Отчет 3
1. Постановка задачи

Лабораторная №3

Разработка приложения печати графиков.

Исходные входные данные

Исходные данные для печати соответствуют некоторому типу, который определятся пользователем. Данные определенного типа могут о конкретным графиком, который ориентирован на этот тип данных.

Примеры данных.

1. Данные характеризуются парой **[значение, дата]**, хранятся в БД SQLite(архив с файлами прилагается).

   Информация по организации работы с БД SQLite.

   https://habr.com/ru/post/128836/

2. Данные представлены JSON файлом. Формат данных **[значение , дата].**

   Информация по организации взаимодействия с JSON файлами.

   https://doc.qt.io/qt-5/json.html

Дано: предложен начальный вариант архитектуры ПО, в которую требуется внести изменения с целью снижения связности архитектуры принцип внедрения зависимости. Реализация внедрения зависимости с помощью IOC контейнера.

При разработке архитектуры учесть

1. Возможность добавления новых графиков (графики отличаются видом и данными
2. Изменение визуального стиля графиков (цветной, черно белый).

Общие требования к GUI

1. Загружаем данные, путем выбора нужного файла. Данные в ПО не отображаем, отображаем только график, построенный относи данных.
2. При печати в pdf выбираем место сохранения графика.

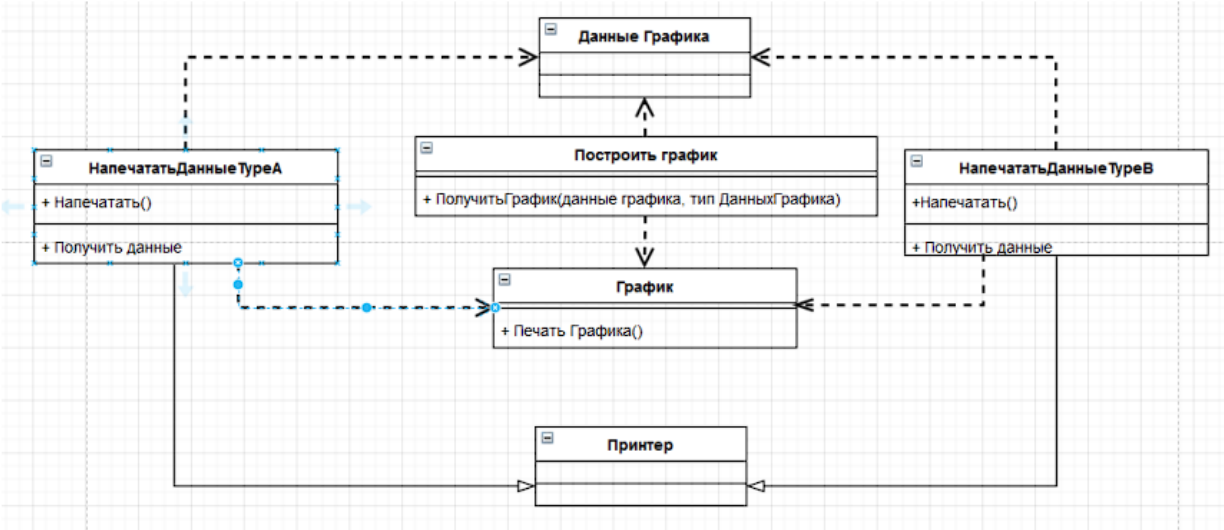
Использование предложенной реализации IOC контейнера на с++

Необходимо разобраться в предложенной реализации IOC контейнера.

Код сопроводить соответствующими объяснениями.

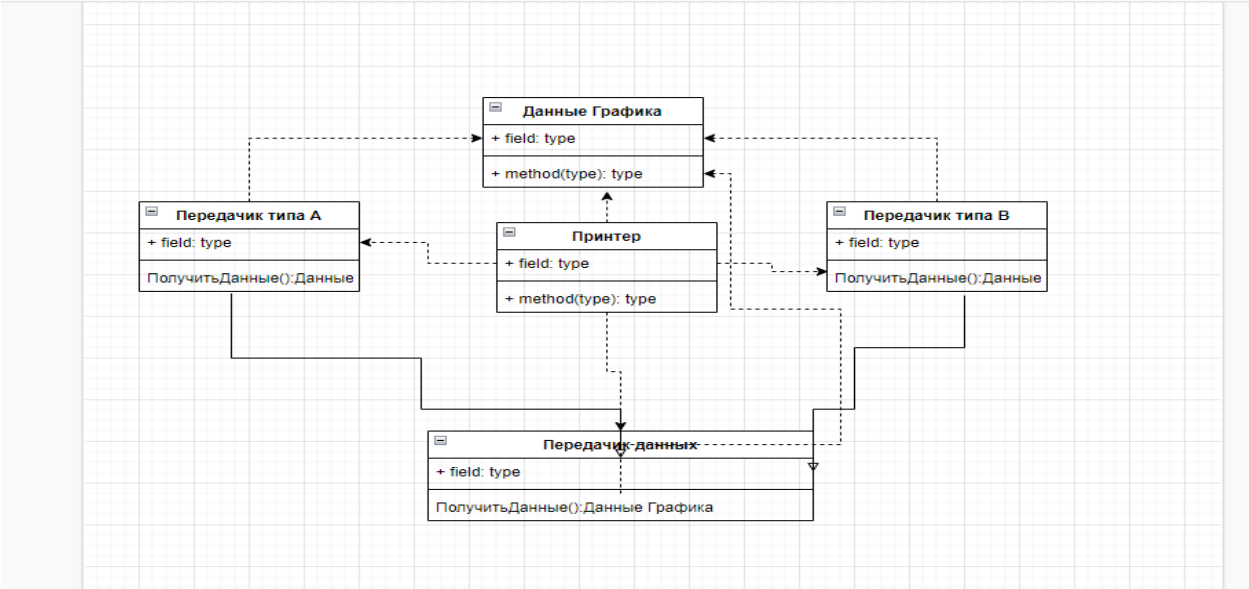
Рассмотреть необходимые темы, используемые при реализации IOC контейнера

