AURUS_test_task

Тестовое задание на должность инженера программиста. ООО АУРУС

Сборка и запуск проекта

1. с использованием cmake (тестировалось на archlinux)
    1. `mkdir build`
    2. `cd build`
    3. `cmake ..`
    4. `cmake --build .`
    5. `./AURUS_RUN ../example.flt`
2. с использование docker (тестировалось на archlinux)
    1. `docker build -t aurus_test .`
    2. `docker run aurus_test`

О программе

- Программа считывает записи из OpenFlight файла и ищет в них следущие записи: Header record, Group Record, Object record, Face record.
- Расшифровывает поля ID, Material Index, Color name index
- Выводит их в консоль

Т.к. в файле были записи, ID которых не поместились в стандартное поле записи (polygon1 polygon2 polygon3) дополнительно ищутся и расшифровываются Long ID record

Исходный код разделен на 3 блока:

1. Чтение данных из файла в оперативную память (см. FileReader), записи разделяются и хранятся как массив указателей на std::byte

2. Классы, представляющие тип записи и расшифровывающие указатель на std::byte, возращающие переменную (например HeaderNode получает std::byte Heder записи и позволяет легко получить значение полей)

3. Класс, получающий на вход вектор двоичных представлений записей и возращающий вектор объектов представлений соответстующих записям из п.2

Такая архитектура позволяет легко расширить функционал программы т.к. для добавления возможности парсить новые типы записей достаточно создать класс из п.2 и добавить инструкции для его создания в фабричный метод класса из п.3

Запись хранится в виде указателя на std::byte, такой способ хранения выбран в силу специфики std::byte (std::byte - cppreference.com)

throw и assert

Исключения в коде выбрасываются там, где случается неконтролируемая программой ошибка (например файл не существует)

assert используется там, где необходимо подтвердить, что объект готов к вызову метода (например перед чтением файл должен быть открыт, если файл не был открыт - это ошибка программиста и assert должен упасть)

Потоки чтения бинарных файлов в стандартной библиотеке c++ не выбрасывают исключений, поэтому в FileReader были добавлены проверки на "удачность чтения файла"