**Проект на тему «OpenGL. Pineapple»**

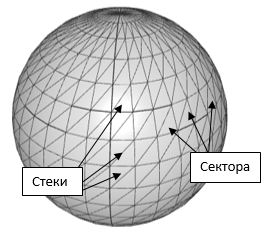
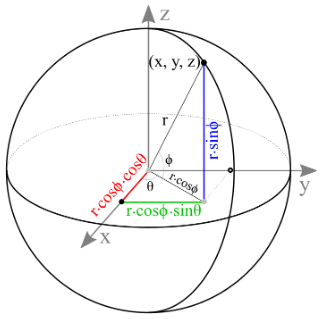
**Выполнила: Бидник Наталья, группа КС-18-1**

**Постановка задачи**

Создать графический объект в среде разработки Microsoft Visual Studio 2017 с помощью графической библиотеки «OpenGL». В этой работе будет создана модель ананаса.

**Математические вычисления**

Фигура была создана с помощью деформированной сферы был создан цилиндр, его верх и низ сделаны выпуклыми. Поскольку мы не можем нарисовать все точки на сфере, мы выбираем только ограниченное количество точек путем деления сферы на сектора(долгота) и стеки(широта). Затем необходимо соединить эти выбранные точки вместе, чтобы сформировать поверхность.

Стеки и секторы на сфере Расположение координатных осей

Координаты каждой точки рассчитываются так:

Диапазон углов секторов составляет от 0о до 360о, а стеков – от -90о до 90о.

Для того, чтобы получить деформированную сферу, задаются такие значения:





Также задаются цвета:



**Реализованные классы**

В этой работе присутствуют такие классы:

* Shader
* Buffers
* ObjectCoords
* Render

**Диаграмма классов**

Shader

+ Program: GLuint

+ Shader(vertexPath: const GLchar\*, fragmentPath: const GLchar\*)

+ Use(): void

ObjectCoords

x, sectorAngle, stackAngle, … , sectorStep, stackStep: float

+ vertices: vector <GLfloat>

+ indices: vector <GLuint>

+ ObjectCoords ();

+ ObjectCoords (sectorCount: int, stackCount: int, Rad: float)

+ Hemisphere (sectorCount: int, stackCount: int, Rad: float): void

+ Greens0 (sectorCount: int, stackCount: int, Rad: float): void

…

Render

WIDTH = 800, HEIGHT = 800: const GLuint

vertexShader, fragmentShader: GLuint

+ window: GLFWwindow\*

+ VAO: GLuint

+ Render()

+ GameLoop( Obj: Buffers, n: const int, ObjCoord: ObjectCoords\*): void

Buffers

vertexShader, fragmentShader, success, infoLog[512]: GLuint

+ window: GLFWwindow\*

+ VBO[7], VAO[7], EBO[7], … : GLuint

+ CreateNode (Obj: ObjectCoords\*, n: const int): void

**Описание классов**

**Класс** Shader

*Назначение класса:* считывание шейдера из файла, его компиляция, линковка, проверка на ошибки.

Методы и переменные класса:

Shader(const GLchar\* vertexPath, const GLchar\* fragmentPath)

const GLchar\* vertexPath – путь к файлу, в котором хранится текст вершинного шейдера

const GLchar\* fragmentPath - путь к файлу, в котором хранится текст фрагментного шейдера

Конструктор класса – считывает файл шейдера, используются стандартные потоки С++, собирает шейдерную программу.

Входные данные: текстовые файлы шейдеров

Выходные данные: шейдерная программа

void Use()

Запускает шейдерную программу

Входные данные: шейдерная программа

**Класс** ObjectCoords

*Назначение класса:* генерация индексов и координат вершин каждого элемента фигуры

ObjectCoords(int sectorCount, int stackCount, float Rad)

Конструктор класса – инициализация переменных.

int sectorCount – количество секторов

int stackCount – количество стеков

float Rad – радиус

void ObjectCoords ::Hemisphere(int sectorCount, int stackCount, float Rad)

Метод генерирует индексы и координаты вершин поверхности ананаса.