2. Предлагаемое решение.



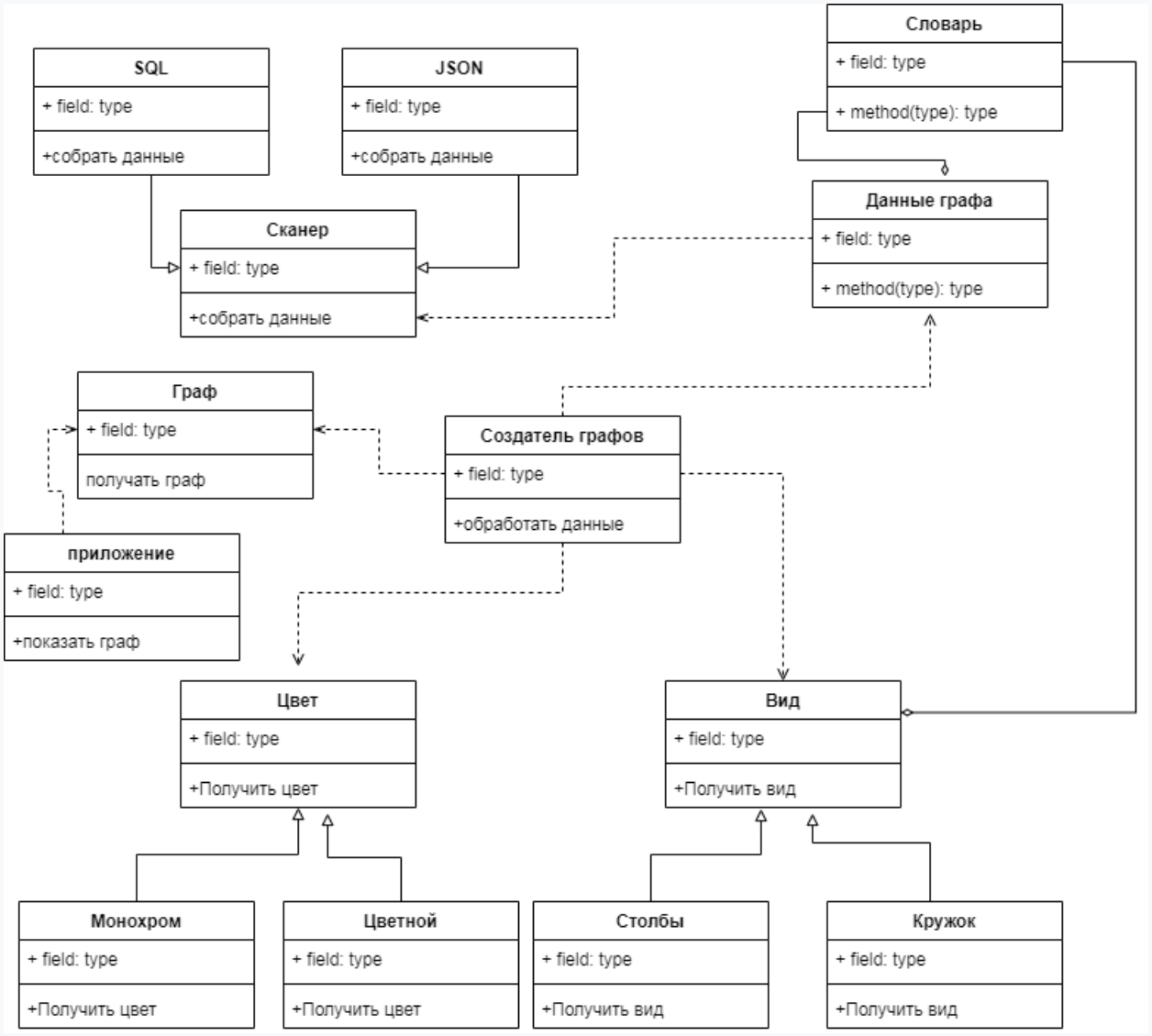
Исходная схема ,которая нужно доработать.

