

Сравнение функций определения равенства строк

Гончаров М.И. Группа 31-КЭ

Строковые классы стандартной библиотеки C++ позволяют работать со строками как с обычными типами, не создающими проблем для пользователя. Это означает, что строки можно копировать, присваивать и сравнивать как базовые типы, не беспокоясь о возможной нехватке памяти или размерах внутреннего блока, предназначенного для хранения символов. Вы просто используете нужный оператор, например, = (присваивание), == (проверка на равенство) или + (конкатенация). В этой статье термин "строка" означает объект любого из строковых типов стандартной библиотеки C++ (string или wstring). "Традиционные" же строки типов char* и const char* будут называться C-строками.

Все строковые типы и функции определяются в заголовочном файле <string>:

```
#include <string>
```

Как обычно, все идентификаторы принадлежат пространству имён std.[1]

Строки предоставляют возможность чтения и записи хранящихся в них символов. Обращение к отдельному символу производится одним из двух способов: оператором индексирования [] или функцией at(). Обе операции возвращают символ в позиции с заданным индексом. Как обычно, первому символу соответствует индекс 0, а последнему – индекс length()-1. Тем не менее существуют и различия.

Оператор [] не проверяет правильность индекса, передаваемого в аргументе, а функция at() выполняет такую проверку. При вызове функции at() с недействительным индексом генерируется исключение out_of_range. При вызове оператора [] с недействительным индексом возможны непредсказуемые последствия – скажем, недопустимые обращения к памяти, которые становятся причиной неприятных побочных эффектов, или аварийное завершение

программы.

Для константной версии оператора [] позиция за последним символом считается действительной. В этом случае текущее количество символов является допустимым индексом. Оператор возвращает значение, сгенерированное конструктором по умолчанию для типа символов. Следовательно, для объектов типа string возвращается символ \0.

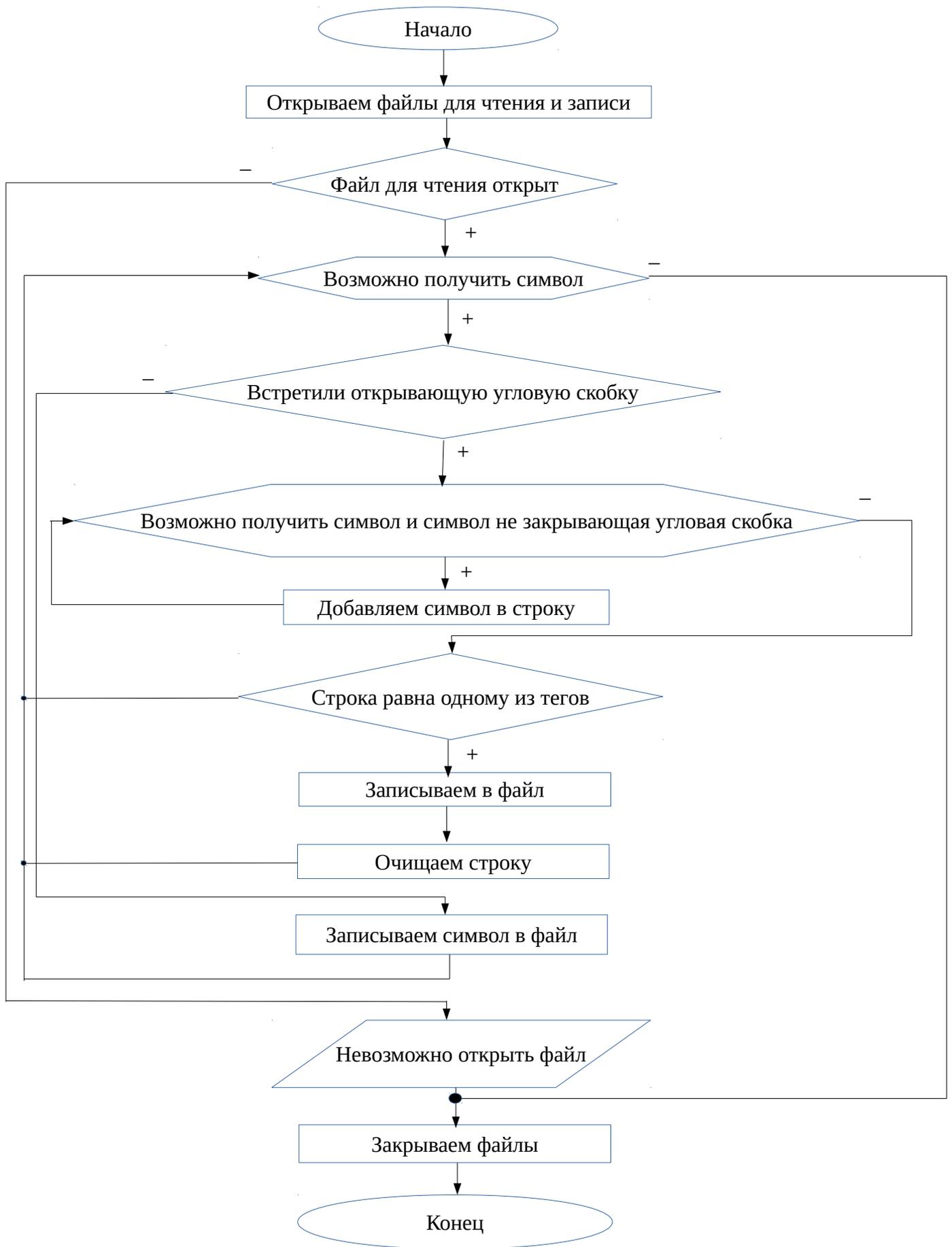
Во всех остальных случаях (для неконстантной версии оператора [] и функции at()) индекс, равный текущему количеству символов, считается недействительным. Его использование приводит к исключениям или непредсказуемым последствиям.

