5
Промежуточный контроль								0,5																													0,5
9. Классические алгоритмы									2																												2
10. Работа с текстовыми файлами в Python										2																											2
11. Классы в Python											2																										2
12. Основы ООП в Python												1,5																									1,5
Промежуточный контроль												0,5																									0,5
13. Основы Pygame													2																								2
14. Создание игры-бродилки. Часть 1														2																							2
15. Создание игры-															2																						2

[illegible]

4. РАБОЧАЯ ПРОГРАММА

1. Знакомство с Python

- Что такое переменная?
- Функции ввода и вывода
- Типы данных в Python
- Создание текстовой игры

Практическое задание: создание текстовой игры.

2. Операторы в Python

- Арифметические операторы в Python
- Операторы сравнения
- Логические операторы
- Знакомство со случайными числами
- Создание игры на заданную тему

Практическое задание: создание игры «Магический лес».

3. Условные операторы

- Знакомство с условными операторами if-else
- Булевы значения
- Решение задач с условными алгоритмами
- Изучение конструкции match-case
- Составление блок-схем

Практическое задание: Создание игры «Текстовое RPG».

4. Работа со строками

- Работа со строками
- Знакомство со срезами
- Методы строк
- Создание игры «Города»

Практическое задание: Создание игры «Города».

5. Циклы в Python

- Знакомство с циклами
- Изучение циклических алгоритмов
- Команды, связанные с циклами
- Вложенные циклы
- Создание игры на заданную тему

Практическое задание: создание игры «Камень-ножницы-бумага».

6. Массивы в Python

- Знакомство с массивами
- Изучение команд для работы с массивами
- Создание матрицы 2x3
- Создание игры на заданную тему

Практическое задание: создание игры «Крестики-нолики».

7. Словари и множества в Python

- Знакомство со словарями и множествами
- Изучение основных команд для словарей и множеств
- Обработка и изменение данных из словарей и множеств
- Создание игры на заданную тему

Практическое задание: Создание игры «Викторина».

8. Функции в Python

- Изучение работы функций
- Возвращаемые значения
- Знакомство с параметрами и аргументами функций
- Создание игры на заданную тему

Практическое задание: Создание игры «Математический квиз».

9. Классические алгоритмы

- Изучение алгоритмов сортировки
- Знакомство с алгоритмами поиска
- Изучение реализации математических алгоритмов (поиск НОД и НОК)
- Создание игры с использованием алгоритмов

Практическое задание: создание игры «Уничтожь пару».

10. Работа с текстовыми файлами в Python

- Знакомство с командой open
- Обработка текстовых файлов
- Сохранение значений и запись в файлы
- Создание игры с использованием файлов

Практическое задание: создание игры «Работники».

11. Классы в Python

- Знакомство с понятием класс
- Создание объектов
- Изучение понятий поле и метод
- Создание игры с использованием классов

Практическое задание: Создание игры «Карточки».

12. Основы ООП в Python

- Изучение понятия и принципов ООП
- Знакомство с наследованием
- Изучение понятия полиморфизм и инкапсуляция
- Создание игры с использованием изученных знаний

Практическое задание: Создание игры «Морской мир».

13. Основы Pygame

- Знакомство с библиотекой Pygame
- Создание экрана
- Рисование геометрических фигур
- Вывод текста на экран
- Создание игры с помощью Pygame

Практическое задание: создание приложения с графическим интерфейсом.

14. Создание игры-бродилки. Часть 1

- Обработка команд с клавиатуры
- Вывод изображений в Pygame
- Создание основы игры-бродилки

Практическое задание: создание основы игры-бродилки.

15. Создание игры-бродилки. Часть 2

- Обработка касания объектов в Pygame
- Создание собираемых элементов
- Реализация условия выигрыша и проигрыша

Практическое задание: добавление в игру-бродилку собираемых элементов и условия выигрыша и проигрыша.

16. Создание игры-бродилки. Часть 3

- Изучение работы со временем
- Добавление усилителей (зелий) в игру
- Реализация системы жизней

Практическое задание: доработка игры-бродилки.

17. Создание игры Flappy Bird. Часть 1

- Создание основы новой игры
- Обработка отображения спрайта птицы, его анимация
- Реализация движения птицы

Практическое задание: создание основы игры Flappy Bird: отладка движения платформы, доработка движения птицы.

