



Урок 22. C++: Строка и число

Часть 1. Материалы ученика

Программирование на Python и C++ · 8 класс

Урок 22 из 33 · 45 минут · Модуль «Символьные строки»

Уровень: начинающий программист; C++17.

Программа: <https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

В строках "-007" и "7x" начало похоже на число. Можно ли принять только часть записи и забыть оставшиеся символы?

Результат урока: Проверить всю строку до преобразования и отделить корректный ноль от ошибки ввода.

01. Проблемная ситуация

0–5 минут. Разделите "-007", "0", "+7", "7x", "-", "1000" на допустимые и недопустимые по правилу: необязательный минус, затем от одной до трёх ASCII-цифр. Объясните каждый отказ.

02. Понятия и разобранный пример

5–14 минут: 4 минуты на понятия, 3 на пример, 2 на объяснение способа партнёру.

Проверка формата предшествует вычислению

Контракт `parse_int`: необязательный первый знак -, затем 1–3 ASCII-цифры. Пробелы и плюс запрещены; диапазон результата от -999 до 999. Ведущие нули разрешены. Проверяем каждый символ, чтобы "7x" не превратилось молча в 7.

`std::optional<int>` в C++17 хранит либо целое, либо отсутствие значения. `std::nullopt` означает ошибку. Проверка `if(v)` проверяет наличие значения: `optional` с числом 0 считается заполненным. Запись `*v` получает число только после проверки наличия.

Число собирается слева направо: `value = value*10 + digit`. Для 007 промежуточные значения 0,0,7. Ограничение в три цифры проверено заранее, поэтому вычисления не переполняют `int`. Преобразование теряет ведущие нули и запись -0; исходный текст храните отдельно, если его вид важен.

Отступы набирайте обычными пробелами: 4 на уровень. При копировании с веб-страницы замените неразрывные пробелы обычными; полный код с обычными пробелами есть в PDF.

Для "-007" `start=1`, цифр 3; накопленное значение 7, после знака ответ -7. "-" отклоняется из-за отсутствия цифр, "7x" — из-за x. Ноль корректен, поэтому ошибка не обозначается числом 0.

```
#include <iostream>
#include <string>
#include <optional>
std::optional<int> parse_int(const std::string& s) {
    std::size_t start = (!s.empty() && s[0]!='-') ? 1 : 0;
    if (s.size() <= start || s.size() - start > 3) {
        return std::nullopt;
    }
    int value = 0;
    for (std::size_t i = start; i < s.size(); ++i) {
        if (s[i] < '0' || s[i] > '9') { return std::nullopt; }
        value = value * 10 + (s[i] - '0');
    }
    return start == 1 ? -value : value;
}
int main() {
    std::string s; std::getline(std::cin, s);
    auto v = parse_int(s);
    if (v) { std::cout << *v << "\n"; }
    else { std::cout << "error\n"; }
}
```



-007	знак -, цифры 007
Накопление	$0 \rightarrow 0 \rightarrow 7$
Результат	-7; исходная запись потеряна
7x	отказ при проверке x

Разделение проверки формата и накопления целого; ошибочный символ не игнорируется.

03. Практика 1. Проверяем материал

Знак, длина и лишний символ

14–25 минут. Проверьте 11 строк. Для отказа запишите этап: число цифр или недопустимый символ. Кавычки не вводите; пробел перед 7 сохраните.

Ввод	Ответ
"-007"; "0"; "-0"	-7; 0; 0
"999"; "-999"	999; -999
"1000"; "-"; ""	error; error; error
"7x"; "+7"; " 7"	error; error; error

2 минуты — прогноз, 4 — запуск, 3 — удалите проверку символа и объясните, почему вычисление для "7x" теряет смысл, 2 — восстановите исходник. Критерии: все ответы, причины, границы, успешный ноль.

04. Практика 2. Перенос способа

Две части времени по строгому формату

25–33 минуты. Строка времени имеет ровно вид HH:MM: две цифры часа 00–23 и две цифры минут 00–59. Верните минуты от полуночи или ошибку. За 2 минуты разберите "09:05", "23:59", "24:00", "9:05", "12:60"; за 4 реализуйте, за 2 проверьте ещё "00:00" и "ab:12". Критерии: длина, двоеточие, все цифры, оба диапазона.

