|  |  |  |
| --- | --- | --- |
|

|  |  |
| --- | --- |
| **окончательный шеврон 87x105.jpg** | **Государственное бюджетное образовательное учреждение** **Гимназия №261 Кировского района****Санкт - Петербурга** |

**Итоговая****проектная работа****Тема: «Компьютерная графика:** **прошлое и настоящее»****Предметная область: *Информатика******Работу выполнил******Марченко Артём******Ученик 9б класса******Куратор: Гупалова А. В.*****Санкт-Петербург****2019-2020** |

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc58354619)

[**ИСТОРИЯ РАЗВИТИЯ КОМПЬЮТЕРНОЙ ГРАФИКИ** 5](#_Toc58354620)

[**ВИДЫ КОМПЬЮТЕРНОЙ ГРАФИКИ** 8](#_Toc58354621)

[**СВЕТ И ЦВЕТ В КОМПЬЮТЕРНОЙ ГРАФИКЕ** 10](#_Toc58354622)

[**ОБЛАСТИ ПРИМЕНЕНИЯ КОМПЬЮТЕРНОЙ ГРАФИКИ** 12](#_Toc58354623)

[**ПРАКТИЧЕСКАЯ ЧАСТЬ** 14](#_Toc58354624)

[ПРОГРАММА «ПЛАНЕТЫ СОЛНЕЧНОЙ СИСТЕМЫ» 14](#_Toc58354625)

[ИГРА «ВОЗДУШНЫЙ БОЙ» 16](#_Toc58354626)

[**ЗАКЛЮЧЕНИЕ** 24](#_Toc58354627)

[**ПРИЛОЖЕНИЕ** 25](#_Toc58354628)

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ** 26](#_Toc58354629)

# **ВВЕДЕНИЕ**

**Название проекта: «**Компьютерная графика: прошлое и настоящее».

**Учебный предмет, в рамках которого разрабатывается проект:** информатика.

**Тип проекта:** информационный.

**Актуальность проекта**: компьютерная графика играет существенную роль, как в науке, так и в повседневной жизни любого человека. Знания о компьютерной графике постоянно расширяются. Наука открывает новые виды и способы создания компьютерных изображений, поражающие своей сложностью, красотой и богатством красок.

**Предмет исследования:** компьютерная графика, её виды и свойства.

**Гипотеза:** могу ли я самостоятельно создавать программы в графическом модуле, опираясь на знания, полученные на уроках информатики в школе.

**Цели проекта:**

* продемонстрировать: благодаря своей доступности и наглядности компьютерная графика получила повсеместное распространение;
* создать компьютерную игру

**Задачи проекта:**

* дать понятие определению «компьютерная графика»;
* проследить историю развития компьютерной графики;
* рассмотреть виды компьютерной графики;
* найти примеры применения компьютерной графики в различных областях деятельности человека;
* создать собственную программу в графическом модуле языка программирования Python;
* разработать компьютерную игру в жанре «Аркада»

 **Методы исследования:**

* сбор информации с использованием научно-популярной литературы и других интернет-источников;
* анализ истории развития компьютерной графики;
* выявление проблем при работе с разными видами графики;
* создание графических изображений.

**Предполагаемый продукт проекта:** компьютерная игра на языке программирования Python.

**Руководитель проекта:** Маруга Александра Евгеньевна.

# **ИСТОРИЯ РАЗВИТИЯ КОМПЬЮТЕРНОЙ ГРАФИКИ**

Ещё в древние времена в пещерах первобытные люди рисовали рисунки с помощью различного природного материала. Скалистые выступы на потолке очерчены линиями и покрыты краской. Наскальные росписи выполнялись земляными красками, чёрной сажей и древесным углем с помощью расщеплённых палочек, кусочков меха и просто пальцев. Это изображение бизонов, охотников.

 Ранние формы графики были первыми попытками передачи информации об окружающем мире последующим поколениям. Они представляли собой своеобразный учебник жизни.

 С тех пор прошли многие тысячи лет, появились письменность и книго­печатание, а что изменилось в технике рисования? Стали лучше краски, кисти, появились перья, карандаши, фломастеры, но в принципе все осталось тем же самым, та же цепочка: глаз – рука – инструмент – изображение, те же требования к способностям художника.

 Но вот появилась вычислительная техника. С её помощью стало проще выполнять трудоёмкие математи­ческие задачи, ЭВМ из большого калькулятора неожиданно превратилась в интеллектуальный инструмент, сфера приложения которого стремительно расширяется. В начале 1960-х годов родилась новая область вычислитель­ной техники – интерактивная машинная графика (сегодня чаще называемая компьютерной), где компьютер используется уже не столько для обработки чисел, сколько для работы с графической информацией.

 **Компьютерная графика** (машинная, цифровая графика) – область деятельности, в которой компьютеры используются в качестве инструмента для создания изображений, а также для обработки визуальной информации, полученной из реального мира. Также компьютерной графикой называют и результат этой деятельности.

**Задачи компьютерной графики:**

* Перевод словесного описания в графическое изображение;
* Задача распознавания образов, т. е. перевод картинки в описание;
* Редактирование графических изображений.