18. Создание игры Flappy Bird. Часть 1

- Создание динамических элементов
- Обработка касания с динамическими элементами (трубами)
- Реализация ситуации столкновения и проигрыша
- Реализация системы очков

Практическое задание: настройка параметров и улучшение игры Flappy Bird.

19. Создание игры Flappy Bird. Часть 3

- Создание системы уровней
- Сохранение очков в файл
- Создание таблицы рекордов
- Реализация начального меню

Практическое задание: доработка игры Flappy Bird.

20. Компиляция и публикация игр

- Знакомство с библиотекой Pygbag
- Компиляция игр, созданных с помощью Pygame
- Публикация игры в интернет

Практическое задание: Публикация ранее созданных игр.

21. Создание игры в жанре Платформер. Часть 1

- Создание основы игры «Платформер»
- Реализация персонажа и его движения
- Создание твёрдых объектов, платформ

Практическое задание: создание основы игры «Платформер».

22. Создание игры в жанре Платформер. Часть 2

- Реализация движения уровня
- Создание препятствий
- Реализация различных видов платформ
- Реализация проигрыша и перезагрузки игры

Практическое задание: реализация системы уровней в игре «Платформер».

23. Создание игры в жанре Платформер. Часть 3

- Создание патрулирующих врагов
- Реализация атаки персонажа (стрельба)
- Создание движущихся врагов

Практическое задание: добавление врагов и атаки в игре «Платформер».

24. Создание игры в жанре Платформер. Часть 4

- Создание улучшений
- Реализация системы очков
- Музыка и звуковое сопровождение игры
- Доработка игры и публикация игры

Практическое задание: Доработки игры «Платформер».

25. Создание игры в жанре Tower Defence. Часть 1

- Создание основы игры «Tower Defence»
- Реализация управления стрельбой
- Обработка нажатий мышью на экран

Практическое задание: создание основы игры «Tower Defence».

26. Создание игры в жанре Tower Defence. Часть 2

- Реализация движения и атаки персонажей
- Система жизней замка
- Реализация врагов нескольких видов
- Создание системы покупки здоровья

Практическое задание: добавление новых механик в игру.

27. Создание игры в жанре Tower Defence. Часть 3

- Создание вражеских стрелков-лучников
- Создание защитников замка
- Реализация системы уровней в игре

Практическое задание: улучшение защитников и лучников, а также системы уровней.

28. Создание игры в жанре Tower Defence. Часть 4

- Улучшение внешнего вида игры
- Добавление начального меню
- Добавление звуков и музыки
- Компиляция и публикация игры

Практическое задание: публикация игры «Tower Defence» в интернете.

29. Создание игры в жанре 2D Shooter. Часть 1

- Создание основы игры «2D Shooter»
- Реализация персонажа и его движения по уровню
- Знакомство с переключением состояний спрайтовых анимаций
- Подходы по улучшению своего кода

Практическое задание: добавление анимаций движения и прыжка персонажа, настройка скорости его движения.

30. Создание игры в жанре 2D Shooter. Часть 2

- Добавление стрельбы
- Реализация бросков гранат
- Создание системы сбора предмета

Практическое задание: создание новых игровых механик в игре.

31. Создание игры в жанре 2D Shooter. Часть 3

- Реализация системы жизней
- Добавление врагов
- Создание основы для игрового мира

Практическое задание: добавление разных видов врагов и улучшение геймплея.

32. Создание игры в жанре 2D Shooter. Часть 4

- Реализация карты уровня
- Настройка коллизий игрового мира
- Добавление музыки и звуков
- Публикация игры

Практическое задание: настройка игровой карты с различными видами блоков, добавление различных звуков в игру «2D Shooter».

33. Создание 3D-игры в Pygame. Часть 1

- Изучение команд для создания 3D-игры в Pygame
- Исследование основы уровня
- Написание команд для движения персонажа
- Подготовка идеи итогового проекта

Практическое задание: создание основы и персонажа 3D-игры.

34. Создание 3D-игры в Pygame. Часть 2

- Добавление текстур в 3D-игру
- Добавление врагов
- Добавление окружения, системы жизней, диалогов
- Формирование подборки игр для портфолио

Практическое задание: создание полноценного уровня 3D-игры в Pygame.

35. Подготовка к защите проектов

- Закрепление ранее изученных материалов
- Выбора формата проекта
- Обсуждение и реализация итогового проекта
- Подготовка материалов для презентации проекта

Практическое задание: доделать выбранную игру, дооформить презентацию, подготовить речь к демонстрации итогового проекта.