05. Обсуждение и границы вывода

33–40 минут. За 2 минуты сравните числа и школьные коды кабинетов "007". За 3 аргументируйте, когда нужна исходная строка. За 2 объясните, почему проверка только длины не защищает от "ab:12" и почему 00:00 нельзя считать ошибкой.

06. Индивидуальная проверка

40–45 минут. Найдите `parse_int("-090")`, `minutes("01:30")`, объясните различие результатов и назовите строку длины 5, которая не является допустимым временем. Четыре пункта — четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Три случая первой строки допустимы. Для "1000" четыре цифры, для "-" и пустого ввода — ни одной. Остальные отказы связаны с символами. Без проверки `x-'0'` выражение вычисляет кодовую разность, а не десятичную цифру; результат нельзя считать разбором числа.

Практика 2: разбор и критерии

Ответы: 545,1439,error,error,error; дополнительно 0,error. Для 09:05: $9*60+5=545$. Полный самостоятельный исходник:

```
#include <iostream>
#include <string>
#include <optional>
std::optional<int> minutes(const std::string& s) {
    if (s.size()!=5 || s[2]!=':') { return std::nullopt; }
    for (int i : {0,1,3,4}) {
        if (s[i]<'0' || s[i]>'9') { return std::nullopt; }
    }
    int h = (s[0]-'0')*10 + s[1]-'0';
    int m = (s[3]-'0')*10 + s[4]-'0';
    if (h>23 || m>59) { return std::nullopt; }
    return 60*h + m;
}
int main() {
    std::string s; std::getline(std::cin, s);
    auto v = minutes(s);
    if (v) { std::cout << *v << "\n"; }
    else { std::cout << "error\n"; }
}
```

Обсуждение: возможный вывод

Количество можно хранить числом, а код с ведущими нулями — строкой. Длина проверяет структуру, но не допустимость каждого символа. `optional` разделяет отсутствие результата и результат 0.

Индивидуальная проверка: ответ и критерии

-90; 90. Первое значение — знаковое целое, второе — число минут от полуночи. Пример отказа: "25:00" (час вне диапазона), либо "01:99". По одному баллу за пункт.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, редактор, папку с правом записи, распечатку ученической части PDF, бумагу и ручку. Компилятор G++ с поддержкой C++17 и редактор. Сборка: `g++ -std=c++17 -Wall -Wextra -pedantic main.cpp -o lesson`. Запуск `./lesson` в Linux/macOS или `.\lesson.exe` в Windows (при сборке задайте `-o lesson.exe`). Установку и проверку запуска учитель выполняет заранее. Все программы, данные и ожидаемые ответы даны; регистрация детей и интернет на занятии не нужны.

Уровень — начинающий программист, 8 класс. Предпосылки: цикл по строке, ASCII-цифры, целые числа и функции. В паре автор прогнозирует и вводит код, проверяющий ведёт журнал, затем роли меняются. Без компьютера выполните трассировку полного кода на бумаге и сопоставьте с готовыми результатами; учитель демонстрирует подготовленный запуск. Бумажный вариант не считается фактическим запуском ученика. Перед практикой учитель выдаёт готовый файл разобранного примера, чтобы дети тратили время на изменение и проверку, а не длинное перепечатывание. Матрица программы предназначена учителю.



Минуты	Работа	Результат
0–5	Проблема	Вопрос и первое решение
5–14	Понятия и пример	Способ проверки
14–25	Практика 1	Готовый продукт и проверка
25–33	Практика 2	Перенос способа
33–40	Обсуждение	Аргумент и ограничение
40–45	Индивидуальная проверка	Самостоятельный ответ

07. Соответствие исходной программе

Пункт программы	Действие ученика	Проверяемый результат
Символьные строки	Проверить всю строку до преобразования и отделить корректный ноль от ошибки ввода.	Исходник, трассировка и таблица проверок
Обращение к символам; поиск, замена и преобразования	Две части времени по строгому формату	Самостоятельный алгоритм, ключ и индивидуальная проверка

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

C++: optional — значение или его отсутствие

<https://eel.is/c++draft/optional>

Тексты, учебные данные, реплики, программы и схемы подготовлены для уроков портала «Школы России». Числовые серии — учебные модели, не результаты проведённых классом опытов. Схемы авторские, сторонние фотографии не используются. Методические ссылки приведены для учителя; ученик получает все материалы здесь. Проверка первичных источников: 18.10.2026.