Стандарт не указывает, как должен быть реализован строковый класс, - он лишь определяет интерфейс строкового класса. В зависимости от выбранного концептуального подхода и приоритетов реализации могут существенно различаться по скорости работы и расходу памяти.

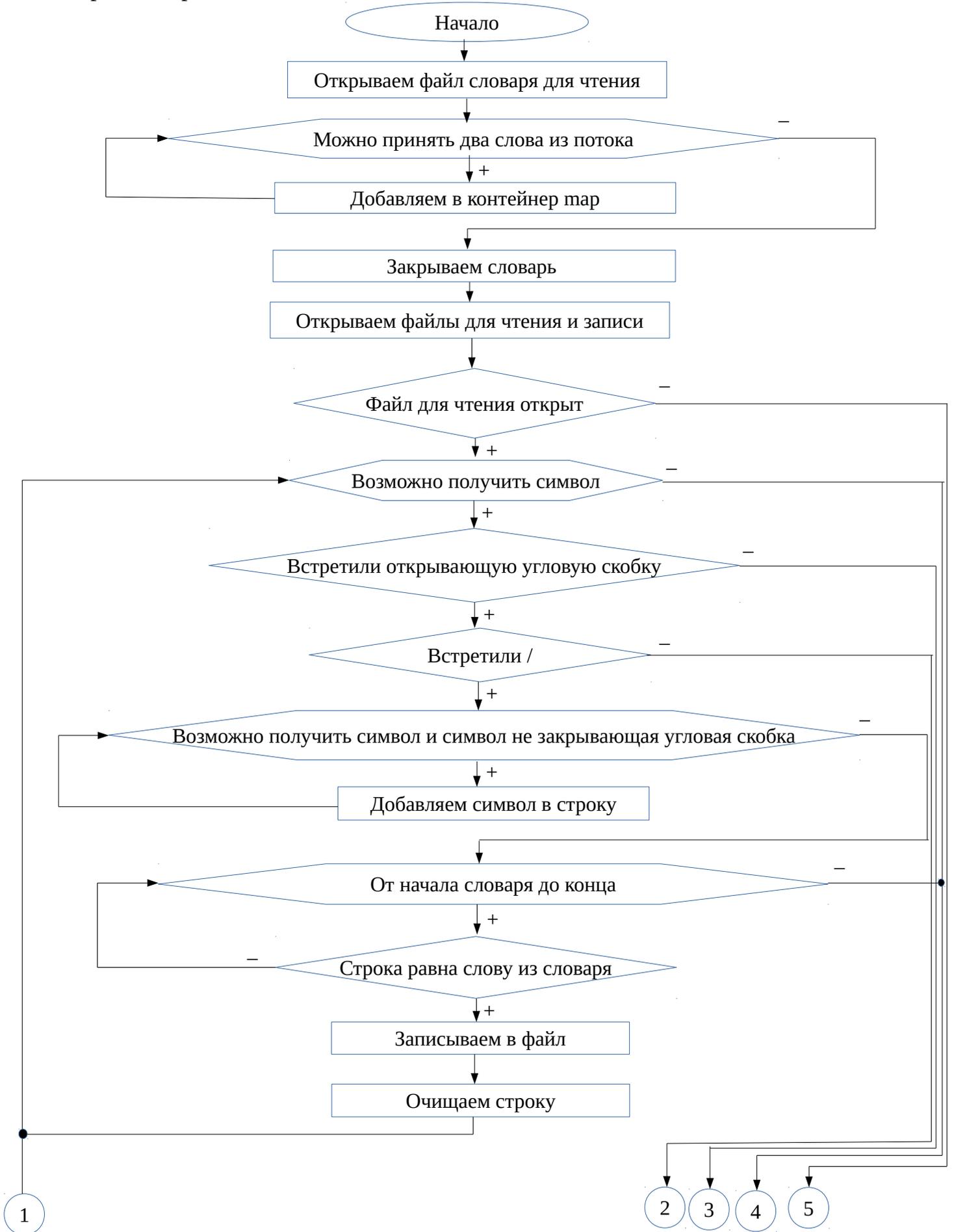
В данной статье рассмотрены несколько алгоритмов определения равенства строк, реализованных на языке программирования C++. Каждый из алгоритмов работает по принципу простейшего переводчика xml/html страниц. Текст между угловыми скобками сравнивается с каждым тегом и, в случае равенства, заменяется на русский эквивалент. Отличие состоит в том, что в первом алгоритме каждый из тегов прописывается вручную в тексте программы, а в последующих создан специальный файл-словарь, в котором содержатся теги и их перевод. Ещё одно отличие состоит в том, что в первом и втором алгоритме производится сравнение слов оператором ==, в третьем же сравнивается длина слова, первый и последний символ текста и слова из словаря.