36. Защита проекта

- Итоговая подготовка к презентации проекта
- Выступления по защите проекта по намеченному плану
- Обсуждение результатов работы, ответы на вопросы
- Подведение итогов курса, изучение перспектив обучения

Практическое задание: выступления и обсуждение созданных проектов, анализ и обратная связь по работам.

5. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

5.1. Контроль знаний, умений и навыков

Формой подведения итогов реализации дополнительной образовательной программы выступает текущая, промежуточная и итоговая аттестация.

Образовательный процесс осуществляется на основании учебного плана и регламентируется расписанием занятий для каждой учебной группы.

В целях оценки показателей знаний, умений и навыков обучающихся по дополнительной образовательной общеразвивающей программе "Программирование игр на Python" проводится текущий и промежуточный контроль знаний, а также итоговая аттестация.

Виды текущего контроля:

- устный ответ на поставленный вопрос;
- проверка результатов выполнения практических заданий.

Виды промежуточного контроля:

- тестирование устное/письменное/с помощью электронных форм
- проверка результатов выполнения практических работ/проектов по итогам учебного модуля

Тестирование — это форма измерения знаний обучающихся, основанная на применении тестов. Материалы для промежуточного и итогового тестирования предоставляются вместе с комплектом учебно-методических материалов к программе.

Итоговая аттестация

Итоговая аттестация проводится с целью установления уровня знаний обучающихся с учетом прогнозируемых результатов обучения и требований к результатам освоения образовательной программы.

Итоговая аттестация обучающихся осуществляется в форме презентации проекта с демонстрацией результатов – веб-сайта с играми, созданными в рамках данного курса. Презентация – это электронный документ, предназначенный для визуальной демонстрации выполненной работы. Как правило, презентация имеет сюжет, сценарий и структуру, созданную для удобного восприятия информации.

Выдача обучающимся документов о дополнительном образовании (сертификат о прохождении курса) осуществляется при условии успешного прохождения итоговой аттестации.

5.2. Критерии оценивания освоения программы при проведении различных форм контроля:

Тестирование (Приложение 1. Примерные вопросы для промежуточного тестирования).
Процент результативности (правильных ответов при выполнении тестовых заданий):

Выполнение теста	Итоговая оценка
60% и более правильных ответов	"Зачтено"
Менее 60% правильных ответов	"Не зачтено"

Проверка выполнения практических работ (Приложение 2. Примерные задания для проверки усвоения качества учебного материала).

Система оценивания:

"Зачтено" – необходимый уровень выполнения задания достигнут, обучающийся демонстрирует хорошее знание теоретической и практической части материала занятия/учебного модуля, достигнуты промежуточные и/или итоговые результаты работы над заданием.

"Не зачтено" - необходимый результат/уровень освоения не достигнут, обучающийся не усвоил теоретические основы и/или изученные практические приемы программирования на языке Python, не достиг промежуточных и итоговых результатов при выполнении задания.

Проверка результатов создания проекта на итоговой аттестации:

Критерии оценки	БАЛЛЫ
Обоснование выбора сюжета игры и его актуальность	0-2 балла
Реализовано не менее двух изученных типов игр	0-4 балла
Реализовано не менее двух уровней игры и пере между ними	0-4 балла
Реализованы программы для управления не менее двух персонажей (игрока и противника) в одной игре	0-4 балла
Реализовано игровое взаимодействие игрока с NPC	0-3 балла
Реализовано игровое взаимодействие игрока с предметами	0-3 балла
Показана самостоятельность работы над проектом	0-2 балла
Компетентность докладчика (ответы на вопросы)	0-2 балла
Итоговая оценка: «Не зачтено» «Зачтено»	0-14 баллов 15-24 балла

6. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ (УЧЕБНИКИ, РАЗДАТОЧНЫЕ МАТЕРИАЛЫ, МЕТОДИЧЕСКИЕ УКАЗАНИЯ, ПЛАКАТЫ, СЛАЙДЫ, ИНТЕРНЕТ-РЕСУРСЫ)

6.1. Информационные и учебно-методические условия реализации программы

Учебно-методический комплект

Для реализации целей и задач обучения по общеобразовательной общеразвивающей программе "Программирование игр на Python" используется комплект материалов преподавателя, который включает:

1. Текстовое методическое пособие с описанием целей, результатов каждого занятия, теоретического материала и практических работ.
2. Видеоурок для преподавателя с методическими указаниями и порядком объяснения учебного материала.
3. Раздаточный материал для учащихся - описание дополнительной самостоятельной работы учащихся по каждому занятию с примерами и рекомендациями по выполнению.
4. Описание мероприятий по контролю знаний – тестовые вопросы, практические задания.
5. Рекомендации по проведению итоговой аттестации и защиты проектов.
6. Дополнительные материалы – презентации по тематике занятий, материалы по работе с дополнительными источниками и программными средствами.
7. Дополнительные материалы – инструкции по установке необходимого программного обеспечения, описание технических требований к компьютерному оборудованию.