Правильная реализация



В ходе раздумий после реализации своей архитектуры,было обдуманна оптимальная модель графа с учетом Solid.Было принято решение воспользоваться принципом DIP.В класс Данные графа передать данные о графе,вместо печати сделать функции

получение данных,а сам принтер заставлять печатать на основе этих данных,что дало уменьшение компонентов ,то есть уменьшение зависимостей.
Моя реализация:



На своей практике я решил воспользоваться паттерном фабрики.Я создал три продукта цвет ,вид и сканнер,путем контейнера связи между создателем графа и конкретными продуктами ,потому что контейнер помогает убрать их,что улучшает качество по DIP
Шаги разработки:

1. В начале после просмотра данных в таблицах было принято решение создать структура наподобие словаря из Python в которой содержится ключ и значение,где ключ QStrig ,а val float
2. Далее проделана работа по создание класса содержащие данные класса и обработки(DataGraph)
   переменные
   QList <Dictonary> data-хранилища данных графа
   Int elementCounter- количество элементов

Boll isEmpty проверка пустоты
функции
void GotData() обнуляет isEmpty
bool ChechEmpthy() выводит значение isEmpty
void Push(QString header, float val)добавляет данных в лист
QList <Dictonary> GetData() возвращает данных словаря
int GetElementsCount () возращает количество элементов
void SenElemenCount(int count)ставит колличетсво элементов

Разработка сканера реализован через фабрику путем создание интерфейса IScanner и конкретных классов для вывода данных Sqlite и Json. IScanner имеет виртуальную функция DataGraph* GetData(QString path(путь к файлу)) на основе которой формируется выборка данных
GetData Sqlite создается переменная дата граф где будут хранится данные парсера путем подключения базе данных через бибиотек QSqlDatabase QSqlQuery создается переменная dbase где хранятся данные о базе после открываем бд и через QSqlQuery пишем запрос для получения данных бд,потом путем итерации заполняем дату через Push вставляя первое и второе значение бд после ставим на бд пусто и возвращаем данные
GetData Json инициализация даты графа и val значение открытия файли ,считывание его ,перевод формат toUtf8 и преобразование в QJsonDocument ,после преобразование jsonObject,после создание списка имен проходим с помощью foreach по QJsonDocument заполняя дату,после возвращая ее

Далее я решил реализовать интерфейс IColor отвечающий за цвет ,в предложенной задаче есть два варианта черно-белый и цветной. IColor имеет виртуальную функцию QList <QColor>* GetColors(int colorCont(количество цветов)) ,которая отвечает за генерацию цветов в количестве colorCont.Класс Colored генерирует цвета по ргб путем генерации значении от 0-255(R,G,B).Добавляя их лист цветов.Аналогично поступает Monohrom только возвращает оттенок серого цвета.

Класс IView тип графика ,просто заполнение графика путем передачи даты в зависимости от вида PIE or BAR

3. Далее создание класса сборки всех данных в едины и передачи их в приложение(CreatroGraphs).Тут и происходит уменьшение связей с помощью контейнера
4. Начал добавлять sql сканер(почему то на мое пк метод не считывает данные)
5. Создание формы и слотов с учетом условии
   OpenDirectorySlot-открытие директории папки
   PrintCharSlot- сохранение файла в пдф
   ChoseFileSlot-выбор файла из деректории
   RepaintChart полная смена
   RecolorChareSlot Смена цвета
   ChangeTypeSlot смена вида

