# **КОНТРОЛЬНЫЕ ВОПРОСЫ**

## Как организована иерархия классов потоков в библиотеке iostream?

Базовым классом является ios\_base. Он отвечает за состояния и флаги. От него наследуется класс basic\_ios, работающий с буфером (basic\_streambuf); от basic\_ios наследуются basic\_istream и basic\_ostream; от них двоих — basic\_iostream (отв. за ввод/вывод на «пользовательском» уровне). От последних трёх упомянутых классов наследуются соответствующие stringstream‑ы (используются, например, при форматировании) и fstream-ы (работа с файлами). Все указанные классы (кроме ios\_base) имеют реализации для работы с char и wchar.

Буферы basic\_streambuf и наследованный от него basic\_stringbuf не входят в данную иерархию.

## Какие состояния потоков реализованы в базовом классе ios\_base?

В базовом классе реализованы следующие состояния потоков (ios\_state):

* goodbit — ошибок нет;
* badbit — неустранимая ошибка;
* failbit — ошибка операции ввода/ вывода, например, ошибка чтения
* eofbit — достигнут eof (конец файла).

Состояния потоков выставляются с помощью побитовых операций (на «пользовательском» уровне — метод setstate), могут быть получены из соответствующих методов (например, good(), eof() и т.п., тж. rdstate).

Заметим, что failbit является необходимым условием для eofbit, т.е. при нахождении потока во втором состоянии он находится и в первом.

## В чём разница между манипуляторами и флагами форматирования?

Манипуляторы и флаги форматирования влияют на работу потока.

Манипуляторы изменяют работу потока в общем случае не более, чем в рамках одной строки, в то время как флаги форматирования изменяют работу данного потока в целом: они содержатся в его состоянии.

## Из каких основных элементов состоят пути в файловой системе?

Пути в файловой системе (верно для ОС Windows) состоят из следующих элементов:

1. Корня (диска)
2. Корневой директории (обратный слэш)
3. Вложенных директорий
4. Имени файла

Последнее может быть разбито на stem и расширение.

## Зачем нужны форматы обмена данными, такие как JSON и XML?

Наличие общепринятых форматов позволяет стандартизировать обмен данными, что упрощает взаимодействие между различными объектами, например, веб-серверами и браузерами, базами данных.

Такие форматы также могут быть полезны при взаимодействии разработчиков, пишущих на разных языках, например, Java и Scala. Так, существуют кодогенераторы, способные «распаковать» JSON-формат в корректный проект на данном языке.