Материалы преподавателя размещаются на учебном портале преподавателей, размещенном на сервере информационно-телекоммуникационной сети "Интернет", и доступны по ссылке для всех преподавателей курса. Материалы обучающихся раздаются в печатном виде или рассылаются преподавателем индивидуально каждому обучающемуся.

6.2. Рекомендованная литература для обучающихся

Основная:

1. Свейгарт Эю Большая книга проектов Python./ Эл Свейгарт — СПб.: Питер, 2022. — 432 с.: ил.
2. Дауни А. Основы Python. Научитесь думать как программист / Аллен Б. Дауни ; пер. с англ. С. Черникова ; [науч. ред. А. Родионов]. — Москва : Манн, Иванов и Фербер, 2021. — 304 с.
3. Широков А.И., Пышняк М. Информатика. Разработка программ на языке программирования Питон. Часть 1. Базовые языковые конструкции / А. И. Широков, Марина Пышняк – Москва: ИД МИСиС, 2020. – 144 с.
4. Бриггс Джейсон. Python для детей. Самоучитель по программированию / Д.Бриггс. - Москва: Манн Иванов и Фербер (МИФ), 2016. – 289 с.
5. Грацианова Т.Ю. Информатика. Программирование в примерах и задачах / Т.Ю. Грацианова, Москва: ВМК МГУ (Лаборатория знаний), 2020. – 393 с.
6. Мэттиз Э., Python. Карманный справочник/ Эрик Мэттиз – С.-Петербург:, Питер, 2020 – 512 с.

Дополнительная:

1. Любанович Б. Простой Python. Современный стиль программирования. 2-е изд. / Билл Любанович — СПб.: Питер, 2021. — 592 с.: ил.
2. Лутц М., Изучаем Python. Том 1 / Марк Лутц — Москва: Диалектика, 2019 — 832 с.
3. Лутц М., Изучаем Python. Том 2 / Марк Лутц — Москва: Вильямс, 2020 — 720 с.
4. Брайсон Пейн. Программирование на Python для детей и родителей / Б. Пейн. — Москва: Эксмо, 2017 — 354 с.
5. Introduction to Game Theory. A Discovery Approach / Jennifer Firkins Nordstrom — Linfield College, McMinnville, OR, 2020

6.3. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", рекомендованных для освоения программы

- <https://pythontutor.ru/> - проект Питонтьютор
- <https://ru.hexlet.io/courses/python/101> – Хекслет. Курс «Введение в Python»
- <https://www.udemy.com/course/python-project/> - Python. Практика для начинающих
- <https://github.com> - веб-сервис для хостинга IT-проектов и их совместной разработки
- <https://habr.com/> - Портал habr.com
- <https://www.python.org/downloads/windows/> - официальный сайт Python

7. ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИЕ УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ

7.1. Материально-техническая и ресурсная база

Для реализации программы предполагается использование учебных аудиторий для проведения занятий лекционного типа, занятий семинарского типа, выполнения проектных работ, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации:

- Компьютерные классы, обеспечивающие доступ в Интернет, и оснащенные мультимедиа проектором или иными средствами визуализации учебного материала, магнитной доской или флипчартом.
- Электронный информационно-образовательный портал, размещенный на сервере в информационно-телекоммуникационной сети "Интернет".
- Специальное программное обеспечение для Python-разработки, необходимое для реализации образовательных задач курса.
- Стандартное программное обеспечение для работы над разработкой учебно-методических материалов.
- Мастерские и аудитории для проведения открытых занятий.

Специальных помещений, предполагающих наличие какого-либо специального оборудования для реализации данной программы, не предусматривается.

7.2. Кадровое обеспечение программы

Образовательный процесс по программе осуществляется педагогом дополнительного образования с профильным высшим или средним профессиональным образованием.