3 Файлы

CreatroGraphs.h
DataGraph.h
Dictonary.h
Graph.h
Icolor.h/cpp
IOCContainet.h

Iscanner.h/cpp
IView.h
mainwindow.h/cpp

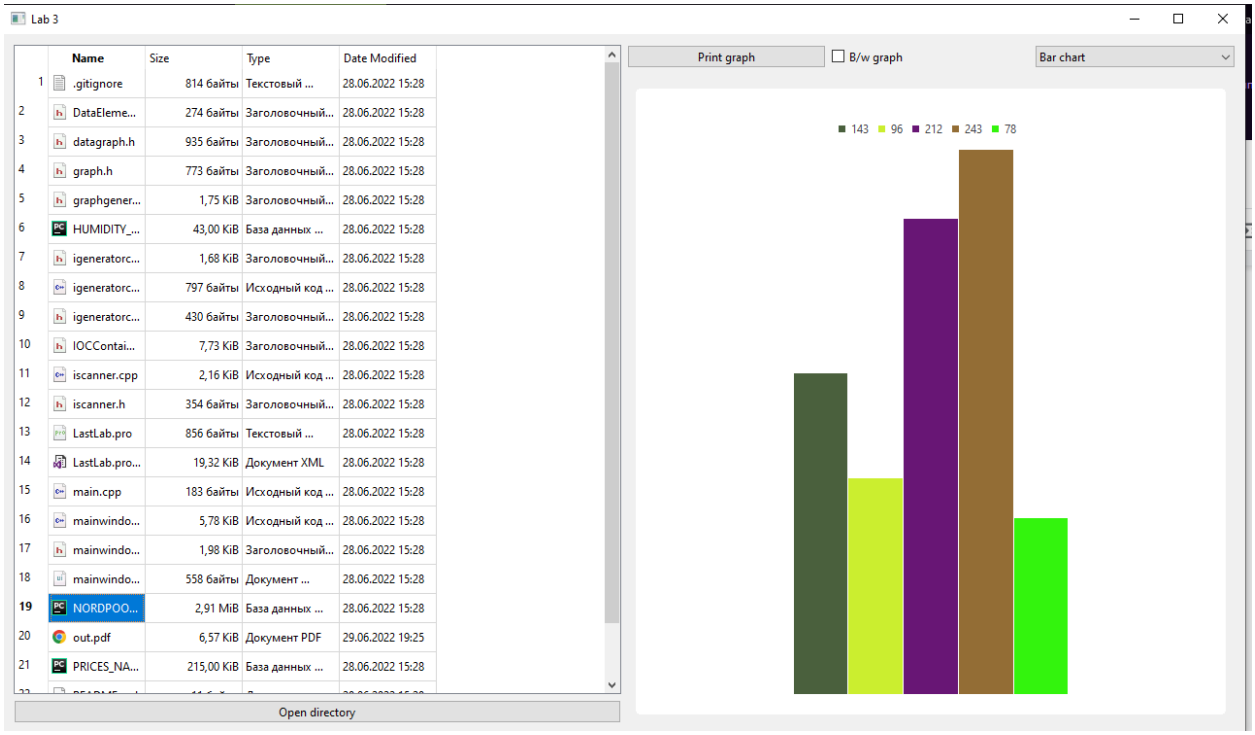
4 Руководства пользователя

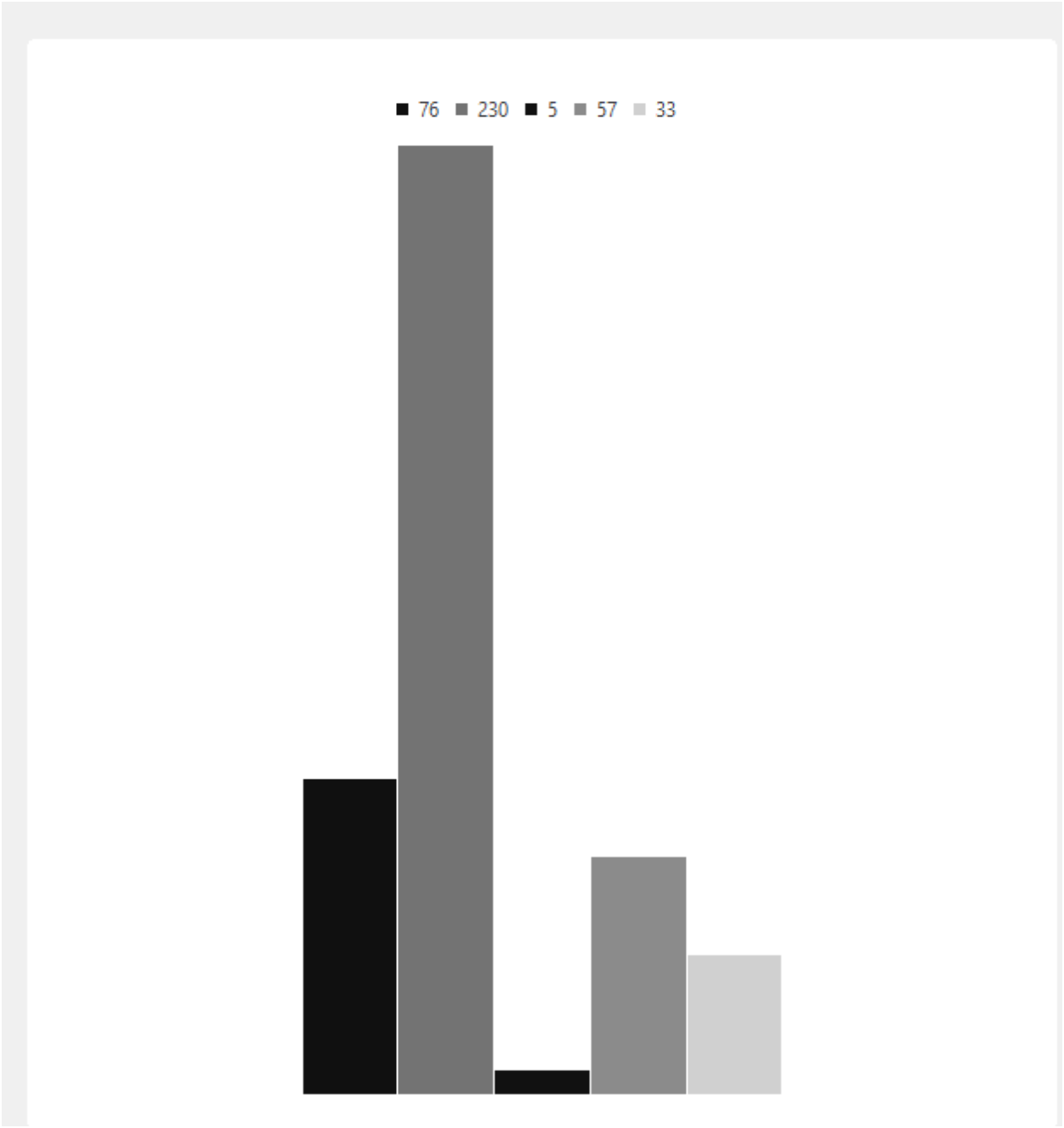
Шаги:

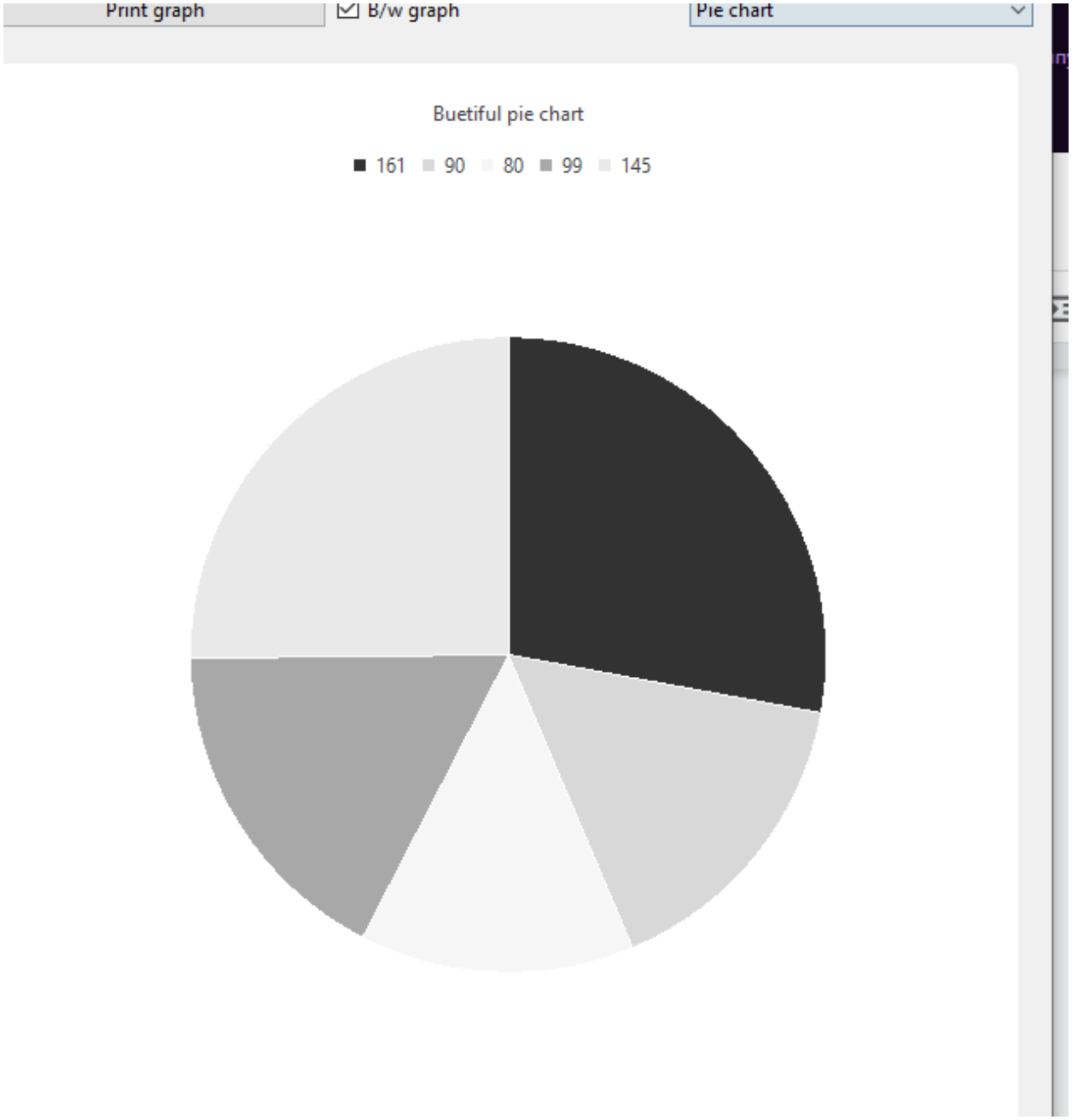
1. Запуск программы
2. Выбрать файл
    a. Если находится вашей то просто наведите на файл
    b. Если нет нажмите на кнопку open directory и выберите папку где лежит файл
3. В правом верхнем углу может выбрать вид графика и цвет
4. После если вам нужен график нажмите на кнопку прин график и он сохранится в дектории под именем out.pdf