Входные данные:

int sectorCount – количество секторов

int stackCount – количество стеков

float Rad – радиус

Выходные данные:

std::vector <GLfloat> vertices – вектор вершин

std::vector <GLuint> indices – вектор индексов

void LowPart(int sectorCount, int stackCount, float Rad);

Метод генерирует индексы и координаты вершин для нижней части ананаса

Входные данные:

int sectorCount – количество секторов

int stackCount – количество стеков

float Rad – радиус

Выходные данные:

std::vector <GLfloat> vertices – вектор вершин

std::vector <GLuint> indices – вектор индексов

void Greens0(int sectorCount, int stackCount, float Rad);

void Greens(int sectorCount, int stackCount, float Rad);

void Greens2(int sectorCount, int stackCount, float Rad);

void Greens3(int sectorCount, int stackCount, float Rad);

void Greens4(int sectorCount, int stackCount, float Rad);

Методы генерируют индексы и координаты вершин листвы ананаса

Входные данные:

int sectorCount – количество секторов

int stackCount – количество стеков

float Rad – радиус

Выходные данные:

std::vector <GLfloat> vertices – вектор вершин

std::vector <GLuint> indices – вектор индексов

const unsigned int\* getIndices()

Метод возвращает индексы каждого объекта фигуры

**Класс** Buffers

*Назначение класса:* создание вершинного, фрагментного и индексного буфера.

CreateNode(ObjectCoords\* Obj, const int n)

Метод создает буферы для каждого объекта.

Входные данные:

ObjectCoords\* Obj – координаты вершин объекта

const int n – количество объектов

Выходные данные:

GLuint VBO [7], VAO [7], EBO [7] – массивы буферов

**Класс** Render

*Назначение класса:* создание окна, матрицы вида, вращения объекта вокруг своей оси, отрисовка всех элементов.

Render();

Конструктор создает окно, в котором будет производиться отрисовка.

void GameLoop(Buffers Obj, const int n, ObjectCoords\* ObjCoord);

Определяет местонахождение каждого объекта отрисовки, определяет скорость, радиус вращения. Отрисовка всех объектов.

Входные данные:

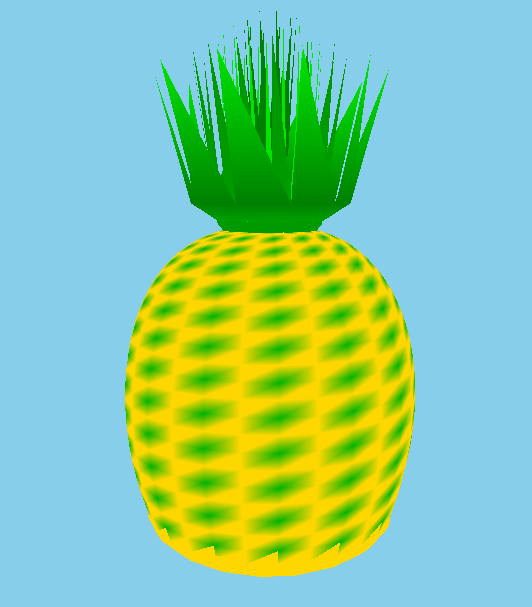
Buffers Obj – объект, сожержащий буферы объектов

const int n – количество объектов, которые нужно отрисовать

ObjectCoords\* ObjCoord – координаты вершин

**Реализация**

В файле Source.cpp создается объект класса Render, далее создается массив объектов класса ObjectCoords, вызываются методы, которых генерируются индексы и кооординаты вершин всех составляющих ананаса, после создается объект класса Buffers и вызывается метод этого класса, который генерирует буферы для всех частей фигуры. И последним шагом является вызов метода GameLoop класса Render, в котором происходит отрисовка фигуры. На выходе получаем такое изображение:



**Возникшие проблемы в ходе выполнения проекта**

1. По началу было непонятно из каких частей создавать ананас, особенно листву. Я решила просто деформировать сферу, где-то меняя радиус по оси У, чтобы фигура вышла более вытянутая, где-то просто ее недостраивать(так была построена нижняяя часть фигуры). Листва создана также на базе сферы, просто выбрано малое количество стеков и большая длина секторов, отрисована нижняя половина сферы, и это выглядит как резная листва.
2. Из-за сложности подключения текстур пришлось так окрасить нижнюю часть ананаса: вершины секторов и стеков были окрашены таким образом: желтый, зеленый, желтый и т.д. В результате образовались зеленые многоугольники на желтом фоне, что и напоминает ячейки ананаса.