К занятию педагогической деятельностью по дополнительной общеобразовательной программе также допускаются лица, обучающиеся по образовательным программам высшего образования по специальностям и направлениям подготовки, соответствующим направленности дополнительных общеобразовательных программ, и успешно прошедшие промежуточную аттестацию не менее чем за два года обучения.

Реализация дополнительной общеобразовательной (общеразвивающей) программы обеспечивается руководящими и педагогическими работниками организации, а также лицами, привлекаемыми к реализации программы на условиях гражданско-правового договора.

У педагогического работника, реализующего дополнительную общеобразовательную программу, должны быть сформированы основные компетенции, необходимые для обеспечения успешного достижения обучающимися планируемых результатов освоения программы, в том числе умения:

- обеспечивать условия для успешной деятельности, позитивной мотивации, а также самомотивирования обучающихся;
- осуществлять самостоятельный поиск и анализ информации с помощью современных информационно-поисковых технологий;
- разрабатывать программы учебных предметов, выбирать учебники и учебно-методическую литературу, рекомендовать обучающимся дополнительные источники информации, в том числе Интернет-ресурсы;
- реализовывать педагогическое оценивание деятельности обучающихся;
- работать с текстовыми редакторами, электронными таблицами, электронной почтой и браузерами, мультимедийным оборудованием.

ПРИЛОЖЕНИЯ

Приложение 1. Примерные вопросы для промежуточного тестирования

Пример 1. Тест по основам Python

1. Какой из перечисленных типов данных не является встроенным в Python?

- a) int
- b) str
- c) bool
- d) array**

2. Что будет результатом выполнения выражения $5 / 2$?

- a) 2
- b) 2.5**
- c) 3
- d) Ошибка

3. Какой из перечисленных циклов в Python используется для перебора элементов в списке?

- a) for**
- b) while
- c) do-while
- d) repeat-until

4. Какая функция используется для получения длины строки в Python?

- a) str.length()
- b) str.size()
- c) len(str)**
- d) size(str)

5. Какой из перечисленных операторов используется для объединения двух списков в Python?

- a) +**
- b) -
- c) *
- d) /

6. Какой символ используется для обозначения комментария в Python?

- a) //
- b) #**
- c) /*
- d) /

7. Какой из перечисленных операторов используется для сравнения двух значений на равенство в Python?

- a) ==**
- b) =
- c) !=

d) <>

8. Какой из перечисленных операторов используется для логического И в Python?

a) &&

b) and

c) ||

d) or

9. Какой из перечисленных операторов используется для логического НЕ в Python?

a) !

b) not

c) ~

d) -

10. Какой метод используется для удаления элемента из массива в Python?

a) remove()

b) delete()

c) del()

d) discard()

11. Какой метод используется для добавления элемента в конец списка в Python?

a) append()

b) add()

c) insert()

d) push()

12. Какой метод используется для объединения двух множеств в Python?

a) union()

b) merge()

c) add()

d) update()

13. Какой метод используется для чтения содержимого файла в Python?

a) read()

b) write()

c) open()

d) close()

14. Какой метод используется для записи данных в файл в Python?

a) read()

b) write()

c) open()

d) close()

15. Какой метод используется для замены подстроки в строке в Python?

a) find()

b) match()

c) search()

d) replace()

16. Как объявить функцию в Python?

- a) `def my_function():`
- b) `function my_function():`
- c) `def my_function`
- d) `function my_function`

17. Как объявить класс в Python?

- a) `class MyClass:`
- b) `MyClass()`
- c) `def MyClass():`
- d) `def MyClass:`

18. Что такое конструктор класса?

- a) Метод, который вызывается при создании объекта класса
- b) Метод, который вызывается при удалении объекта класса
- c) Метод, который вызывается при изменении объекта класса
- d) Метод, который вызывается при вызове метода класса

19. Как объявить наследование классов?

- a) `class MyClass(MyParentClass):`
- b) `class MyClass extends MyParentClass:`
- c) `class MyClass(MyParentClass()):`
- d) `class MyParentClass + MyClass():`

20. Что такое инкапсуляция в Python?

- a) Возможность скрыть переменные и методы класса от других объектов
- b) Возможность использовать переменные и методы класса из других объектов
- c) Возможность создавать новые переменные и методы класса
- d) Возможность изменять переменные и методы класса

Пример 2. Тест по библиотеке pygame

1. Pygame — это:

- A) Язык программирования
- B) Редактор графики
- C) Кросс-платформенная библиотека для разработки игр и мультимедийных приложений на языке Python
- D) Фреймворк для разработки программного обеспечения

2. Как обработать нажатие клавиши в Pygame?