Малышева А., Компьютерная графика что такое? Виды компьютерной графики [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://fb.ru/article/190005/kompyuternaya-grafika-chto-takoe-vidyi-kompyuternoy-grafiki Студенческая библиотека онлайн, Общие сведения графических систем [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://studbooks.net/2178722/informatika/obschie\_svedeniya\_graficheskih\_sistem

 Первые вычислительные машины не имели специальных средств для работы с графикой, однако уже использовались для получения и обработки изображений. Программируя память первых электронных машин, построенную на основе матрицы ламп, можно было получать узоры.

 В **1931 г**. в США В. Зворыкин создал ЭЛТ, что впервые позволило получить на экране движущиеся механические частицы.

 В **1951 г**. в Массачусетском технологическом институте для системы противовоздушной обороны военно-морского флота США разработан первый дисплей для компьютера «Вихрь» (Джей Форрестер).

 В **1961 г**. программист С. Рассел возглавил проект по созданию первой компьютерной игры с графикой. Игра Spacewar была создана на машине PDP-1.

 В **1963 г**. американский ученый Айвен Сазерленд создал программноаппаратный комплекс Sketchpad («Блокнот») который позволял рисовать точки, линии и окружности на трубке цифровым пером (световое перо (англ. light реп) – один из инструментов ввода графических данных в компьютер, разновидность манипуляторов). Поддерживались базовые действия с примитивами – перемещение, копирование и др. По сути, это был первый векторный редактор, реализованный на компьютере. Также программу можно назвать первым графическим интерфейсом, причем она являлась таковой еще до появления самого термина.

 В середине **1960-х гг**. появились разработки в промышленных приложениях компьютерной графики. Так, под руководством Т. Мофетта и Н. Тейлора фирма Itek разработала цифровую электронную чертежную машину.

 В **1964** г. General Motors представила систему автоматизированного проектирования DAC-1, разработанную совместно с IBM.

 В **1964 г**. группой под руководством II. II. Константинова была создана компьютерная математическая модель движения кошки. Машина БЭСМ-4, выполняя написанную программу решения дифференциальных уравнений, рисовала мультфильм "Кошечка", который для своего времени являлся прорывом. Для визуализации использовался алфавитно-цифровой принтер.

 В **конце 70-х гг**. в компьютерной графике произошли значительные изменения. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Абдулаев М., Движущиеся картинки Владимира Зворыкина [Электронный ресурс] - Электрон. Дан. – Режим доступа: https://tass.ru/lyudi-i-veschi/6820038

Хелпикс.Орг, История. Первые вычислительные машины [Электронный ресурс] - Электрон. Дан. – Режим доступа: https://helpiks.org/5-61421.html

Появилась возможность создания растровых дисплеев, имеющих множество преимуществ: вывод больших массивов данных, устойчивое, не мерцающее изображение, работа с цветом. Впервые стало возможным получение цветовой гаммы. Растровая технология в конце 70-х стала явно доминирующей. Наиболее знаменательным событием в области компьютерной графики стало создание конце 70-х гг. персонального компьютера.

 В **1977 г**. компания Apple создала Apple-II. Появление этого устройства вызывало смешанные чувства: графика была ужасной, а процессоры медленными. Однако персональные компьютеры стимулировали процесс разработки периферийных устройств. Конечно, персональные компьютеры развивались как важная часть машинной графики, особенно с появлением в **1984 г.** модели Apple Macintosh с их графическим интерфейсом пользователя. Первоначально областью применения персонального компьютера были не графические приложения, а работа с текстовыми процессорами и электронными таблицами, но его возможности как графического устройства побуждали к разработке относительно недорогих программ как в области CAD/CAM, так и в более общих областях бизнеса и искусства.

 В **1982 г**. вышел фильм «Трон», в котором впервые использовались кадры, синтезированные на компьютере.

 К **концу 80-х гг.** программное обеспечение имелось для всех сфер применения: от комплексов управления до настольных издательств. В конце 80-х возникло новое направление рынка на развитие аппаратных и программных систем сканирования, автоматической оцифровки. Оригинальный толчок в таких системах должна была создать магическая машина Ozalid, которая бы сканировала и автоматически векторизовала чертеж на бумаге, преобразовывая его в стандартные форматы. Однако акцент сдвинулся в сторону обработки, хранения и передачи сканируемых пиксельных изображений.

 В **90-х гг**. стираются отличия между компьютерной графикой и обработкой изображения. Машинная графика часто имеет дело с векторными данными, а основой для обработки изображений является пиксельная информация. Кроме того, появляется возможность работы с видео.

 Возрастающий потенциал персональных компьютеров и их громадное число - порядка 100 миллионов - обеспечивает устойчивый рост индустрии в отрасли.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Учебные материалы онлайн, Компьютерная графика и ее применение во всех сферах человеческой деятельности [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://studwood.ru/1622419/informatika/oblasti\_primeneniya\_kompyuternoy\_grafiki

# **ВИДЫ КОМПЬЮТЕРНОЙ ГРАФИКИ**

 По способам задания изображений компьютерную графику можно разделить на категории.

**Растровая графика**

 **Растровое** изображение представляет из себя нечто, похожее на клетчатый лист бумаги, где одна клетка, это одна точка–пиксел, а образуемые ими строки и столбцы собираются в матрицу (растр). У каждого пикселя свой цвет и место, где он расположен. В комплексе, все пикселе образуют изображение.

**Растровые изображения обладают следующими характеристиками**:

* Разрешение – количество пикселей, приходящихся на единицу площади;
* Размер – ширина и высота в пикселях;
* Цветовое пространство – метод отображения цветов в координатах какой-либо цветовой системы;
* Глубина цвета – наибольшее количество оттенков цветов, которое может содержать изображение.

