# Общая информация

Программа names\_man выполняет формирование ФИО по заданному списку имен и проверку правильности поданных на вход ФИО. Names\_man — консольное приложение с использованием Qt.

Формат командной строки:

Names\_man.exe vocabulary –o optOutFile –I optInFile –g optLogFile

Vocabulary — входной файл с описанием имен. На основе этой информации генерируется список ФИО и выполняется проверка поданных ФИО.

optOutFile — выходной файл, в который записывается список сгенерированных ФИО

optInFile — входной файл со списком ФИО для проверки

optLogFile — выходной файл с результатами проверки ФИО, заданных в optInFile.

Алгоритм работы программы

Из входного файла vocabulary читается список имен.

Если задано имя выходного файла optOutFile, программа генерирует список всевозможных ФИО и сохраняет его в файле.

Если задано имя optInFile, то читает из этого файла список ФИО и выполняет их проверку. Сообщения о результатах проверки записываются в optLogFile, или выводятся на консоль, если optLogFile не задан.

# Описание заданий

## Задание 1. Чтение входного файла со списком имен и генерация списка ФИО

Следует прочитать входной файл vocabulary. Файл содержит список имен и опциональную информацию о каждом имени. Чтение должно выполняться функцией

VecNameInfo ReadNameInfo( QTextStream& stream )

(см. read\_name\_info.cpp) из входного потока stream. Функция возвращает массив записей с именами.

Формат входного фала:

Файл состоит из записей об именах. Каждая запись начинается с новой строки и состоит из обязательного имени и необязательного списка свойств, следующего за именем и заключенным в фигурные скобки.

Пример:

Федор

Илья

{

surname=Ильин

fathername\_m = Ильич

fathername\_f = Ильинична

gender = m

}

Первая запись состоит только из имени “Федор”. Вторая запись “Илья” состоит из имени и списка свойств.

В файле допустимы пустые строки. Они должны игнорироваться.

Синтаксис записи свойства в списке свойств:

propertyName = value

Где propertyName — имя свойства, value — его значение. При чтении свойств необходимо учитывать следующие правила.

1. Каждое свойство начинается с новой строки.
2. В начале строки может быть произвольное число пробелов.
3. Перед и после знака равенства может быть произвольное число пробелов.
4. Порядок следования свойств в списке свойств произвольный.
5. Все свойства необязательны
6. Регистр букв имен свойств не учитывается (surname=Surname=SURNAME).

Программа должна распознавать следующие свойства:

1. Surname — Фамилия мужского рода, образованная от имени.
2. Fathername\_m — образованное от имени отчество мужского рода
3. Fathername\_f — образованное от имени отчество женского рода
4. Gender — род имени. Допустимые значения m и f.

Пример входного файла в репозитории: names.txt.

Дополнительно к этому необходимо реализовать генерацию ФИО в функции

VecNameRecord GenerateNameRecords( const VecNameInfo& info )

На вход функции подается массив с данными об именах (например, прочитанный из файла на предыдущем шаге). Возвращаемое значение — массив ФИО сгенерированный из этих имен по принципу “каждый с каждым”.

## Задание 2. Структура с информацией об имени NameInfo

Следует реализовать класс NameInfo, обеспечивающий получение информации об имени. Перед использованием объект класса NameInfo инициализируется явно именем с помощью метода

void SetName( const QString& name );

после чего у него можно запросить следующие свойства имени

1. Name — собственно, имя
2. Surname — производная от имени фамилия
3. Fathername — производное от имени отчество
4. Gender — род имени.

Эти данные могут быть явно установлены в объект с помощью соответствующих Set-методов, или вычислены на основании синтаксического анализа поданного имени (именно это и является центральной задачей задания).

## Задание 3. Верификация ФИО, чтение/запись списка ФИО

Следует реализовать проверку заданного массива ФИО в функции

void VerifyNames( const VecNameRecord& vrec, const VecNameInfo& info, QTextStream& log )

vrec — массив ФИО для проверки

info — массив с данными об именах, известных программе.

log — поток для вывода сообщений об ошибках.

Функция должна перебрать массив vrec и по каждому ФИО выполнить ряд проверок, записывая диагностику в поток log.

Проверка ФИО включает в себя следующие этапы.

1. Тестирование имени.
   1. Поиск имени в vrec. Если запись об имени найдена, то известен пол человека. Если запись не найдена, нужно выдать в log предупреждение «неизвестное имя name» и определить пол эвристически на основании анализа окончания имени.
2. Тестирование фамилии
   1. Поиск фамилии в vrec. Если удалось найти, проверить соответствие родов фамилии и имени. В случае несоответствия вывести диагностическое сообщение.
   2. Если фамилию найти не удалось, вывести сообщение «неизвестная фамилия surname» и определить ее род эвристически на основании анализа окончания фамилии. В случае несоответствия вывести диагностическое сообщение.
3. Тестирование отчества. Аналогично фамилии.

Дополнительно следует реализовать функцию чтения массива ФИО из файла

VecNameRecord ReadNameRecords( QTextStream& stream )

Формат файла предельно прост, смотри to\_verify.txt. Файл может содержать пустые строки, их необходимо игнорировать. Между элементами ФИО может содержаться произвольное число пробелов и символов табуляции.

Также следует реализовать функцию записи массива ФИО в файл в том же формате

void WriteNameRecords( QTextStream& stream, const VecNameRecord& vrec )