

1) Как организована иерархия классов потоков в библиотеке IOStream?

`ios_base` – базовый класс с флагами и локалями → `basic_ios` со специализациями для символьных типов `char` и `wchar_t`. Его наследует буферный класс `basic_streambuf` (со своими итераторами в наследниках), а также виртуально наследуют `basic_(i)(o)stream`. Их наследуют `w(i)(o)stream` и `(i)(o)stream` со специализациями `cin`, `cout`, `cerr`, `clog`, а также комбинированный класс `basic_iostream`. `basic_(io)(i)(o)stream` наследуют классы `(-)(i)(o)fstream` для файлового ввода-вывода и `(-)(i)(o)stringstream` для форматированных строк.

2) Какие состояния потоков реализованы в базовом классе `basic_ios`?

- 1) `goodbit` – все хорошо
- 2) `eofbit` – достигли конца файла (влечет подъем флага 3)
- 3) `failbit` – произошла не критическая ошибка
- 4) `badbit` – произошла критическая ошибка

3) В чем разница между манипуляторами и флагами форматирования?

Манипуляторы производят редактирование в моменте, тогда как флаги отражают состояние и оказывают глобальное влияние на потоки.

4) Из каких основных элементов состоят пути в файловой системе?

- 1) `root path` – корневая директория, например, `C:\` или `D:\` в системе Windows
- 2) `parent path` – директория, в которой хранится файл, например, `C:\boost_1_75_0\`
- 3) `filename` – имя файла, состоящее из `stem` – его названия и `extension` – его расширения, например, в `C:\boost_1_75_0\b2.exe` `b2.exe` – `filename`, `b2` – `stem`, `.exe` – `extension`

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

Форматы обмена данными нужны для обеспечения взаимодействия между модулями программы, написанными на разных языках программирования, а также для хранения в текстовом виде информации, которую могут использовать программы и читать/редактировать человек.