**Достоинства растра:**

* Реалистичность;
* Возможность автоматизированного ввода информации;
* Быстрая обработка трудных иллюстраций;
* Адаптивность под всевозможные устройства и программы просмотра;

**Недостатки растровых изображений:**

* Большой размер занимаемой памяти;
* Увеличение изображения приводит к эффекту пикселизации, иллюстрация искажается.

**Применение:** с растровой графикой работают дизайнеры интерьеров, аниматоры, художники, web-разработчики, графические дизайнеры. К распространённым редакторам можно отнести Adobe Photoshop.

Академия дизайна, Виды компьютерной графики [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://sansans.ru/blog/illyustrator/vidy-kompyuternoj-grafiki.html

**Векторная графика**

 **Векторный** рисунок можно представить в облике элементарных геометрических объектов: точки, прямые, кривые, окружности, многоугольники, и т.д. Фигурам присваиваются какие-либо качества, например, толщина линий, цвет заливки. Для создания иллюстраций используются формулы и координаты. К примеру, чтобы нарисовать треугольник нужно указать его вершины, цвет заполнения и обводку. Для сложных рисунков используют набор геометрических фигур, которые собираются вместе как аппликация из бумаги на уроке труда в начальной школе, но при этом сохраняется возможность в дальнейшем редактировать получившуюся картинку.

**Достоинства векторной графики:**

* Малый объем занимаемой памяти на ПК;
* Трансформация и масштабирование без потери качества;
* Выглядит всегда одинаково, независимо от характеристик устройства отображения.

**Недостатки векторной графики:**

* Невозможность представления всех изображений с помощью примитивов;
* Трудоемкий процесс перевода растровых изображений в векторные;
* Отсутствие автоматического ввода;
* Проблемы с совместимостью программ просмотра и создания.

**Применение:** Векторные картинки широко востребованы на предприятиях, занимающихся проектированием, конструкторских бюро, в рекламных агентствах, типографиях, и т. д. Графические редакторы, работающие с данными иллюстрациями, являются: Adobe Illustrator, Corel Draw, AutoCad, ArhiCad.

**Фрактальная графика**

 Во **фрактальной** графике реализован принцип наследования геометрических качеств, передающихся от одного элемента к другому. Основана данная модель на математических вычислениях (формулах) и так как детализированного описания мелких составляющих не \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Академия дизайна, Виды компьютерной графики [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://sansans.ru/blog/illyustrator/vidy-kompyuternoj-grafiki.html требуется, то обрисовать такой объект можно несколькими уравнениями, результаты которых в дальнейшем машина отображает автоматически, и не требует хранения в памяти компьютера каких-либо объектов. Фрактальная графика незаменима при создании искусственных гор, облаков, морских волн. Благодаря фракталам легко изображаются сложные объекты, образы которых похожи на природные. Фракталы позволяют описывать целые классы изображений, для детального описания которых требуется относительно мало памяти. С другой стороны, фракталы слабо применимы к изображениям вне этих классов.

**Применение:** Фрактальный принцип отображения графики нашел широкое применение во многих областях компьютерной графики, науки и искусства. Фракталы широко применяются в растровой, векторной и 3D графике. Можно отметить несколько программ для генерирования фракталов: Fractal Explorer, Apophysis, Mandelbulb3D.

**Трёхмерная графика**

 **Трёхмерная** графика работает с объектами в трёхмерном пространстве – ширина, высота и глубина. Предметы моделируются и перемещаются в виртуальном пространстве и могут быть рассмотренными под различным углом.

**Трёхмерные модели могут быть двух типов:**

* Полигональная – совокупность вершин, ребер и граней, которые определяют форму многогранного объекта, обволакивая пустое 3D пространство;
* Воксельная – совокупность элементов объемного изображения, содержащая значение растра, которые выкладываются в объёмные модели объектов, имеющие внутренности.

**Применение:** Трёхмерная графика встречается повсеместно и используется в создании изображений во всевозможных областях деятельности человека: машиностроение, архитектура, дизайн интерьера, реклама, игровая и киноиндустрия, интерактивные обучающие проекты. Можно выделить следующие редакторы: 3ds Max, Autodesk Maya, Cinema 4D, Blender.

# Академия дизайна, Виды компьютерной графики [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://sansans.ru/blog/illyustrator/vidy-kompyuternoj-grafiki.html

# **СВЕТ И ЦВЕТ В КОМПЬЮТЕРНОЙ ГРАФИКЕ**

 Свет, как физическое явление, представляет собой поток электромагнитных волн разной длины и амплитуды. Глаз воспринимает эти волны в диапазоне 350~780 нм. Цвет - это характеристика восприятия глазом электромагнитных волн разной длины (длина волны определяет для глаза цвет, а ее амплитуда - яркость). (См. Приложение 1)

 Именно таким образом рецепторы глаза посылают в мозг человека сигнал RGB (красный, зеленый, синий). Если цвет содержит все видимые длины волн в приблизительно равных количествах, то он называется ахроматичным и при максимальной интенсивности воспринимается как белый, а при более низких интенсивностях – как оттенки серого. Интенсивность цвета удобно рассматривать в диапазоне 0~1, і тогда нулевое значение соответствует чёрному цвету. Если цвет содержит длины волн в неравных пропорциях, то его называют хроматичным.