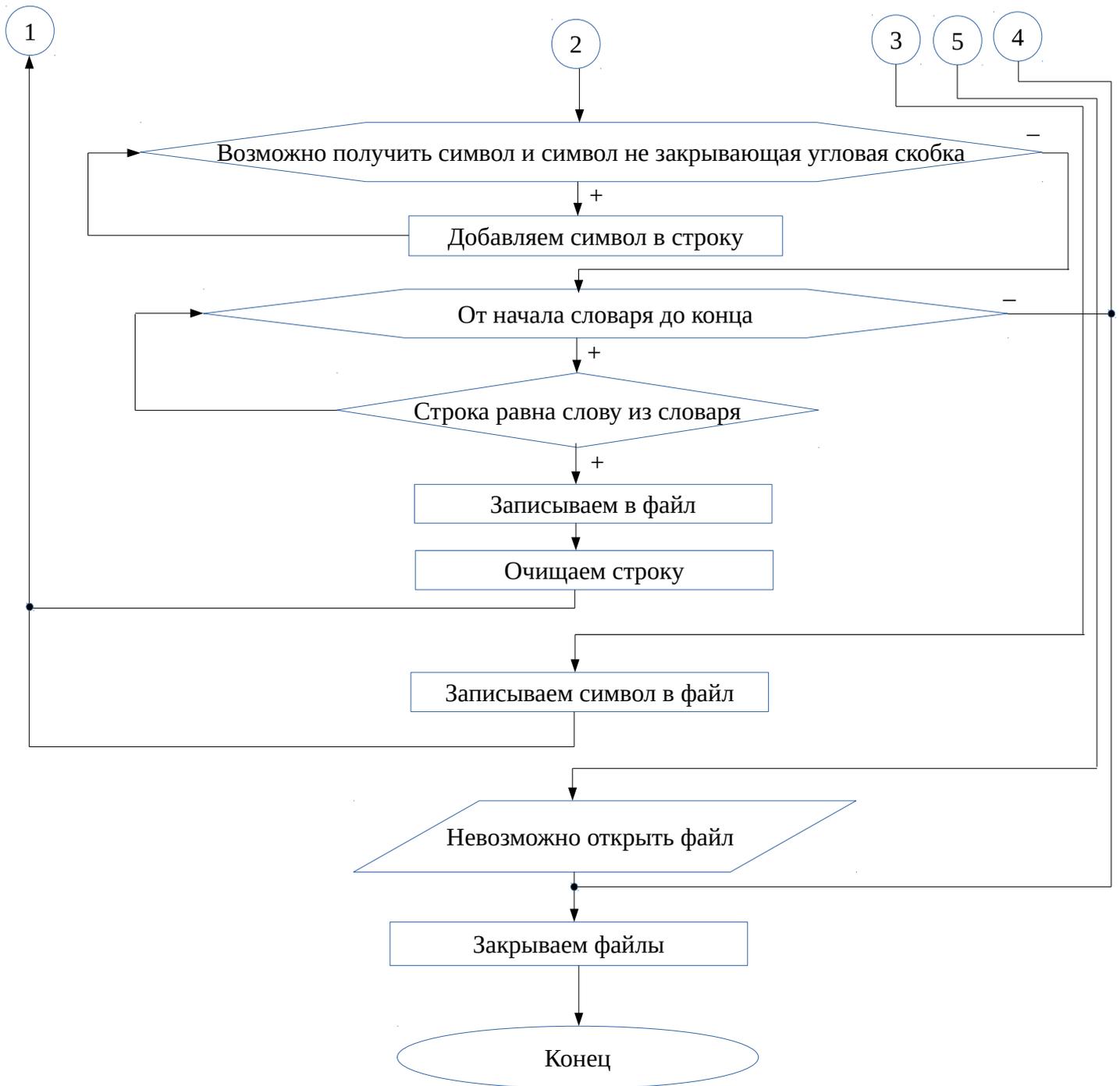
Далее рассмотрим блок-схемы алгоритмов:

Алгоритм первый:

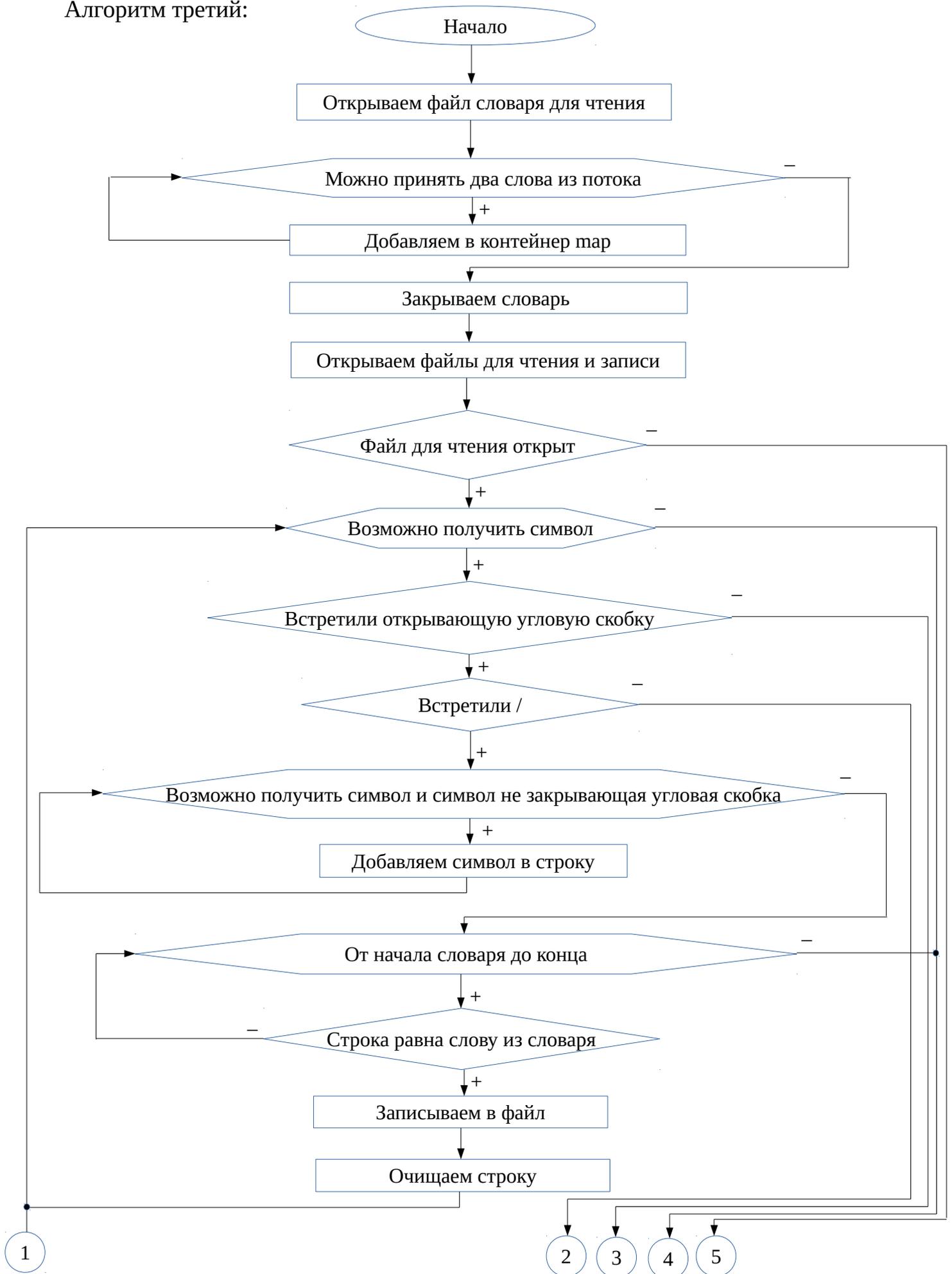


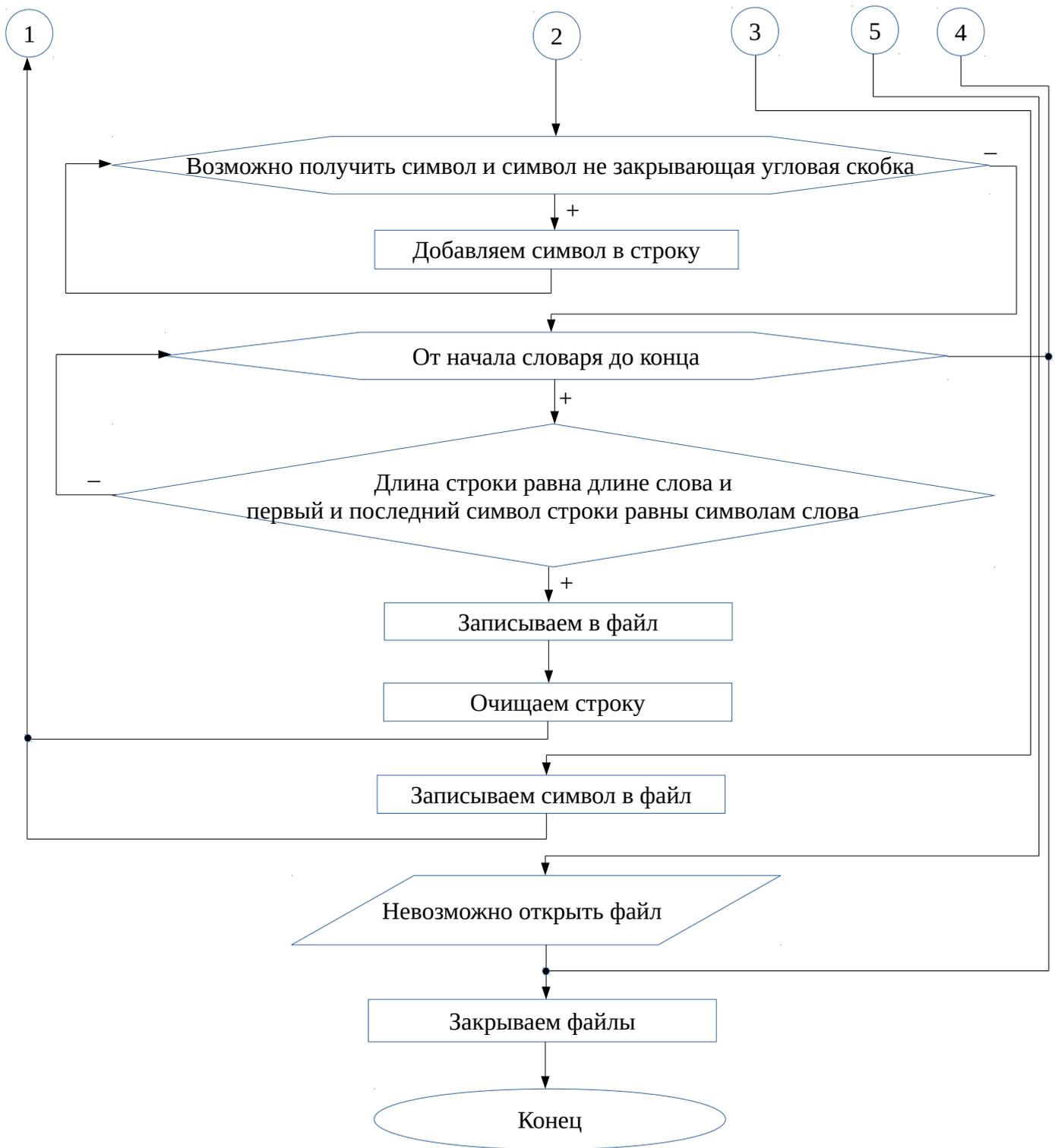
Алгоритм второй:





Алгоритм третий:





Экспериментально было получено, что время работы первого алгоритма составляет 5 секунд, второго и третьего – 6 секунд. Результаты по каждому алгоритму снимались на основе показаний функции `time()` заголовочного файла `<ctime>`. В теле функции `main()` вызывалась функция `time()` первый раз, с целью получения точки отсчёта, далее, после 100000000 итераций цикла с алгоритмом, функция вызывалась вновь. Разность показаний и представляла собой время работы.

На основе времени работы и удобства использования был выбран второй алгоритм, как оптимальный. Так как при добавлении новых слов достаточно изменить файл-словарь, не меняя содержимое компилируемого и исполняемого файла. Разницей во времени работы с первым алгоритмом можно пренебречь вследствие её незначительности.

Список литературы

1. Джосьютис, Н. С++ стандартная библиотека [Текст] : Пер. с англ. - М.: "Питер", 2004. – 729 с.
2. Страуструп, Б. Язык программирования С++ [Текст] : Пер. с англ. - М.: "Бином", 2008. – 369 с.
3. Мюссер, Д. С++ и STL справочное руководство [Текст] : Пер. с англ. - М.: "Вильямс", 2010. – 430 с.
4. С++. Лекция 4. Работа со строками. [Электронный ресурс]. - Режим доступа: <http://www.itmathrepetitor.ru/s-lekciya-4-rabota-so-strokami/>