**Opg. 1 – Screendump**

A screenshot of a computer

Description automatically generated with low confidence

**Opg. 1 – Table.h**

#ifndef TABLE\_H

#define TABLE\_H

#include <vector>

#include <ostream>

#include <tuple>

class Table

{

public:

    Table() {}

    void setHeaders(const std::string& h1, const std::string& h2, const std::string& h3);

    void addData(const int i, const std::string& s, const double d);

    void outputData(std::ostream &os);

private:

    std::tuple<std::string, std::string, std::string> mHeaders;

    std::vector<std::tuple<int, std::string, double>> mData;

};

#endif // TABLE\_H

**Opg. 1 – Table.cpp**

#include "Table.h"

#include <iomanip>

#include <string>

#include <sstream>

void Table::setHeaders(const std::string &h1, const std::string &h2, const std::string &h3)

{

    mHeaders = std::make\_tuple(h1, h2, h3);

}

void Table::addData(const int i, const std::string &s, const double d)

{

    mData.push\_back(std::make\_tuple(i, s, d));

}

void Table::outputData(std::ostream &os)

{

    os << "\n"

       << std::get<0>(this->mHeaders)

       << " "

       << std::get<1>(this->mHeaders)

       << std::setw(27) << std::right

       << std::get<2>(this->mHeaders)

       << "\n------------------------------------\n";

    for (std::tuple elem : this->mData)

    {

        std::stringstream stream;

        stream << std::fixed << std::setprecision(2) << std::get<2>(elem);

        std::string fixed\_deci\_string = stream.str();

        os << std::setw(3)

           << std::get<0>(elem)

           << " "

           << std::get<1>(elem)

           << std::setw(32-std::get<1>(elem).size()) << std::right

           << fixed\_deci\_string

           << "\n";

    }

}