 Для описания цвета в компьютерной графике используют различные цветовые модели. **Цветовая модель** - это способ формального описания цвета на основе составляющих его компонентов. Любой цвет является комбинацией трёх основных: красного (Red), зелёного (Green), синего (Blue). При “смешивании” двух основных цветов образуется результирующий: красный+зеленый=жёлтый, зелёный+синий=бирюзовый, синий+красный=лиловый. “Смешать” все три цвета – белый. Все цвета “выключить” – чёрный. Модель RGB является аддитивной (add - сложить), т.е. для получения нового оттенка необходимо смешать основные цвета в соответствующих пропорциях. Каждой точке соответствуют три цифры, отражающие интенсивность каждого цвета. Модель CMYK является субтрактивной (subtract – отнять). Цветовые компоненты CMYK- цвета, полученные отниманием основных из белого.

 Компьютерные программы при работе с цветовыми моделями имеют специальные инструменты для смешивания различных цветов. Разработчики ПО предложили три свойства цвета: оттенок (Hue) – тон радуги; насыщенность (Saturation) – содержание в цвете серого; яркость (Brightness) – интенсивность, с которой излучается цвет. HSB-свойство. Модель HSB является перцепционной (perception – восприятие) – настроена под восприятие цвета человеком. Основные цвета убывают по яркости сверху вниз.

Мельниченко В. В., Легейда В. В., Настоящий самоучитель компьютерной графики – К.: Век+, СПб.: КОРОНА принт, К.: НТИ, 2005. – 560 с.

**ОБЛАСТИ ПРИМЕНЕНИЯ КОМПЬЮТЕРНОЙ ГРАФИКИ**

 **Научная графика** – первые компьютеры использовались лишь для решения научных и производственных задач. Чтобы лучше представить полученные результаты, производилась их графическая обработка, строились графики, диаграммы, чертежи рассчитанных конструкций. Первые графики на машине получали в режиме символьной печати. Современная научная компьютерная графика дает возможность проводить вычислительные эксперименты с наглядным представлением их результатов.

 **Деловая графика** – область компьютерной графики, предназначенная для наглядного представления различных показателей работы учреждений. Плановые показатели, отчетная документация, статистические сводки – для них с помощью компьютерной графики создаются иллюстративные материалы. Программные средства деловой графики включаются в состав электронных таблиц.

 **Конструкторская графика** используется в работе инженеров-конструкторов, архитекторов, изобретателей новой техники. Этот вид компьютерной графики является обязательным элементом САПР (систем автоматизации проектирования). Средствами конструкторской графики можно получать как плоские изображения (проекции, сечения), так и пространственные трехмерные изображения.

 **Иллюстративная графика** – рисование, черчение, моделирование на экране компьютера. Пакеты иллюстративной графики относятся к прикладному программному обеспечению общего назначения. Программные средства иллюстративной графики называются графическими редакторами.

 **Художественная и рекламная графика** популярна во многом благодаря развитию фотографии, рекламы и телевидения. С помощью компьютера создаются печатные материалы, различного рода рекламная продукция, мультфильмы, компьютерные игры, интерактивные и видеоуроки, слайд- и видеопрезентации.

 **Компьютерная анимация** – создание движущихся изображений. Художник создает на экране рисунки начального и конечного положения движущихся объектов, все промежуточные состояния рассчитывает и изображает компьютер, выполняя расчеты, опирающиеся на математическое описание данного вида движения. Полученные рисунки, выводимые последовательно на экран с определенной частотой, создают иллюзию движения.

Учебные материалы онлайн, Компьютерная графика и ее применение во всех сферах человеческой деятельности [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://studwood.ru/1622419/informatika/oblasti\_primeneniya\_kompyuternoy\_grafiki **Мультимедиа** – объединение высококачественного изображения на экране компьютера со звуковым сопровождением. Наибольшее распространение системы мультимедиа получили в области обучения, рекламы, развлечений.

 Область применения компьютерной графики не ограничивается одними художественными эффектами. Во всех отраслях науки, техники, медицины, в коммерческой и управленческой деятельности используются построенные с помощью компьютера схемы, графики, диаграммы, предназначенные для наглядного отображения разнообразной информации.

 Конструкторы, разрабатывая новые модели автомобилей и самолетов, используют трёхмерные графические объекты, чтобы представить окончательный вид изделия. Художники, архитекторы и дизайнеры уже не мыслят своей работы без использования компьютерной графики. Трёхмерная графика позволяет смоделировать архитектурный объект и позволяет оценить его достоинства более объективно, чем это возможно сделать на основе чертежей или макетов. Дизайнер по интерьерам сейчас может предложить заказчику почти фотографическое изображение его будущего жилья, тогда как раньше, возможно было довольствоваться только эскизами. Современный кинематограф, СМИ, реклама - пример широкого применении компьютерной графики.

 Наши знания постоянно расширяются. Мы уже используем графику не только в научных целях, но и в повседневной жизни, отправляя друзьям с телефона картинки и анимации. Сейчас, наверное, каждый школьник может отредактировать изображение, воспользовавшись Paint или Adobe Photoshop. У наших родителей информатика была только в старших классах, а мы её проходим уже с пятого. Нас учат работать в Microsoft Word, Microsoft Power Point, Microsoft Excel, мы изучаем ЛогоМиры, КуМир, Python и Pascal.ABC.

