



Урок 27. C++: Ввод, заполнение и перебор

Часть 1. Материалы ученика

Программирование на Python и C++ · 8 класс

Урок 27 из 33 · 45 минут · Модуль «Массивы и списки»

Уровень: начинающий программист; C++17.

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

Программа прочитала три числа, хотя в начале строки был указан размер четыре. Как отличить полный ввод от обрывка и сохранить повторяемый набор для проверки?

Результат урока: Прочитать ровно заданное число элементов и заполнить vector воспроизводимыми псевдослучайными учебными значениями.

01. Проблемная ситуация

0–5 минут. Сопоставьте "3 4 -2 0", "3 4 -2", "2 4 -2 0". Объясните, почему последние две строки нельзя молча считать корректными списками.

02. Понятия и разобранный пример

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

Проверяем каждое чтение и лишний остаток

Вход — одна строка: n от 0 до 6, затем ровно n целых от -20 до 20. getline читает строку целиком, istreamstringstream разбирает её. Каждое чтение проверяется; ошибка не заменяется случайным или старым значением переменной.

После n элементов пытаемся прочитать ещё один строковый токен. Если он есть, ввод лишний. Это обнаруживает и лишнее число, и хвост вроде x после последнего числа. Проход for(int x:a) выводит значения в порядке заполнения. Пустой корректный вход записывается как "0".

std::mt19937(seed) создаёт псевдослучайный генератор. В учебном примере остаток rng()%10 даёт цифру 0..9 и повторяемую последовательность при том же seed. Такое отображение не идеально равномерно и не предназначено для статистического моделирования или защиты данных. Для обычной равномерной генерации используют uniform_int_distribution; для урока важна полностью воспроизводимая запись.

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

"3 4 -2 0": n=3; последовательно добавляем 4,-2,0; остаточных токенов нет; вывод "3: 4 -2 0". "2 4 -2 0" отклоняется, потому что после двух элементов остался токен 0.

```
#include <iostream>
#include <sstream>
#include <string>
#include <vector>
int main() {
    std::string line; std::getline(std::cin, line);
    std::istringstream in(line);
    int n;
    if (!(in >> n) || n < 0 || n > 6) {
        std::cout << "error\n"; return 0;
    }
    std::vector<int> a;
    for (int i = 0; i < n; ++i) {
        int x;
        if (!(in >> x) || x < -20 || x > 20) {
            std::cout << "error\n"; return 0;
        }
        a.push_back(x);
    }
    std::string extra;
    if (in >> extra) { std::cout << "error\n"; return 0; }
    std::cout << a.size() << " ";
    for (int x : a) { std::cout << " " << x; }
    std::cout << "\n";
}
```



Ввод	3 4 -2 0
После шага 1	[4]
После шага 2	[4, -2]
После шага 3	[4, -2, 0]; лишних токенов нет

Последовательное заполнение vector через push_back; размер увеличивается после каждого успешного чтения.

03. Практика 1. Проверяем материал

Размер, диапазон и конец строки

14–25 минут. Для восьми строк запишите прогноз и причину отказа. Сохраните точную строку, включая хвост x, а не только предполагаемые числа.

Строка ввода	Ответ
"3 4 -2 0"; "0"	3: 4 -2 0 и 0:
"3 4 -2"; "2 4 -2 0"	error; error
"1 21"; "1 7x"	error; error
"1 x"; "7"	error; error

2 минуты — прогноз, 4 — сборка и запуск, 3 — уберите проверку extra и найдите ошибочно принятый ввод, 2 — восстановите код. Критерии: неполный ввод, лишний ввод, диапазон и хвост последнего токена.

04. Практика 2. Перенос способа

Повторяемый генератор с фиксированным seed

25–33 минуты. Создайте generated(n,seed): для уже проверенного n от 0 до 6 вернуть n значений rng()%10 от нового mt19937(seed). За 2 минуты предскажите равенство двух свежих запусков с seed=17, за 4 реализуйте, за 2 сравните n=6 для seed=17 и 1, затем n=0. Критерии: длина, диапазон, повторение, сохранённый полный список.

05. Обсуждение и границы вывода

33–40 минут. За 2 минуты сравните тест по seed и тест с явно записанным вектором. За 3 объясните отличие нового генератора от продолжающего работу старого. За 2 решите, какие тесты следует добавлять вручную.

06. Индивидуальная проверка

40–45 минут. Для "2 -20 20" укажите вывод, для "2 -20 20 0" — результат и причину, затем объясните, что записать для повторения случайного теста. Четыре пункта — четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Без проверки extra строка "2 4 -2 0" будет ошибочно принята. "1 7x" тоже пройдет чтение числа 7, но хвост x должен вызвать отказ. "0" — корректный пустой список. "7" нарушает допустимый размер, "1 21" — диапазон значения.

Практика 2: разбор и критерии

Для seed=17 список [9,5,1,2,0,9], для seed=1 — [5,9,4,8,3,3]. Два свежих объекта с seed=17 совпадают; n=0 даёт []. Не сравнивайте это с Python Random(17): совпадение названия алгоритма или seed само по себе не означает одинаковый интерфейс и последовательность результатов.

```
#include <iostream>
#include <random>
#include <vector>
std::vector<int> generated(int n, unsigned seed) {
    std::mt19937 rng(seed);
    std::vector<int> a;
    for (int i = 0; i < n; ++i) {
        a.push_back(static_cast<int>(rng() % 10));
    }
    return a;
}
int main() {
    for (unsigned seed : {17u, 17u, 1u}) {
        for (int x : generated(6, seed)) { std::cout << x << " "; }
        std::cout << "\n";
    }
    std::cout << "empty-size=" << generated(0, 17).size() << "\n";
}
```

Обсуждение: возможный вывод

Явный список проще передать между языками. Новый генератор начинает последовательность заново, старый продолжает её. Нужны отдельные тесты n=0, n=6, неверного размера, обрывка ввода и лишнего токена; случайная генерация их не гарантирует.

Индивидуальная проверка: ответ и критерии

"2: -20 20"; второй ввод даёт error; причина — лишний токен 0. Для повторения сохраняют полный список, код генерации и seed; сведения о языке и среде помогают воспроизвести программу.

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: vector, push_back, потоки ввода, for и if. 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
Массивы и списки	Прочитать ровно заданное число элементов и заполнить vector воспроизводимыми псевдослучайными учебными значениями.	Исходник, трассировка и таблица проверок
Создание, ввод и обработка массивов; перебор элементов	Повторяемый генератор с фиксированным seed	Самостоятельный алгоритм, ключ и индивидуальная проверка

Источники и права на изображения

Исходная программа курса

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

C++: генератор Mersenne Twister

<https://eel.is/c++draft/rand.eng.mers>

C++: строковые потоки

<https://eel.is/c++draft/string.streams>

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.