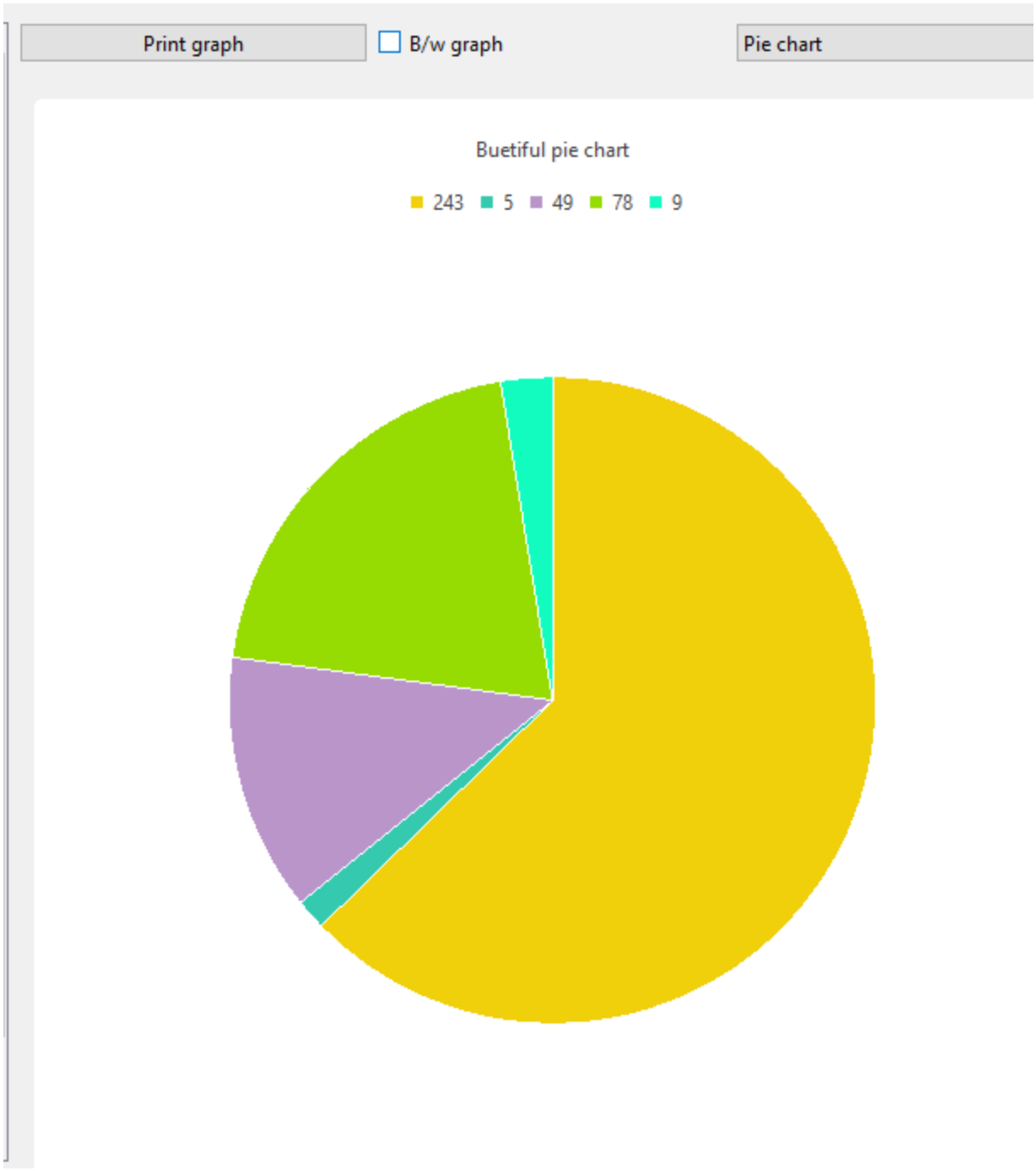
5 Тестирование

SQLite(генерит рандомные данные потому что не читает SQLite)