- A) С помощью цикла событий и проверки наличия события с типом KEY_DOWN
- B) С помощью цикла событий и проверки наличия события с типом KEY_UP
- C) Через консоль.
- D) Нельзя обрабатывать нажатия клавиш в Pygame.

3. Какие примитивы доступны для рисования графики в Pygame?

- A) Только линии
- B) Только прямоугольники
- C) Только окружности
- D) Различные примитивы, такие как линии, прямоугольники, окружности и многое другое.

4. Как отрисовывать текст в Pygame?

- A) С помощью метода blit из объекта pygame.text
- B) С помощью метода draw_text из объекта pygame.font
- C) С помощью метода render из объекта pygame.text
- D) С помощью метода display_text из объекта pygame.font

5. Какой метод в Pygame предназначен для создания нового окна на экране?

- A) pygame.display.set_caption()
- B) pygame.display.set_mode()
- C) pygame.time.Clock()
- D) pygame.init()

6. Как обработать отпускание клавиши в Pygame?

- A) С помощью цикла событий и проверки наличия события с типом KEY_DOWN
- B) С помощью цикла событий и проверки наличия события с типом KEY_UP
- C) Через консоль.
- D) Нельзя обрабатывать нажатия клавиш в Pygame.

7. Каким методом в Pygame выводится фон игры?

- A) screen.blit()
- B) screen.draw.()
- C) screen.fill()
- D) screen.get()

8. Какой метод в Pygame используется для настройки размера экрана?

- A) pygame.display.set_mode()
- B) pygame.display.set_caption()
- C) pygame.display.set_palette()
- D) pygame.init()

9. Какой метод в Pygame используется для определения столкновения между прямоугольниками?

- A) pygame.rect.collide_rect()
- B) pygame.sprite.collide_rect()
- C) pygame.rect.colliderect()
- D) pygame.sprite.colliderect()

10. Что представляет собой sprite в Pygame?

- A) Это просто изображение
- B) Это объект, который может быть перемещен на экране
- C) Это класс для создания анимированных спрайтов
- D) Это функция для загрузки текстур и звуков

11. Что такое rect в Pygame?

- A) Это функция для вывода текста на экран
- B) Это класс для создания прямоугольников на экране, которые могут сталкиваться между собой
- C) Это функция для чтения данных из файла
- D) Это класс для управления изображениями

12. Как получить время между игровыми кадрами

- A) dt.wait()
- B) dt = clock.tick()
- C) pygame.time.Clock()
- D) dt = clock.wait()

13. Как добавить объект яблока в массив яблок?

- A) `apples.add(Apple)`
- B) `apples.append(Apple())`
- C) `apples += Apple()`
- D) `apples.addend(Apple)`

14. Какая команда завершает игру?

- A) `running = False`
- B) `game.stop()`
- C) `screen.close()`
- D) `not running`

Приложение 2. Примерные задания для оценки качества освоения учебного материала

Задание 1. Необходимо добавить в игру собираемые элементы, например, кристаллы, монеты.