 Работая над проектом, я задумался: «Что я смогу создать самостоятельно, используя знания, полученные на уроках информатики в школе?» Для создания графического изображения я выбрал Python. Это простой в изучении язык программирования, он особенно хорош для начинающих. В отличие от многих других языков, Python-код легко читается, а интерактивная оболочка позволяет вводить программы и сразу же получать результат.

Учебные материалы онлайн, Компьютерная графика и ее применение во всех сферах человеческой деятельности [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://studwood.ru/1622419/informatika/oblasti\_primeneniya\_kompyuternoy\_grafiki

# **ПРАКТИЧЕСКАЯ ЧАСТЬ**

## ПРОГРАММА «ПЛАНЕТЫ СОЛНЕЧНОЙ СИСТЕМЫ»

 Создавать графическое изображение в Python я буду при помощи элемента интерфейса Canvas (холст). Canvas предоставляет методы для рисования простых двумерных фигур, таких как линии, прямоугольники, овалы, многоугольники и т.д.

 Canvas имеет многочисленные методы для рисования графических фигур:

* create\_line () – чертит на холсте отрезки прямой между двумя или несколькими точками;
* create\_rectangle () – чертит на холсте прямоугольник;
* create\_oval () – чертит овал, или эллипс;
* create\_arc () - чертит дугу;
* create\_polygon () - чертит замкнутый многоугольник;
* create\_text () - для нанесения текста.

 Используемый в методе create\_text () шрифт задаётся путём создания объекта font и его передачи в качестве аргумента font=. Класс font находится в модуле tkinter.font, поэтому необходимо вызвать его в коде командой: import tkinter

**Программа «Планеты Солнечной системы»**

1. **from** tkinter **import** **\***
2.
3. window **=** Tk()                      # Создаём окно
4. window.title('Солнечная система')  # Даём окну название "Солнечная система"
5. c **=** Canvas(window, width**=**550, height**=**300,
6. bg**=**'black')             # Задаём параметры созданного окна
7.
8. c.pack()                           # Вызываем окно
9.
10.
11. c.create\_text(275,50, text**=**'Планеты Солнечной системы',
12. fill**=**'white', font**=**('Times',30))
13.
14. # Создаём планеты
15.
16. sun **=** c.create\_oval(25,124,150,244,
17. fill**=**'yellow')
18. c.create\_text(88,249,text**=**'Солнце', fill**=**'white')
19.
20. mercury **=** c.create\_oval(170,175,190,193,
21. fill**=**'gray43')
22. c.create\_text(182,202,text**=**'Меркурий', fill**=**'white')
23.
24. venus **=** c.create\_oval(200,170,230,198,
25. fill**=**'tan1')
26. c.create\_text(215,160, text**=**'Венера', fill**=**'white')
27.
28. earth **=** c.create\_oval(240,173,265,195,
29. fill**=**'medium blue')
30. c.create\_text(252,200, text**=**'Земля', fill**=**'white')
31.
32. mars **=** c.create\_oval(275,178,290,191,
33. fill**=**'red3')
34. c.create\_text(282,170, text**=**'Марс', fill**=**'white')
35.
36. jupiter **=** c.create\_oval(300,160,350,208,
37. fill**=**'khaki3')
38. c.create\_text(325,215, text**=**'Юпитер', fill**=**'white')
39.
40. ring **=** c.create\_oval(355,164,435,204,
41. fill**=**'gray74')
42.
43. saturn **=** c.create\_oval(370,160,420,208,
44. fill**=**'dark khaki')
45. c.create\_text(395,152, text**=**'Сатурн', fill**=**'white')
46.
47. uranus **=** c.create\_oval(436,164,478,204,
48. fill**=**'cyan2')
49. c.create\_text(457,210, text**=**'Уран', fill**=**'white')
50.
51. neptune **=** c.create\_oval(488,166,524,202,
52. fill**=**'royalblue3')
53. c.create\_text(506,159, text**=**'Нептун', fill**=**'white')
54.
55. window.mainloop()       # При запуске через консоль программа
56. # закроется не сразу

 Результатом программы яляется рисунок «Планеты Солнечной системы» (См. Приложение 2).

## ИГРА «ВОЗДУШНЫЙ БОЙ»

 Каждому интересны определённые жанры игры. Я люблю аркады - игры, характеризующийся коротким по времени, но интенсивным игровым процессом.

 В большинстве своём, аркады не заставляют сильно думать. Там большая часть геймплея основана на рефлексах человека. Но, тем не менее, мы можем получить массу удовольствия, так как ты в один и тот же момент и отдыхаешь, и развиваешь отдельные части своего мозга. Люди, играющие в подобные игры, отличаются максимально быстрой реакцией, так как реагируют на десятки незапланированных, не поддающихся анализу явлений.

 Суть моей игры заключается в том, чтобы уничтожить вражеские космические корабли, а самому выжить. Цель – пройти 15 уровней. На прохождение игры даётся 3 жизни.

ПРАВИЛА ИГРЫ

* Ваша главная задача – уничтожить всех противников на уровне;
* Для уничтожения корабля пришельцев достаточно одного попадания;
* Если все противники уничтожены, вы выиграли и переходите на следующий уровень;
* Если в вас попали, вы теряете часть жизни;
* Если у вас закончились все 3 жизни, то вы проиграли, и игра предложит вам начать заново.

 Создавать свою игру я буду на языке программирования Python.

