Первые шаги в C++

Шпаргалка: команды компиляции, типичные ошибки, анатомия Hello World и ввод-вывод

Часть 1: Команды g++

КОМАНДА	ЧТО ДЕЛАЕТ	ПРИМЕР
`g++ файл.cpp -o имя`	Компилирует исходник в исполняемый файл	`g++ hello.cpp -o hello`
`g++ -Wall`	Включает все предупреждения (Warnings all)	`g++ -Wall hello.cpp -o hello`
`g++ -std=c++17`	Указывает стандарт языка C++17	`g++ -std=c++17 hello.cpp -o hello`
`g++ --version`	Показывает версию установленного компилятора	`g++ --version`
`./hello`	Запуск программы (Linux / macOS)	`./hello`
`hello.exe`	Запуск программы (Windows)	`hello.exe`
`g++ файл.cpp`	Компиляция без указания имени: создаст `a.out` или `a.exe`	`g++ hello.cpp`

Совет: всегда используй флаг `-o` с понятным именем файла. Так ты точно будешь знать, какой файл запускать, и не перепутаешь `a.out` с другим проектом.

Часть 2: Типичные ошибки и как их исправить

ОШИБКА	СООБЩЕНИЕ КОМПИЛЯТОРА	КАК ИСПРАВИТЬ
Пропущена точка с запятой `;`	`error: expected ';' before 'return'`	Добавить `;` в конец строки
Забыт префикс `std::`	`error: 'cout' was not declared in this scope`	Добавить `std::` перед `cout`, `cin`, `endl`
Нет функции `main`	`undefined reference to 'main'`	Добавить `int main() { ... }`
Опечатка в имени файла при компиляции	`error: ... No such file or directory`	Проверить имя файла. Использовать Tab для автодополнения
Не закрыта кавычка `"`	`error: missing terminating " character`	Добавить закрывающую кавычку `"`
Не закрыта фигурная скобка `}`	`error: expected '}' at end of input`	Добавить `}` в конец файла

Помни: ошибки компилятора — это подсказки, а не наказание. В сообщении всегда указан файл, строка и описание проблемы. Читай их внимательно — компилятор буквально говорит, что нужно исправить.

Часть 3: Анатомия Hello World

```cpp
// Подключает библиотеку ввода-вывода
#include <iostream>

// Точка входа: отсюда начинается программа
int main() {
    // Вывод в консоль
    std::cout
        << "Hello, World!"    // Текст для вывода (строковый литерал)
        << std::endl;         // Перевод строки (end line)

    // Возврат 0 = программа завершилась успешно
    return 0;
}
```

СТРОКА	ЧТО ОЗНАЧАЕТ
`#include <iostream>`	Подключает библиотеку для ввода и вывода текста
`int main()`	Точка входа в программу. Каждая программа на C++ обязана иметь `main`
`{` и `}`	Фигурные скобки обрамляют тело функции — всё, что между ними, будет выполнено
`std::cout << "текст";`	Выводит текст в консоль. `<<` — оператор передачи данных
`std::endl`	Переводит курсор на новую строку (end line)
`return 0;`	Завершает программу. 0 = успех, любое другое число = ошибка
`;`	Точка с запятой завершает каждый оператор. Обязательна!

Часть 4: Ввод и вывод

ОПЕРАЦИЯ	КОД	ЧТО ПРОИСХОДИТ
Вывод текста	`std::cout << "Привет";`	Выводит «Привет» в консоль
Вывод с переменной	`std::cout << "Имя: " << name;`	Склеивает текст и значение переменной в одной строке
Ввод одного слова	`std::cin >> переменная;`	Читает ввод с клавиатуры до первого пробела
Ввод целой строки	`std::getline(std::cin, str);`	Читает всю строку до нажатия Enter (нужен `#include <string>`)
Объявление строки	`std::string name;`	Создаёт переменную для хранения текста
Объявление числа	`int age;`	Создаёт переменную для хранения целого числа

Важно: `std::cin >>` читает только одно слово (до пробела). Если пользователь введёт «Иван Петров», в переменную попадёт только «Иван». Для чтения строки с пробелами используй `std::getline(std::cin, переменная);`.

Шпаргалка из статьи «Как начать писать на C++: установка среды, компиляция и первые программы»
Сохрани и держи рядом, пока учишься. Удачи в твоих первых проектах!