ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

*«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

Институт машиностроения, материалов и транспорта

Высшая школа автоматизации и робототехники

**Курсовой проект**

по дисциплине «Объектно-ориентированное программирование»

**«Ray Casting алгоритм»**

Пояснительная записка

|  |  |  |
| --- | --- | --- |
| Выполнил  студент гр. 3331506/10401 | *(подпись)* | Арсланов М. И. |
| Работу принял | *(подпись)* | Ананьевский М.С. |

Санкт-Петербург

2024 г.

# Оглавление

[Оглавление 2](#__RefHeading___Toc1073_1567543617)

[Техническое задание 3](#__RefHeading___Toc1075_1567543617)

[1. Введение 4](#__RefHeading___Toc1077_1567543617)

[2. Ход работы 6](#__RefHeading___Toc1079_1567543617)

[2.1. Работа с консолью. 6](#__RefHeading___Toc1081_1567543617)

[2.2. Работа с векторами. 7](#__RefHeading___Toc1087_1567543617)

[2.3 Работа с геометрическими фигурами. 10](#__RefHeading___Toc1089_1567543617)

[2.4. Работа со сценой. 12](#__RefHeading___Toc1091_1567543617)

[Заключение 15](#__RefHeading___Toc1103_1567543617)

[Список литературы 16](#__RefHeading___Toc1105_1567543617)

# Техническое задание

Реализовать метод рендеринга компьютерной графики «бросание лучей» (*ray casting*). Для вывода изображения, использовать терминал xterm в Ubuntu 22.04.

# 1. Введение

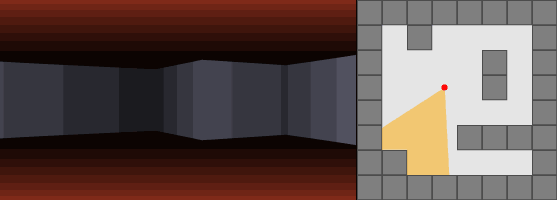
**Рендеринг** или **отрисовка** — термин в компьютерной графике, обозначающий процесс получения изображения по модели с помощью компьютерной программы.

**Методы рендеринга:**

* Растеризация (*rasterization*) совместно с методом сканирования строк (Scanline rendering). Визуализация производится проецированием объектов сцены на экран без рассмотрения эффекта перспективы относительно наблюдателя.
* Ray casting (*“бросание лучей”*). Сцена рассматривается, как наблюдаемая из определённой точки. Из точки наблюдения на объекты сцены направляются лучи, с помощью которых определяется цвет пиксела на двумерном экране. При этом лучи прекращают своё распространение (в отличие от метода обратного трассирования), когда достигают любого объекта сцены либо её фона. Возможно использование каких-либо очень простых способов добавления оптических эффектов. Эффект перспективы получается естественным образом в случае, когда бросаемые лучи запускаются под углом, зависящим от положения пикселя на экране и максимального угла обзора камеры.
* Трассировка лучей (*ray tracing*) похожа на метод бросания лучей. Из точки наблюдения на объекты сцены направляются лучи, с помощью которых определяется цвет пиксела на двумерном экране. Но при этом луч не прекращает своё распространение, а разделяется на три луча-компонента, каждый из которых вносит свой вклад в цвет пикселя на двумерном экране: отражённый, теневой и преломлённый. Количество таких компонентов определяет глубину трассировки и влияет на качество и фотореалистичность изображения. Благодаря своим концептуальным особенностям, метод позволяет получить очень фотореалистичные изображения, однако из-за большой ресурсоёмкости процесс визуализации занимает значительное время.

**Где может использоваться рейкастинг?**

Рейкастинг можно использовать по-разному, особенно в трёхмерном пространстве. Я выделю три наиболее важных по моему мнению применения, часто встречающихся в игровых 2D-движках. Самой известной игрой, использующей эту технику, является Wolfenstein 3D. Лучи в ней трассировались для определения ближайших объектов, а их расстояние от позиции игрока использовалось для правильного масштабирования.

Рисунок 1. Простая иллюстрация рейкастинга в играх.