Для начала работы с программой нужно запустить файл ai.py

Для запуска программы, на компьютере должен быть установлен интерпретатор языка Python, версии 3.x. и модуль PyGame через консольную команду:

«python –m pip install –user pygame»

 Программу начнём со строки: import pygame, time, random. Этим мы загружаем необходимые модули и библиотеки.

В нашем случае, это:

* PyGame - набор библиотек языка Python, предназначенный для написания компьютерных игр и мультимедиа-приложений
* time - для работы со временем
* random - для генерации случайных чисел
* mixer – метод для добавления музыки в игру
* font – метод, с которым можно использовать шрифты в программе
* transform – если загруженный фон будет меньше игрового поля, то он растянется по ширине и длине поля

Комментарии буду обозначать значком **#** (т.е. всё, что находится в строке после этого символа, является комментарием).

1. **import** pygame, time, random
2.
3. # Для работы со шрифтами:
4. pygame.font.init()
5.
6. # Чтобы проигрывать музыку:
7. pygame.mixer.init()
8.
9. # Параметры окна
10. WIDTH, HEIGHT **=** 1024, 768
11. WIN **=** pygame.display.set\_mode((WIDTH, HEIGHT))
12. pygame.display.set\_caption("Вторжение пришельцев")
13.
14. # Корабль игрока
15. PLAYER\_SHIP **=** pygame.image.load("img/ship.png")
16.
17. # Пули
18. LASER **=** pygame.image.load("img/pixel\_laser.png")
19.
20. # Задний фон
21. BG **=** pygame.transform.scale(pygame.image.load("img/bg.jpg"), (WIDTH, HEIGHT))
22.
23. **class** Laser:
24. # Создаём пулю и задаём её логику
25. **def** \_\_init\_\_(self, x, y, img):
26. self.x **=** x**-**15
27. self.y **=** y**-**20
28. self.img **=** img
29. self.mask **=** pygame.mask.from\_surface(self.img)
30.
31. **def** draw(self, window):
32. window.blit(self.img, (self.x, self.y))
33.
34. **def** move(self, vel):
35. self.y **+=** vel
36.
37. **def** off\_screen(self, height):
38. **return** **not**(self.y <**=** height **and** self.y >**=** 0)
39.
40. **def** collision(self, obj):
41. **return** collide(self, obj)
42.
43. **class** Ship:
44. # Скорость стрельбы
45. COOLDOWN **=** 20
46.
47. # Настройки корабля
48. **def** \_\_init\_\_(self, x, y, health**=**100):
49. self.x **=** x
50. self.y **=** y
51. self.health **=** health
52. self.ship\_img **=** None
53. self.laser\_img **=** None
54. self.lasers **=** []
55. self.cool\_down\_counter **=** 0
56.
57. **def** draw(self, window):
58. window.blit(self.ship\_img, (self.x, self.y))
59. **for** laser **in** self.lasers:
60. laser.draw(window)
61.
62. **def** move\_lasers(self, vel, obj):
63. self.cooldown()
64. **for** laser **in** self.lasers:
65. laser.move(vel)
66. **if** laser.off\_screen(HEIGHT):
67. self.lasers.remove(laser)
68. **elif** laser.collision(obj):
69. obj.health **-=** 10
70. self.lasers.remove(laser)
71.
72. **def** cooldown(self):
73. **if** self.cool\_down\_counter >**=** self.COOLDOWN:
74. self.cool\_down\_counter **=** 0
75. **elif** self.cool\_down\_counter > 0:
76. self.cool\_down\_counter **+=** 1
77.
78. **def** shoot(self):
79. **if** self.cool\_down\_counter **==** 0:
80. laser **=** Laser(self.x, self.y, self.laser\_img)
81. self.lasers.append(laser)
82. self.cool\_down\_counter **=** 1
83.
84. **def** get\_width(self):
85. **return** self.ship\_img.get\_width()
86.
87. **def** get\_height(self):
88. **return** self.ship\_img.get\_height()
89.
90. **class** Player(Ship):
91. # Создаём корабль игрока
92. **def** \_\_init\_\_(self, x, y, health**=**100):
93. super().\_\_init\_\_(x, y, health)
94. self.ship\_img **=** PLAYER\_SHIP
95. self.laser\_img **=** LASER
96. self.mask **=** pygame.mask.from\_surface(self.ship\_img)
97. self.max\_health **=** health
98.
99. **def** move\_lasers(self, vel, objs):
100. self.cooldown()
101. **for** laser **in** self.lasers:
102. laser.move(vel)
103. **if** laser.off\_screen(HEIGHT):
104. self.lasers.remove(laser)
105. **else**:
106. **for** obj **in** objs:
107. **if** laser.collision(obj):
108. objs.remove(obj)
109. **if** laser **in** self.lasers:
110. self.lasers.remove(laser)
111.
112. **def** draw(self, window):
113. super().draw(window)
114. self.healthbar(window)
115.
116. **def** healthbar(self, window):
117. pygame.draw.rect(window, (255,0,0), (self.x, self.y **+** self.ship\_img.get\_height() **+** 10,
118. self.ship\_img.get\_width(), 10))
119.
120. pygame.draw.rect(window, (0,255,0), (self.x, self.y **+** self.ship\_img.get\_height() **+** 10,
121. self.ship\_img.get\_width() **\*** (self.health**/**self.max\_health), 10))
122. **class** Enemy(Ship):
123. # Пришельцы
124. **def** \_\_init\_\_(self, x, y, health**=**50):
125. super().\_\_init\_\_(x, y, health)
126. self.ship\_img **=** pygame.image.load('img/ufo.png')
127. self.mask **=** pygame.mask.from\_surface(self.ship\_img)
128.
129. **def** move(self, vel):
130. self.y **+=** vel
131.
132. **def** collide(obj1, obj2):
133. offset\_x **=** obj2.x **-** obj1.x
134. offset\_y **=** obj2.y **-** obj1.y
