Задание 2

Выполнил:	Захаревич Николай Сергеевич
Имя входного файла:	Консоль
Имя выходного файла:	Консоль

1. Курс валют. app.exe —data=daily,xml —from=USD —to=EUR

Формат входного файла

Файл содержит поля NumCode, CharCode, Nominal, Name, Value для каждой валюты

Формат выходного файла

Выводится курс валют, переданных как аргументы командной строки

Пример

Консоль	Консоль
-from=USD -to=EUR	1 USD = 0.86 EUR

main.cpp

```
#include <iostream>
#include <cstdlib>
#include <libxml++/libxml++.h>

enum CurrencyState {
  NOT_PARSED,
  IS_PARSING,
  IS_PARSED,
};

struct Currency {
  Glib::ustring code, name, nominal;
  double value;
  CurrencyState state;
  Currency(Glib::ustring code_) : code(std::move(code_)) {state = NOT_PARSED;}
  Currency() = default;
};


class RateParser : public xmlpp::SaxParser {
private:
  Glib::ustring currentTag;
  Currency from, to;
  void parseCurrency (Currency &currency, const Glib::ustring& chars) {
    if (currentTag == "Nominal") {
      currency.nominal = chars;
    } else if (currentTag == "Name") {
      currency.name = chars;
    } else if (currentTag == "Value") {
      currency.value = std::stod(chars) / std::stod(currency.nominal);
      currency.state = IS_PARSED;
    }
  }
```

```
protected:
  void on_start_element(const Glib::ustring& name,
                        const AttributeList& properties) override {
    currentTag = name;
  }

  void on_characters(const Glib::ustring& chars) override {

    if (from.state == IS_PARSING) {
      parseCurrency(from, chars);
    }

    if (to.state == IS_PARSING) {
      parseCurrency(to, chars);
    }

    if (from.state == NOT_PARSED && chars == from.code) {
      if (currentTag == "CharCode") {
        from.state = IS_PARSING;
      }
    }

    if (to.state == NOT_PARSED && chars == to.code) {
      if (currentTag == "CharCode") {
        to.state = IS_PARSING;
      }
    }
  }

public:
  RateParser() = default;
  RateParser(Currency &from_, Currency &to_) : from(from_), to(to_) {}

  void print_result() {
    if (from.state == NOT_PARSED || to.state == NOT_PARSED) {
      std::cout << "There is no such charcode";
    } else {
      std::cout << "1 " << from.name << " = " << from.value / to.value << " " << to.n
    }

  }

  ~RateParser() override = default;
};

int main(int argc, char** argv) {
  std::locale::global(std::locale(""));

  const short CHAR_CODE_LENGTH = 3;

  char filenameStr[1002];
  char fromStr[CHAR_CODE_LENGTH + 1];
```

```
  char toStr[CHAR_CODE_LENGTH + 1];
  bool areGoodArgs = true;

  for (unsigned int i = 1; i < argc; i++) {
    size_t argLength = strlen(argv[i]);
    if (i == 1 && strncmp(argv[i], "-data=", 6) == 0) {
      strncpy(filenameStr, argv[i] + 6, argLength - 5);
    } else if (i == 2 && strncmp(argv[i], "-from=", 6) == 0 && argLength - 6 == CHAR
      strncpy(fromStr, argv[i] + 6, CHAR_CODE_LENGTH);
    } else if (i == 3 && strncmp(argv[i], "-to=", 4) == 0 && argLength - 4 == CHAR_C
      strncpy(toStr, argv[i] + 4, CHAR_CODE_LENGTH);
    } else {
      areGoodArgs = false;
    }
  }

  if (areGoodArgs) {
    Glib::ustring filename(filenameStr);
    Currency from(fromStr), to(toStr);
    RateParser parser(from, to);
    parser.parse_file(filename);
    parser.print_result();
  } else {
    std::cerr << "Bad arguments";
  }

  return 0;
}
```