# 2. Ход работы

## 2.1. Работа с консолью.

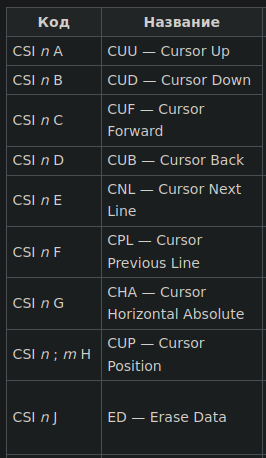
Для начала, напишем класс Console, который будет отвечать за все операции с терминалом xterm.

Базовые функции, которые нам нужны:

* Установка четких размеров окна терминала.
* Очистка терминала
* Спрятать/показать курсор
* Установить курсор в определенное положение окна.

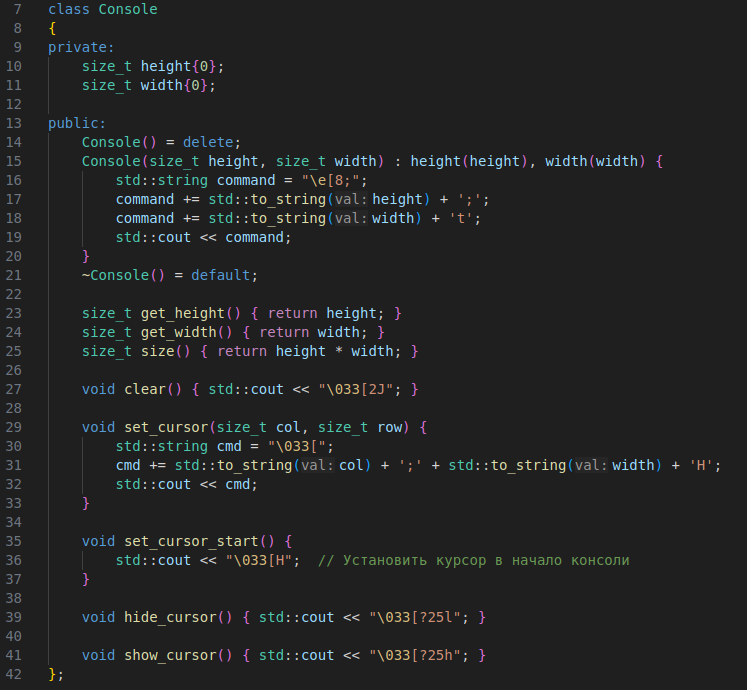
Для реализации данных методов, я использовал управляющие символы ANSI (*ANSI escape code*) - символы, встраиваемые в текст, для управления форматом, цветом и другими опциями вывода в текстовом терминале.

Ниже приведены некоторые команды, отвечающие за управления терминалом.

Рисунок 2. Некоторые управляющие последовательности ANSI (неполный список).

Например, чтобы установить курсор в определенное место в терминале, в поток вывод необходимо записать вот такую команду:

«\033[col;rowH», где col и row — колонна и ряд соответственно.

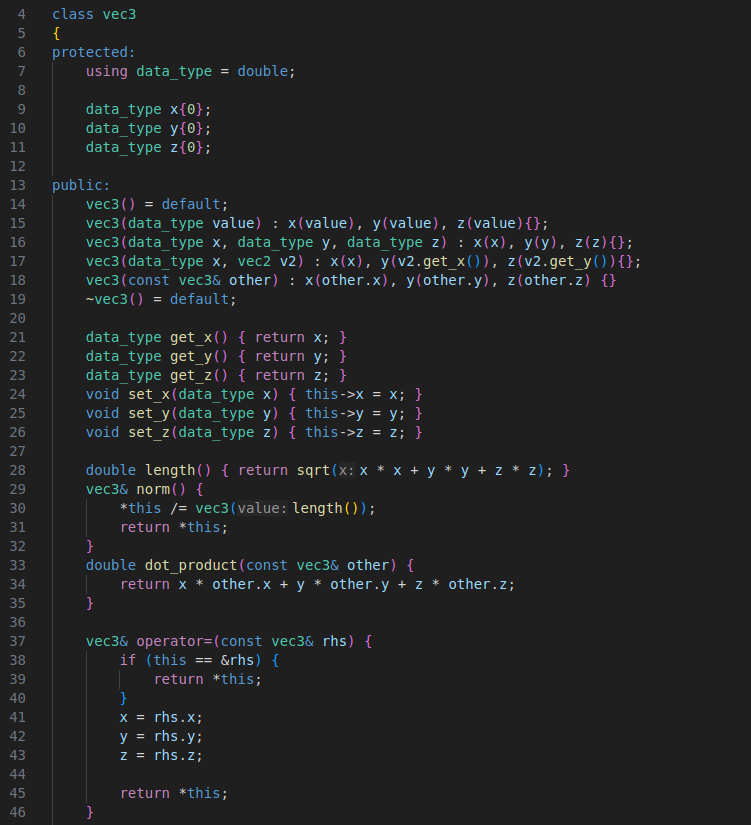
Рисунок 3. Реализация класса Console.