135. **return** obj1.mask.overlap(obj2.mask, (offset\_x, offset\_y)) !**=** None
136.
137. **def** main():
138. run **=** True
139. FPS **=** 60
140. level **=** 0
141. lives **=** 3
142.
143. main\_font **=** pygame.font.SysFont("comicsans", 50)
144. lost\_font **=** pygame.font.SysFont("comicsans", 60)
145. win\_font **=** pygame.font.SysFont("comicsans", 60)
146.
147. enemies **=** []
148. wave\_length **=** 5
149.
150. # Задаём скорость пришельцев, пуль, игрока
151. enemy\_vel **=** 1
152. player\_vel **=** 5
153. laser\_vel **=** 5
154.
155. # Начальная позиция игрока
156. player **=** Player(500, 600)
157.
158. clock **=** pygame.time.Clock()
159.
160. lost **=** False
161. lost\_count **=** 0
162.
163. win **=** False
164. win\_count **=** 0
165.
166. **def** redraw\_window():
167. WIN.blit(BG, (0,0))
168. # Переменные, которые хранят кол-во жизней и номер уровня
169. lives\_label **=** main\_font.render(f"Жизни: {lives}", 1, (255,255,255))
170. level\_label **=** main\_font.render(f"Уровень: {level}", 1, (255,255,255))
171.
172. # Вывод на экран жизней и номер уровня
173. WIN.blit(lives\_label, (10, 10))
174. WIN.blit(level\_label, (WIDTH **-** level\_label.get\_width() **-** 10, 10))
175.
176. **for** enemy **in** enemies:
177. enemy.draw(WIN)
178.
179. player.draw(WIN)
180.
181. **if** lost:
182. lost\_label **=** lost\_font.render("Попробуй ещё раз!", 1, (255,255,255))
183. WIN.blit(lost\_label, (WIDTH**/**2 **-** lost\_label.get\_width()**/**2, 350))
184.
185. **if** win:
186. win\_label **=** win\_font.render("Хорошая работа!", 1, (255,255,255))
187. WIN.blit(win\_label, (WIDTH**/**2 **-** win\_label.get\_width()**/**2, 350))
188.
189. pygame.display.update()
190.
191. **while** run:
192. clock.tick(FPS)
193. redraw\_window()
194.
195. **if** lives <**=** 0 **or** player.health <**=** 0:
196. pygame.mixer.music.pause()
197. lost **=** True
198. lost\_count **+=** 1
199.
200. **if** level **==** 16:
201. pygame.mixer.music.pause()
202. win **=** True
203. win\_count **+=** 1
204.
205. **if** lost:
206. **if** lost\_count > FPS **\*** 3:
207. run **=** False
208. **else**:
209. **continue**
210.
211. **if** win:
212. **if** win\_count > FPS **\*** 3:
213. run **=** False
214. **else**:
215. **continue**
216.
217. **if** len(enemies) **==** 0:
218. level **+=** 1
219. wave\_length **+=** 5
220. **for** i **in** range(wave\_length):
221. enemy **=** Enemy(random.randrange(50, WIDTH**-**100), random.randrange(**-**1500, **-**100))
222. enemies.append(enemy)
223.
224. **for** event **in** pygame.event.get():
225. **if** event.type **==** pygame.QUIT:
226. quit()
227.
228. # Задаём направления, в которых может двигаться игрок
229. # и запрещаем кораблю игрока вылетать за границы игрового поля
230. keys **=** pygame.key.get\_pressed()
231.
232. # Движение вверх и запрет на вылет за верхнюю границу экрана
233. **if** keys[pygame.K\_w] **and** player.y **-** player\_vel > 0:
234. player.y **-=** player\_vel
235.
236. # Движение влево и запрет на вылет за левую границу экрана
237. **if** keys[pygame.K\_a] **and** player.x **-** player\_vel > 0:
238. player.x **-=** player\_vel
239.
240. # Движение вниз и запрет на вылет за нижнюю границу экрана
241. **if** keys[pygame.K\_s] **and** player.y **+** player\_vel **+** player.get\_height() **+** 15 < HEIGHT:
242. player.y **+=** player\_vel
243.
244. # Движение вправо и запрет на вылет за правую границу экрана
245. **if** keys[pygame.K\_d] **and** player.x **+** player\_vel **+** player.get\_width() < WIDTH:
246. player.x **+=** player\_vel
247.
248. # Теперь корабль будет стрелять при нажании клавиши "Пробел"
249. **if** keys[pygame.K\_SPACE]:
250. player.shoot()
251. # Рестарт игры
252. **if** keys[pygame.K\_q]:
253. pygame.mixer.music.pause()
254. run **=** False
255. **break**
256.
257. **for** enemy **in** enemies[:]:
258. enemy.move(enemy\_vel)
259.
260. # При столкновении пришельца и игрока, здоровье игрока
261. # Уменьшается на 10, а пришелец стирается
262. **if** collide(enemy, player):
263. player.health **-=** 10
264. enemies.remove(enemy)
265. **elif** enemy.y **+** enemy.get\_height() > HEIGHT:
266. lives **-=** 1
267. enemies.remove(enemy)
268.
269. player.move\_lasers(**-**laser\_vel, enemies)
270.
271. **def** main\_menu():
272. title\_font **=** pygame.font.SysFont("comicsans", 70)
273. run **=** True
274.
275. pygame.mixer.music.unpause()
276. pygame.mixer.music.load('sounds/bg.wav')
277. pygame.mixer.music.play(**-**1)
278. pygame.mixer.music.set\_volume(0.2)
279.
280. **while** run:
281.
282. WIN.blit(BG, (0,0))
283. title\_label **=** title\_font.render("Нажмите ЛКМ, чтобы начать...", 1, (255,255,255))
284. WIN.blit(title\_label, (WIDTH**/**2 **-** title\_label.get\_width()**/**2, 350))
285. pygame.display.update()
286. **for** event **in** pygame.event.get():
287. **if** event.type **==** pygame.QUIT:
288. run **=** False
289. **if** event.type **==** pygame.MOUSEBUTTONDOWN:
290. pygame.mixer.music.unpause()
291. main()
292. pygame.quit()
293.
294. main\_menu()