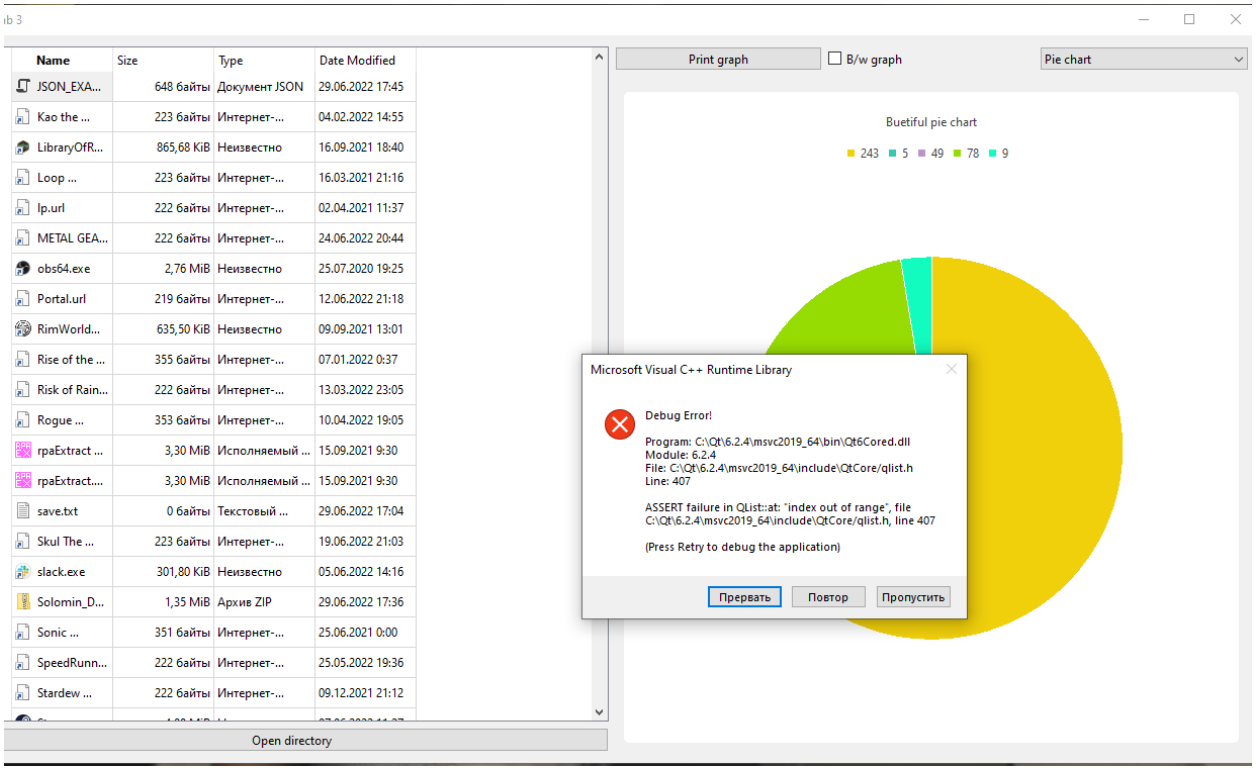



76
230
5
57
33


Print graph
B/w graph
Pie chart
Buetiful pie chart
161
90
80
99
145



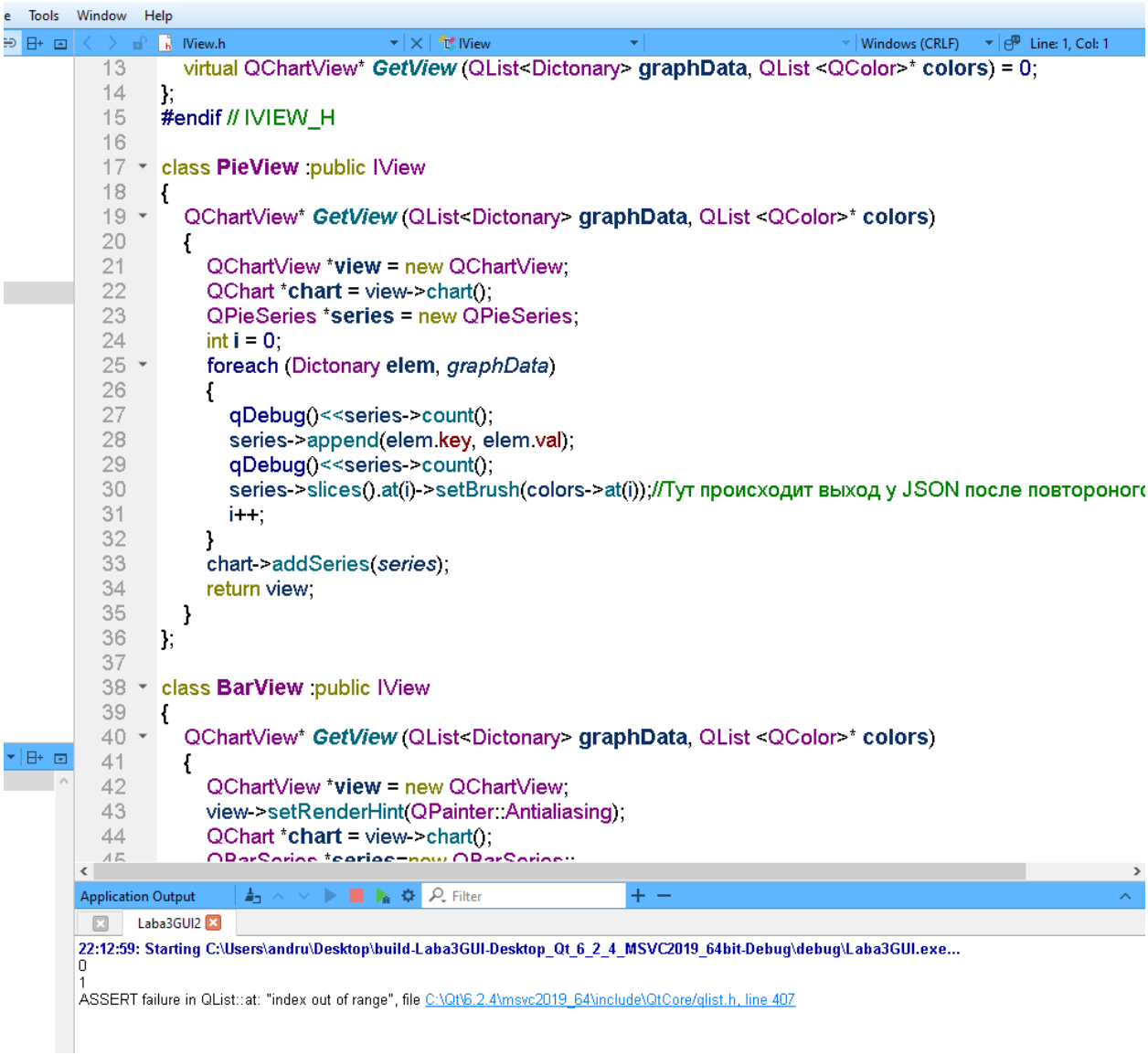
Json:(вылет после смены qt)



