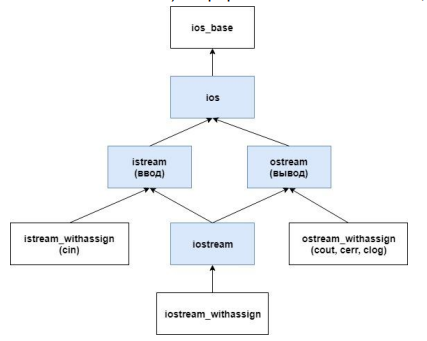
**43. Потоки ввода-вывода. Основные методы ввода-вывода. Обработка ошибок ввода-вывода.**

Функционал потоков ввода/вывода не определён как часть языка C++, а предоставляется Стандартной библиотекой C++ (и, следовательно, находится в пространстве имён std). При подключении заголовочного файла iostream, мы получаем доступ ко всей иерархии классов библиотеки iostream, отвечающих за функционал ввода/вывода данных (включая класс, который называется iostream). Иерархия этих классов выглядит примерно следующим образом:



Абстрактно, поток — это последовательность символов, к которой можно получить доступ. Поток ввода (или ещё «входной поток») используется для хранения данных, полученных от источника данных. Поток вывода (или ещё «выходной поток») используется для хранения данных, предоставляемых конкретному потребителю данных.

Класс ios определяет кучу разных вещей, которые являются общими для потоков ввода/вывода. Класс istream используется для работы с входными потоками. Оператор извлечения >> используется для извлечения значений из потока. Это имеет смысл: когда пользователь нажимает на клавишу клавиатуры, код этой клавиши помещается во входной поток. Затем программа извлекает это значение из потока и использует его.

Класс ostream используется для работы с выходными потоками. Оператор вставки << используется для помещения значений в поток. Это также имеет смысл: вы вставляете свои значения в поток, а затем потребитель данных использует их.

Класс iostream может обрабатывать как ввод, так и вывод данных, что позволяет ему осуществлять двунаправленный ввод/вывод.

Класс ios\_base содержит следующие флаги для обозначения состояния потоков:

goodbit — всё хорошо;

badbit — произошла какая-то фатальная ошибка (например, программа попыталась прочитать данные после конца файла);

eofbit — поток достиг конца файла;

failbit — произошла какая-то не фатальная ошибка (например, пользователь ввёл буквы, когда программа ожидала числа).

ios также предоставляет ряд методов для доступа к состояниям потока выше:

good() — возвращает true, если установлен goodbit (значит, что с потоком всё ок);

bad() — возвращает true, если установлен badbit (значит, что произошла какая-то фатальная ошибка);

eof() — возвращает true, если установлен eofbit (значит, что поток находится в конце файла);

fail() — возвращает true, если установлен failbit (значит, что произошла какая-то не фатальная ошибка);

clear() — сбрасывает все текущие флаги состояния потока и задаёт ему goodbit;

clear(state) — сбрасывает все текущие флаги состояния потока и устанавливает флаг, переданный в качестве параметра;

rdstate() — возвращает текущие установленные флаги;

setstate(state) — устанавливает флаг состояния, переданный в качестве параметра

https://habr.com/ru/post/326578/

**19. Область видимости переменных**

Область видимости — это часть кода. Каждая переменная объявляется в своей области видимости и к памяти, которую она занимает, можно обращаться начиная с ее объявления и заканчивая концом области видимости.

Главное предназначение ОВ — сохранить локальность переменных. В больших программах содержится очень много переменных, поэтому имеет смысл возможность давать им одинаковые имена, получая доступ к каждому из своей ОВ.

При выходе из блока переменные, объявленные в нем, уничтожаются в порядке, обратном объявлению.

Вот какие есть ОВ:

1. глобальные (не входят ни в одну другую ОВ);

2. пространства имен (namespace) (именованная ОВ);

3. класса (в классе);

4. локальная (в {} либо аргументы функции);

5. оператора (если не ставите скобки после for и ему подобных, его ОВ — одна строка);

https://codelessons.ru/cplusplus/oblast-vidimosti-peremennyx-v-c-lokalnye-i-globalnye-peremennye.html