Пример реализации:

```
1 import random
2
3 import pygame
4
5 WIDTH = 600
6 HEIGHT = 600
7 MOVE_SPEED = 1
8 FPS = 120
9 clock = pygame.time.Clock()
10
11 pygame.init()
12 screen = pygame.display.set_mode((WIDTH, HEIGHT))
13
14 background = pygame.image.load("фон.jpg")
15 background = pygame.transform.scale(background, (WIDTH, HEIGHT))
16
17 win = pygame.image.load("win.jpg")
18 win = pygame.transform.scale(win, (WIDTH, HEIGHT))
19
20 class GameObject(pygame.sprite.Sprite):
21     def __init__(self, x, y, image, width, height):
22         pygame.sprite.Sprite.__init__(self)
23         self.image = pygame.image.load(image)
24         self.image = pygame.transform.scale(self.image, (width, height))
25         self.rect = self.image.get_rect(center=(x, y))
26     def show(self):
27         screen.blit(self.image, (self.rect.x, self.rect.y))
28
29 class Coin(GameObject):
30     def __init__(self, x, y):
31         super().__init__(x, y, "coin.png", 50, 50)
32     def respawn(self):
33         self.rect.x = random.randint(0, 550)
34         self.rect.y = random.randint(0, 550)
35
36 class Spike(GameObject):
37     def __init__(self, x, y):
38         super().__init__(x, y, "spike.png", 50, 100)
39
40 class Player(GameObject):
41     def __init__(self, x, y):
42         super().__init__(x, y, "character.png", 50, 50)
43         self.width = 100
44         self.height = 100
45         self.speedx = 0
46         self.speedy = 0
47         self.coins = 0
48     def move(self):
49         self.rect.x += self.speedx
50         self.rect.y += self.speedy
51
```

```

player = Player(80, 80)
spikes = [Spike(200, 500)]
coins = [Coin(400, 400)]

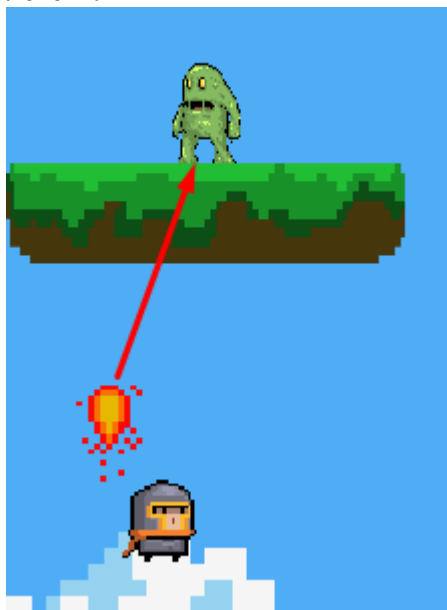
running = True
is_win = False

while running:
    if is_win:
        screen.blit(win, (0, 0))
        pygame.display.flip()
    else:
        screen.blit(background, (0, 0))
        player.show()
        for spike in spikes:
            spike.show()
            if player.rect.colliderect(spike.rect):
                running = False
        for coin in coins:
            coin.show()
            if player.rect.colliderect(coin.rect):
                player.coins += 1
                if player.coins > 2:
                    is_win = True
                coin.respawn()
        player.move()
        f1 = pygame.font.Font(None, 36)
        cointext = f1.render("Монеты: %s" % player.coins, True, (180, 0, 0))
        screen.blit(cointext, (10, 20))
        pygame.display.flip()
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_d and player.rect.x < WIDTH - player.width:
                player.speedx = MOVE_SPEED
            if event.key == pygame.K_a and player.rect.x > 0:
                player.speedx = -MOVE_SPEED
            if event.key == pygame.K_s and player.rect.y < HEIGHT - player.height:
                player.speedy = MOVE_SPEED
            if event.key == pygame.K_w and player.rect.y > 0:
                player.speedy = -MOVE_SPEED
        if event.type == pygame.KEYUP:
            if event.key == pygame.K_d or event.key == pygame.K_a:
                player.speedx = 0
            if event.key == pygame.K_s or event.key == pygame.K_w:
                player.speedy = 0
    clock.tick(FPS)

```



Задание 2. По аналогии с материалом, прооценным на уроке, необходимо добавить еще один вид стрельбы, назначить его на **правую кнопку мыши**. При этом должен появляться другой объект-снаряд, у которого будут другие свойства, например он может намного медленнее лететь.



Задание 3. Создать собираемый элемент, например, какую-нибудь монетку, которая будет создаваться на экране в случайном месте, и лететь в сторону игрока. Если игрок подбирает эту монетку, то количество очков будет увеличиваться, например, сразу на 5.

Задание 4. Придумать новые виды снарядов и собираемых предметов и добавить в игру.

Задание 5. Сделать игру "Фрукты", похожую на игру "Города", созданную на занятии. Проверяться ввод должен с первой буквой. Вводный текст в игру может быть таким: "Поиграем в фрукты! Я загадываю фрукт, ты должен назвать другой фрукт, название которого начинается с той же буквы! Мой фрукт - Манго!".

Задание 6. Сделать сложную версию игры "Крестики-нолики", где размер поля будет 4 на 4, и чтобы выиграть, нужно либо соединить 4 своих знака по вертикали, горизонтали или диагонали, либо собрать фигуру в виде буквы Г (по любой из сторон).

Задание 7. Реализуйте программы для работы со списками: напишите программу, которая выведет на экран максимальный элемент в списке, напишите программу, которая находит наибольший элемент в списке и заменяет его на ноль, напишите программу, которая находит произведение всех элементов списка.