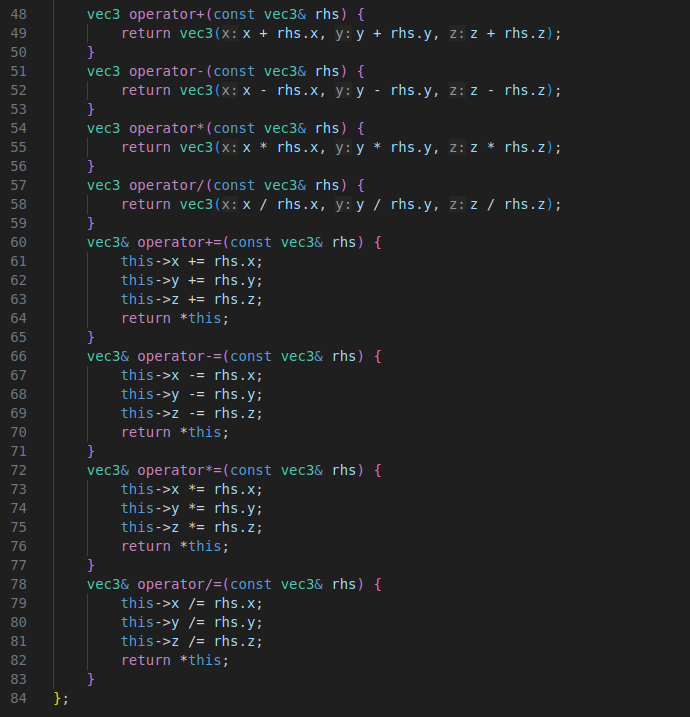
## 2.2. Работа с векторами.

Основа данного проекта — умение работать с векторами в двух и трехмерных пространствах. Напишем соответсвующие классы для двух и трехмерного вектора соответсвенно. Определим базовые методы векторов:

* Длина вектора
* Нормализация вектора
* Скалярное произведение двух векторов.
* А также переопределение некоторых операторов.

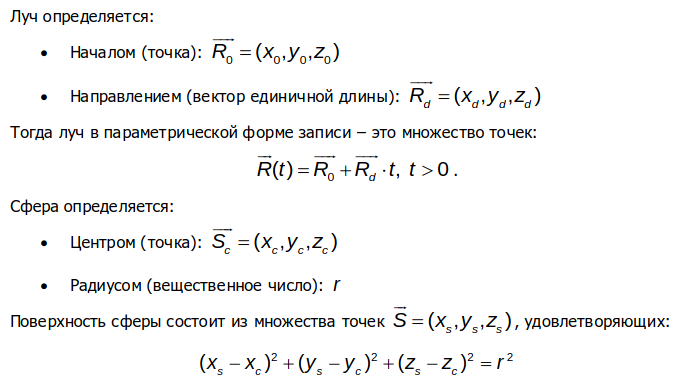
Ниже представлена реализация трехмерного вектора.