Результатом программы является аркадная игра «Воздушный бой» (См. Приложение 3)

# **ЗАКЛЮЧЕНИЕ**

 В ходе выполнения проектной работы, мною была собрана и изучена информация об истории возникновения и развития компьютерной графики. Я выяснил, что прогресс, особенно на начальных этапах, в первую очередь был связан с развитием технических средств, главным образом дисплеев. Сейчас компьютерная графика стала основным средством связи между человеком и компьютером, постоянно расширяющим сферы своего применения, т.к. в графическом виде результаты становятся более наглядными и понятными.

 Я сравнил разные виды графики, выявил достоинства и недостатки в их использовании, а также выяснил, какие цветовые модели существуют. Детально изучив информацию, я решил поработать в области художественной графики.

 Для написания программы я выбрал язык программирования Python, т.к. мы проходим его в школе на уроках информатики. К сожалению, моих знаний оказалось недостаточно, и я стал осваивать дополнительную литературу. Прочитав книгу Тонни Гэддиса «Начинаем программировать на Python», я изучил новый для меня модуль Tkinter и справился с поставленной задачей.

 Следующим этапом для меня стало написание программы для компьютерной игры. Свой «Воздушный бой» я также создавал в Python. В этом мне помогли книги Эрика Мэтиза и Эла Свейгарта. С задачей я также справился, хотя это и оказалось намного труднее, чем я себе представлял в начале проекта.

**Вывод:** С целями проекта я справился, выполнив все поставленные задачи. Однако, одних лишь школьных уроков информатики мне оказалось недостаточно для создания программ.

 В результате работы над проектом я узнал много нового и интересного. Безусловно, знания, полученные мной в процессе написания программ, окажутся полезными и будут способствовать моему дальнейшему развитию в области информатики.

# **ПРИЛОЖЕНИЕ**

**1**



**2**

 **«Планеты Солнечной системы»**



**3**

 **«Воздушный бой»**

****

# **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

* Гэддис Т., Начинаем программировать на Python. – 4-е изд.: Пер. с англ. – СПБ:

 БХВ-Петербург, 2019. – 768 с.: ил.

* Мельниченко В. В., Легейда В. В., Настоящий самоучитель компьютерной графики – К.: Век+, СПб.: КОРОНА принт, К.: НТИ, 2005. – 560 с.
* Мэтиз Эрик, Изучаем Python: программирование игр, визуализация данных, веб-приложения. 3-е изд. – СПб.: Питер, 2020. – 512 с.: ил. – (Серия «Библиотека программиста»).
* Свейгарт, Эл, Учим Python,делая крутые игры / Эл Свейгарт; [пер. с англ. М. А. Райтмана]. – Москва: Эксмо, 2018. – 416 с. – (Мировой компьютерный бестселлер).
* Абдулаев М., Движущиеся картинки Владимира Зворыкина [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://tass.ru/lyudi-i-veschi/6820038
* Академия дизайна, Виды компьютерной графики [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://sansans.ru/blog/illyustrator/vidy-kompyuternoj-grafiki.html
* Малышева А., Компьютерная графика что такое? Виды компьютерной графики [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://fb.ru/article/190005/kompyuternaya-grafika-chto-takoe-vidyi-kompyuternoy-grafiki
* Студенческая библиотека онлайн, Общие сведения графических систем [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://studbooks.net/2178722/informatika/obschie\_svedeniya\_graficheskih\_sistem
* TashaFridrih, Форрестер и его «Вихрь» [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://habr.com/ru/company/ua-hosting/blog/390685/
* Учебные материалы онлайн, Компьютерная графика и ее применение во всех сферах человеческой деятельности [Электронный ресурс] – Электрон. Дан. – Режим доступа: https://studwood.ru/1622419/informatika/oblasti\_primeneniya\_kompyuternoy\_grafiki
* Хелпикс.Орг, История. Первые вычислительные машины [Электронный ресурс] - Электрон. Дан. – Режим доступа: https://helpiks.org/5-61421.html