Оглавление

[Тема 1](#_Toc62544484)

[Преамбула 1](#_Toc62544485)

[Контроль версий 1](#_Toc62544486)

[Задачи 1](#_Toc62544487)

[Отчет 3](#_Toc62544488)

# Тема

Разработка класса для алгоритма Вагнера-Фишера. Редакционное предписание определения различий между файлами. Работа с динамической памятью.

# Преамбула

В этом задании вы разработаете полнофункциональный класс, который реализует алгоритм Вагнера-Фишера, рассмотренный в лекции 4. Примените его для определения различий между файлами.

Прежде чем приступать к выполнению задания необходимо прочесть часть лекции относительно близости строк и алгоритма Вагнера-Фишера.

## Контроль версий

В системах контроля версий (*часто* *Git*) отслеживаются изменения между версиями текстовых файлов. Если попросить сравнить 2 версии файла, то результатом будет список изменений. Изменения выводятся в том порядке, в котором обнаружены различия в файлах.

Если в новой версии файла не хватает строки, то выводится признак удаления <<< и текст удаленной строки, если в новой версии есть новая строка, то выводится признак вставки >>> и текст новой строк. Если найдена измененная строка, то выводится признак замены *rrr* и текст новой и прежней строки.

Вам предстоит реализовать ключевую часть этого функционала – реализовать алгоритм Вагнера-Фишера для определения расстояния между файлами и построить редакционное предписание для преобразования одного файла в другой.

# Задачи

1. Начните новый консольный проект FileCompare. Подключите возможность консольного ввода-вывода.
2. Добавьте в проект класс WF. Его удобно сделать inline, т.е. полностью определить в заголовочном файле.
3. В алгоритме используется целочисленная матрица …. Поэтому, в класс необходимо включить указатель на массив целых (int \* pm) и целочисленные переменные для хранения размерностей – M и N.

Как при этом должны измениться конструктор и деструктор?

При заполнении матрицы нам понадобятся массив строк файлов. Их можно сохранять в контейнерах vector<string>. Например, *vector<string> prevVersion, curVersion*;  
Разумеется, для этого нужно включить модули *<vector>* и *<string>* стандартной библиотеки.  
Команда *using* позволит сократить писанину.

1. Создайте объект класса WF в main и убедитесь, что проект собирается.
2. Матрица и строки будут заполняться в методе   
    void Compare(istream& prev, istream& current).   
   Добавьте его и пусть метод пока остается пустым.

Убедитесь, что проект по-прежнему собирается.

1. Реализуйте метод *Compare*. Пусть он пока читает построчно входные потоки и заполняет строками файлов подготовленные массивы.

Это можно сделать примерно так:

char buf[512];

while (prev.good())

{

prev.getline(buf, sizeof(buf));

prevVersion.push\_back(buf);

}

Заполните *M* и *N*, вспомните как они связаны с размерами сравниваемых последовательностей…

1. Чтобы удобно было работать с объектами класса WF добавьте в него еще 4 метода:
   1. *int Rows() const* – возвращает количество строк матрицы Вагнера-Фишера;
   2. *int Columns() const* – возвращает количество столбцов матрицы;
   3. *int Get(int i, int j) const* – возвращает значение i-го,j-го элемента матрицы;
   4. *void Set(int i, int j, int val)* – задает значение i-го,j-го элемента матрицы;
   5. *int Distance() const* – возвращает расстояние между последовательностями, которыми проинициализирован объект.

***Прим***. Обычно, матрицы располагаются в памяти по строкам, т.е. сначала все элементы первой строки, затем второй и т.д. Как при этом правильно вычислить номер элемента одномерного блока памяти?

Позаботьтесь и о контроле значений i, j. В случае недопустимых значений применяйте *throw out\_of\_range(“< Имя метода>: Index out of range”)*.

Проверьте, что проект по-прежнему собирается.

1. Проверьте правильность чтения файлов. Для этого подготовьте пару текстовых файлов в каталоге проекта. Можно взять прямо копию файла WF.h, только переименуйте ее в “prev.txt” и “cur.txt”.
   1. Подключите в основном файле модуль #include <fstream>
   2. Свяжите входные потоки с файлами “prev.txt” и “cur.txt”, например так ifstream prev“prev.txt”);
   3. Вызовите метод Compare для объекта класса WF с потоками, связанными с файлами;
   4. Выведите и проверьте количество строк и столбцов. Оно должно соответствовать количеству строк файлов.

Основные технические сложности преодолены! Теперь нужно реализовать алгоритм Вагнера-Фишера. Псевдокод алгоритма описан в лекции 4.

1. Реализуйте алгоритм Вагнера-Фишера в Compare сравнивая уже не символы, а строки файлов.

***Совет***. В алгоритме WF часто сравнивают 3 целых. Удобно реализовать собственный метод *min*, который может выбирать наименьшее сразу из 3-х целых (*min* для двух величин уже есть в библиотеке)!

1. Выведите на консоль расстояние между файлами. Теперь расстояние показывает какое минимальное количество строк нужно изменить в одном файле, чтобы получить другой!

Если файлы тождественные, то расстояние должно быть 0. Добавьте в один из файлов пустую строку в начале, а в другой – в конце. Расстояние должно стать 2.

1. Добавьте в класс и разработайте метод *string Prescription() const*. Он должен возвращать редакционное предписание по приведению одного файла в другой. Алгоритм на псевдокоде для вычисления редакционного предписания приведен в лекции 4.

Убедитесь в правильности предписания на примере ваших файлов.

1. Для выполнения завершающего шага задания нам понадобятся еще два не сложных метода. Каждый из них возвращает строку из массива *prevVersion* или *curVersion* по номеру элемента. Их можно объявить, например, как: *string PrevLine(int i) const* и *string CurLine(int i) const*.

Реализуйте методы. Не забывайте про проверку допустимости значения индекса.

1. Завершающий шаг задания заключается в выводе различий как указано в преамбуле, т.е. примерно так, как это делает система контроля версий. Если нужно вставить строку, то выводится признак вставки - >>> и затем строка, если удалить, то <<< и строка, если изменить, то *rrr*, а затем исходная строка, и за ней измененная (со сдвигом).

Читайте посимвольно редакционное предписание, когда встречается код операции (иной чем ‘.’), выводите соответствующую информацию на консоль.

***Предупреждение***. Это не такая уж тривиальная задача. Пусть в нашем случае текущая версия файла “расположена в таблице по строке (*col* меняет колонку таблицы, т.е. движет по строке)” (***это так?***), а предыдущая - вертикально (*row*). Нужно иметь ввиду, что при вставке по строке смещения нет, а при удалении нет движения по колонке.

Но этом программная часть задание завершена. Удалите временные файлы, заархивируйте проект. Архив будет частью ответа на задание.

# Отчет

Подготовьте небольшой отчет в Word. Отчет должен содержать:

* Формулировку понятия “метрика” в математике и результаты ваших вычислений, подтверждающих, что вычисленное значение является метрикой;
* Описание “на пальцах”, т.е. без кода, алгоритма Вагнера-Фишера;
* Кратко опишите понятие редакционного предписания и как оно использовано для выяснения различий в двух файла.

Отчет и архив проекта поместите как ответ на задание.