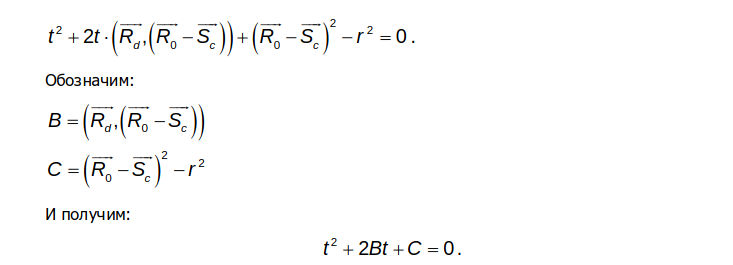


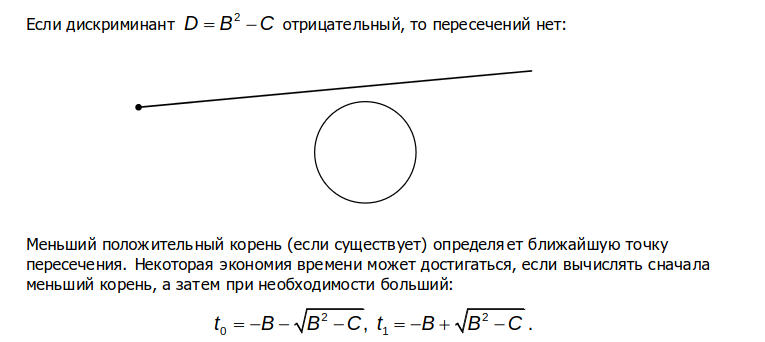
Рисунок 4. Реализация трехмерного вектора

## 2.3 Работа с геометрическими фигурами.

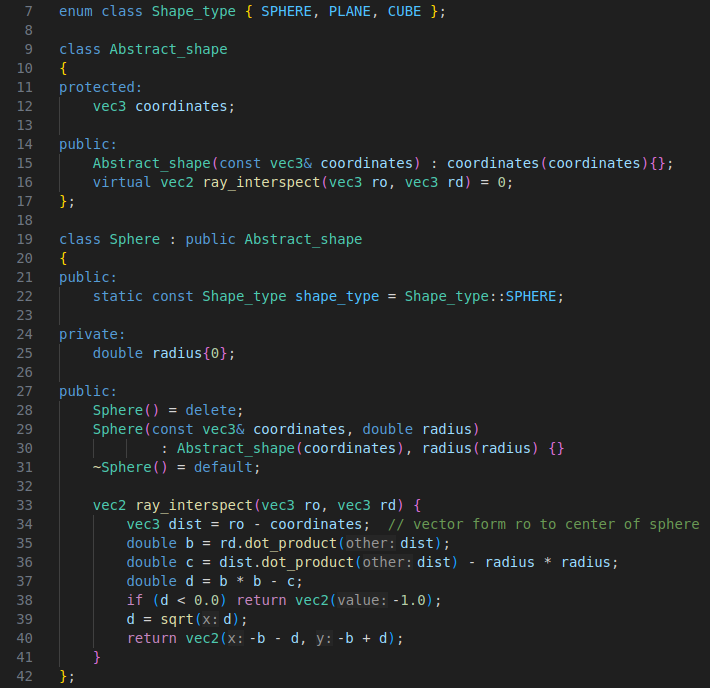
На сцене мы будет отрисовывать простейшие геометрические фигуры. Начнем со сферы. Класс фигуры будет содержать в себе координаты, габариты (в случае сферы — ее радиус) и функцию, определяющую пересечения луча со сферой, в зависимости от начального положения луча и его направления.

 Подставим точку луча в уравнение сферы и решим относительно t:





 Меньший положительный корень (если существует) определяет ближайшую точку пересечения. Некоторая экономия времени может достигаться, если вычислять сначала меньший корень, а затем при необходимости больший:

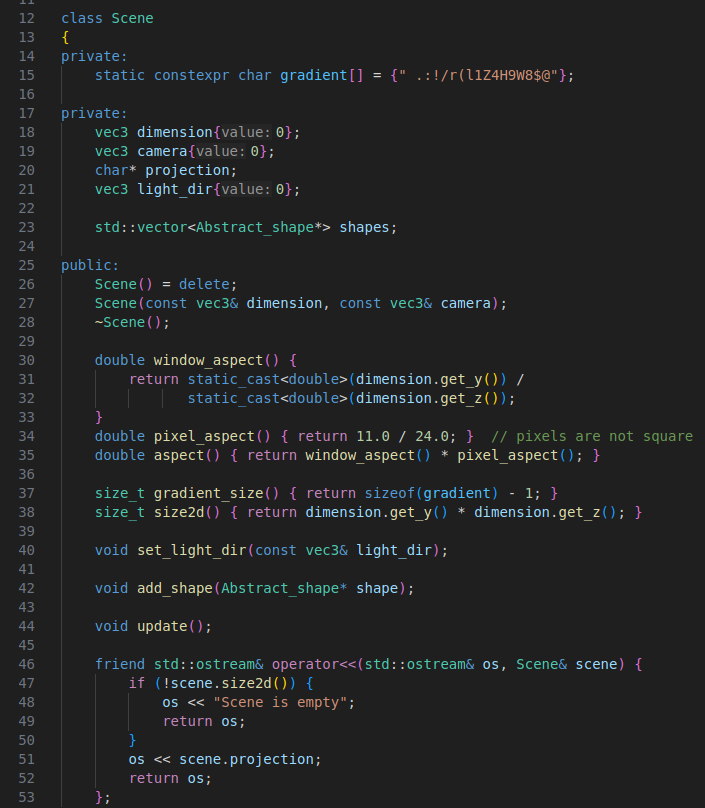
Рисунок 5. Реализация класса Sphere

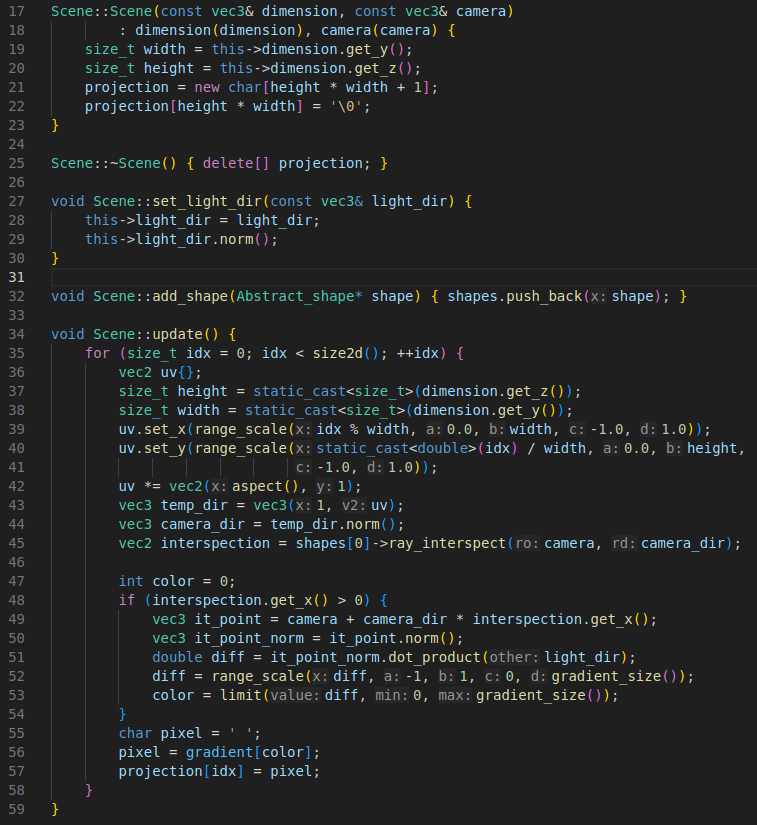
## 2.4. Работа со сценой.

Все наши геометрические фигуры будут располагаться на сцене. Создадим массив указателей на абстрактный класс для хранения разных геометрических фигур на нашей сцене. Еще нам понадобиться градиент символов для эмуляции света. Символ с наибольшим кол-вом символов самый яркий. Символ с наименьшим количеством света самый темный. Также определим на сцене расположение камеры и направление источника света.

Класс сцена содержит в себе функции:

* Добавление новой геометрической фигруы.
* Обновление отрисовки. По сути это основная функция, которая в зависимости от текущего набора геометрических фигур, направления света и направление камеры, отрисовывает картинку в терминале.

Рисунок 6. Scene.hpp

Рисунок 7. Scene.cpp

# Заключение

В ходе выполнения курсовой работы были получены знания о методах рендеринга компьютерной графики. В частности, реализован метод ray\_casting. Также получены навыки работы с трехмерными векторами и работы с терминалом xterm.

# Список литературы

1. https://invisible-island.net/xterm/ctlseqs/ctlseqs.html

2. <http://nsucgcourse.github.io/lectures/Lecture13/Slide_13_Valeev_Rays.pdf>

3. https://habr.com/ru/articles/436790/

4. http://ray-tracing.ru/articles245.html