



Урок 28. C++: Сумма и количество по условию

Часть 1. Материалы ученика

Программирование на Python и C++ · 8 класс

Урок 28 из 33 · 45 минут · Модуль «Массивы и списки»

Уровень: начинающий программист; C++17.

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

По результатам датчика нужно считать только значения не ниже порога. Почему сумма всех значений и число уникальных значений отвечают на другие вопросы?

Результат урока: Найти сумму и число выбранных элементов vector и проверить пустой вход, повторы и отрицательные значения.

01. Проблемная ситуация

0–5 минут. Учебные показания [-4,0,6,6,9], порог 6. Обведите подходящие элементы, отдельно запишите сумму и количество. Сравните со строгим условием $x > 6$.

02. Понятия и разобранный пример

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

Сумма и количество обновляются вместе

Обрабатываем `vector<int>` длиной не более 100 со значениями -1000..1000. При $x \geq t$ обновляем `total += x` и `++count`. `total` имеет тип `long long`, `count` — `std::size_t`. Даже при иных более крупных учебных значениях тип суммы нужно выбирать с учётом её возможного размера.

Функция принимает `const vector<int>&`: читает исходный вектор без изменения и без его полного копирования. Структура `Stats` возвращает два именованных результата. Пустой вход естественно даёт `total=0, count=0`; особое обращение к `a[0]` не требуется.

Для отрицательного чётного x условие $x < 0 \ \&\& \ x \% 2 == 0$. Остаток ноль проверяет делимость и для отрицательных чисел. Не используйте $x \% 2 == 1$ как универсальную проверку нечётности отрицательных чисел в C++. Повторяющиеся элементы считаются отдельно.

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 [-4,0,6,6,9] при $t=6$ состояния выбранных: (6,1),(12,2),(21,3). При $t=7$ результат (9,1). Нулевая сумма может получиться и при непустом выбранном наборе, поэтому проверяем `count` отдельно.

```
#include <iostream>
#include <vector>
struct Stats { long long total; std::size_t count; };
Stats report(const std::vector<int>& a, int t) {
    Stats r{0,0};
    for (int x : a) {
        if (x >= t) { r.total += x; ++r.count; }
    }
    return r;
}
int main() {
    for (auto a : std::vector<std::vector<int>>{
        {-4,0,6,6,9},{},{1,2},{6,6},{0}}) {
        auto r = report(a,6);
        std::cout << r.total << " " << r.count << "\n";
    }
}
```



Значения	-4 0 6 6 9
$x \geq 6$	нет нет да да да
total	0 0 6 12 21
count	0 0 1 2 3

Пороговый отбор в C++: обе шестёрки входят в сумму и количество.

03. Практика 1. Проверяем материал

Повторы — отдельные наблюдения

14–25 минут. В main приведены первые пять входов. Для двух последних отдельно замените список и порог. Сначала вычислите семь пар вручную.

Вектор; порог	total; count
[-4,0,6,6,9]; 6	21; 3
[]; 6 и [1,2]; 6	0; 0
[6,6]; 6 и [0]; 6	12; 2 и 0; 0
[-3,-1]; -2 и [0]; 0	-1; 1 и 0; 1

2 минуты — прогноз, 4 — запуск, 3 — замените $\geq$ на $>$ и подберите минимальный контрпример, 2 — восстановите код. Критерии: сумма, число элементов, граница, пустой и отрицательный случай.

04. Практика 2. Перенос способа

Два условия для отбора отрицательных чётных

25–33 минуты. Новое правило: выбрать только отрицательные чётные значения. Найдите сумму и количество для [-4,-3,0,6,-6], [], [-1,-5], [-2,-2]. За 2 минуты составьте прогнозы, за 4 реализуйте, за 2 выполните четыре теста. Критерии: оба условия через $\&\&$, ноль исключён, повторы сохранены, пустой вход.

05. Обсуждение и границы вывода

33–40 минут. За 2 минуты объясните разницу между подсчётом записей и различных значений. За 3 сравните $\&\&$ и $\|\|$ на -3,0,6. За 2 сформулируйте, как проверить исходник после функции с const-ссылкой.

06. Индивидуальная проверка

40–45 минут. Для [-8,-2,0,5] найдите negative_even, затем report с $t=0$, назовите выбранные индексы первой функции и объясните роль нуля во второй. Четыре пункта — четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Минимальный контрпример [6], t=6: правильно (6,1), ошибочно (0,0). Для [6,6] ошибка теряет сразу две записи. Для [0], t=0 правильно (0,1), а не (0,0). Функция не изменяет вход; правильная реализация выше.

Практика 2: разбор и критерии

Результаты (-10,2),(0,0),(0,0),(-4,2). Ноль чётный, но не отрицательный; -3 отрицательное, но нечётное. Замена && на || ошибочно добавит оба вида неподходящих значений.

```
#include <iostream>
#include <vector>
struct Stats { long long total; std::size_t count; };
Stats negative_even(const std::vector<int>& a) {
    Stats r{0,0};
    for (int x : a) {
        if (x < 0 && x % 2 == 0) {
            r.total += x; ++r.count;
        }
    }
    return r;
}
int main() {
    for (auto a : std::vector<std::vector<int>>{
        {-4,-3,0,6,-6},{},{-1,-5},{-2,-2}}) {
        auto r = negative_even(a);
        std::cout << r.total << " " << r.count << "\n";
    }
}
```

Обсуждение: возможный вывод

Повторы — самостоятельные элементы. При || отрицательное нечётное -3 и неотрицательные чётные 0,6 проходят, что противоречит правилу. Вход можно вывести до и после вызова; const-ссылка также запрещает изменять элементы через этот параметр.

Индивидуальная проверка: ответ и критерии

negative_even: (-10,2); report: (5,2); индексы 0,1. Во второй функции ноль удовлетворяет $x \geq 0$, увеличивает count, но не меняет total.

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, цикл по значениям, if, 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
Массивы и списки	Найти сумму и число выбранных элементов <code>vector</code> и проверить пустой вход, повторы и отрицательные значения.	Исходник, трассировка и таблица проверок
Создание, ввод и обработка массивов; перебор элементов	Два условия для отбора отрицательных чётных	Самостоятельный алгоритм, ключ и индивидуальная проверка

Источники и права на изображения

Исходная программа курса

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

Рабочий текст стандарта C++: `vector`

<https://eel.is/c++draft/vector.overview>

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.