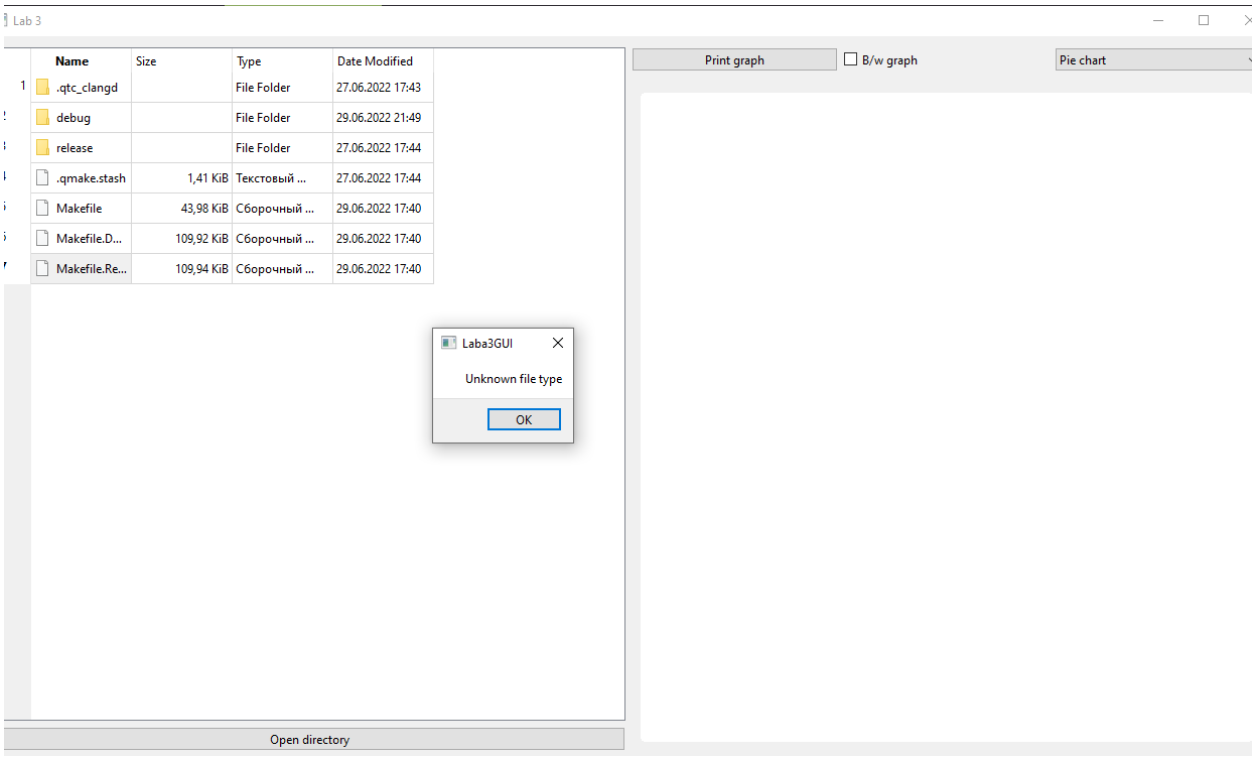
Вылет серии (до определенного момента работал нормально) как SQLITE

```
class PieView :public IView
{
   QChartView* GetView (QList<Dictonary> graphData, QList <QColor>* colors)
   {
      QChartView *view = new QChartView;
      QChart *chart = view->chart();
      QPieSeries *series = new QPieSeries;
      int i = 0;
      foreach (Dictonary elem, graphData)
      {
         series->append(elem.key, elem.val);
         series->slices().at(i)->setBrush(colors->at(i));//Тут происходит выход у JSON после повторонго
         i++;
      }
      chart->addSeries(series);
      return view;
   }
};
```

Хотя компилятор дебаг говорит что есть место



Открытие другого типа файла:


Lab 3
Name
Size
Type
Date Modified
.qtc_clangd
File Folder
27.06.2022 17:43
debug
File Folder
29.06.2022 21:49
release
File Folder
27.06.2022 17:44
.qmake.stash
1,41 KiB
Текстовый ...
27.06.2022 17:44
Makefile
43,98 KiB
Сборочный ...
29.06.2022 17:40
Makefile.D...
109,92 KiB
Сборочный ...
29.06.2022 17:40
Makefile.Re...
109,94 KiB
Сборочный ...
29.06.2022 17:40
Laba3GUI
Unknown file type
OK
Print graph
B/w graph
Pie chart
Open directory