Задание 8. Создайте класс с именем Train, содержащий поля: название пункта назначения, номер поезда, время отправления. Ввести данные в словарь из пяти элементов типа Train. Добавить возможность вывода информации о поезде, номер которого введен пользователем (по счёту в массиве).

Задание 9. Детям нужно сделать так, чтобы при нажатии мышкой в левую часть экрана пули летели влево, а при нажатии в правую часть - вправо. По аналогии с приведённым участком кода программы, в который надо дописать этот функционал игры.

```
direction = 0

bullet_group = pygame.sprite.Group()
grenade_group = pygame.sprite.Group()
explosion_group = pygame.sprite.Group()
items_group = pygame.sprite.Group()

ammo_box = Item(300, 280, "ammobox", 40, 40)
items_group.add(ammo_box)
grenade_box = Item(500, 280, "grenadesbox", 40, 40)
items_group.add(grenade_box)

while running:
    screen.fill("gray")
    pygame.draw.line(screen, "red", (0, 300), (WIDTH, 300))
    player.draw(screen)
    player.update_animation()
    player.move(direction)
    items_group.draw(screen)
    items_group.update(player)
    explosion_group.draw(screen)
    explosion_group.update()
    bullet_group.draw(screen)
    bullet_group.update()
    grenade_group.draw(screen)
    grenade_group.update(explosion_group)
    show_text(screen, "Ammo: %s" % player.ammo, 10, 10, 36, "red")
    show_text(screen, "Grenades: %s" % player.grenades, 10, 40, 36, "red")
    pygame.display.flip()
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_a:
                direction = -1
            if event.key == pygame.K_d:
                direction = 1
            if event.key == pygame.K_SPACE and player.on_ground:
                player.is_jumping = True
        if event.type == pygame.KEYUP:
            if event.key == pygame.K_a or event.key == pygame.K_d:
                direction = 0
        if event.type == pygame.MOUSEBUTTONDOWN:
            if event.button == 1:
                player.shoot(direction, bullet_group)
            if event.button == 3:
                player.throw(direction, grenade_group)
```


Задание 10. Создать карту уровня, по аналогии с приведенным на уроке примером, также связать файл с кодом и файл с картой уровня (map.txt). Добавить, как минимум, ещё один вид блоков, их текстуру и карту уровня для него.

```

1 import pygame.image
2
3 WORLD_WIDTH = 1280 * 2
4 WORLD_HEIGHT = 720
5 TILE_SIZE = 40
6
7 ROWS = WORLD_WIDTH // TILE_SIZE
8 COLS = WORLD_HEIGHT // TILE_SIZE
9
10 tile_paths = {
11     1: "tile_stone.png"
12 }
13
14 tiles = {}
15 for tile in tile_paths:
16     image = pygame.image.load(f"Tiles/{tile_paths[tile]}")
17     image = pygame.transform.scale(image, (TILE_SIZE, TILE_SIZE))
18     tiles[tile] = image
19
20 world_data = []
21
22 with open("map.txt", "r") as file:
23     for line in file:
24         row = []
25         for s in line:
26             if s.isdigit():
27                 row.append(int(s))
28         world_data.append(row)
29
30 class World:
31     def __init__(self):
32         self.list_of_objects = []
33
34     def process_data(self, data):
35         for y in range(len(data)):
36             for x in range(len(data[y])):
37                 if data[y][x] > 0:
38                     img = tiles[tile]
39                     img_rect = img.get_rect()
40                     img_rect.x = x * TILE_SIZE - WORLD_WIDTH // 4
41                     img_rect.y = y * TILE_SIZE
42                     tile_data = (img, img_rect)
43                     self.list_of_objects.append(tile_data)
44
45     def draw(self, screen):
46         for tile_data in self.list_of_objects:
47             screen.blit(tile_data[0], tile_data[1])
48
49 # Main loop
50 screen = pygame.display.set_mode((WORLD_WIDTH, WORLD_HEIGHT))
51 clock = pygame.time.Clock()
52
53 while True:
54     for event in pygame.event.get():
55         if event.type == pygame.QUIT:
56             pygame.quit()
57             sys.exit()
58
59     # Draw the world
60     world = World()
61     world.process_data(world_data)
62     world.draw(screen)
63
64     # Update the display
65     pygame.display.flip()
66
67     # Wait for 60 FPS
68     clock.tick(60)

```