

Ravesli [Ravesli](#)


- [Уроки по C++](#)
- [OpenGL](#)
- [SFML](#)
- [Qt5](#)
- [RegEx](#)
- [Ассемблер](#)
- [Купить .PDF](#)


Урок №76. Массивы и циклы

 [Юрий](#) |

- [Уроки C++](#)

|

 Обновл. 11 Авг 2020 |

 38456

[16](#)

На этом уроке мы рассмотрим использование массивов с циклами, а также подводные камни, с которыми вы можете при этом столкнуться.

Оглавление:

1. [Зачем использовать циклы с массивами?](#)
2. [Циклы и массивы](#)
3. [Использование циклов с массивами](#)
4. [Массивы и «ошибка неучтенной единицы»](#)
5. [Тест](#)

Зачем использовать циклы с массивами?

Рассмотрим случай, когда нужно вычислить средний балл всех студентов в группе. Используя отдельные переменные:

```
1 const int numStudents = 5;
2 int student0 = 73;
3 int student1 = 85;
4 int student2 = 84;
5 int student3 = 44;
6 int student4 = 78;
7
8 int totalScore = student0 + student1 + student2 + student3 + student4;
9 double averageScore = static_cast<double>(totalScore) / numStudents;
```

Мы получим много объявлений переменных и, следовательно, много кода — а это всего лишь 5 студентов! А представьте, если бы их было 30 или 150.

Кроме того, чтобы добавить нового студента, нам придется объявить новую переменную, инициализировать её и добавить в переменную `totalScore`. И это всё вручную. А каждый раз при изменении старого кода есть риск наделать новых ошибок. А вот с использованием массива:

```
1 const int numStudents = 5;
2 int students[numStudents] = { 73, 85, 84, 44, 78 };
3 int totalScore = students[0] + students[1] + students[2] + students[3] + students[4];
4 double averageScore = static_cast<double>(totalScore) / numStudents;
```

Количество объявленных переменных сократится, но в `totalScore` по-прежнему придется заносить каждый элемент массива вручную. И, как указано выше, изменение количества студентов означает, что формулу `totalScore` необходимо будет изменять также вручную.

Если бы был только способ автоматизировать этот процесс.

Циклы и массивы

Из предыдущего урока мы уже знаем, что индекс массива не обязательно должен быть константным значением — он может быть и обычной переменной. Это означает, что мы можем использовать счетчик цикла в качестве индекса массива для доступа к элементам и выполнения с ними необходимых математических и других операций. Это настолько распространенная практика, что почти всегда при обнаружении массива, вы найдете рядом с ним цикл! Когда цикл используется для доступа к каждому элементу массива поочередно, то это называется **итерацией по массиву**. Например:

```
1 int students[] = { 73, 85, 84, 44, 78 };
2 const int numStudents = sizeof(students) / sizeof(students[0]);
3 int totalScore = 0;
4
5 // Используем цикл для вычисления totalScore
6 for (int person = 0; person < numStudents; ++person)
7     totalScore += students[person];
8
9 double averageScore = static_cast<double>(totalScore) / numStudents;
```

Это решение идеально подходит как в плане удобства и чтения, так и поддержки. Поскольку доступ к каждому элементу массива выполняется через цикл, то формула подсчета суммы всех значений автоматически настраивается с учетом количества элементов в массиве. И для вычисления средней оценки нам уже не нужно будет вручную добавлять новых студентов и индексы новых элементов массива!

А вот пример использования цикла для поиска в массиве наибольшего значения (наилучшей оценки среди всех студентов в группе):

```
1 #include <iostream>
```

```
2 |
3 | int main()
4 | {
5 |     int students[] = { 73, 85, 84, 44, 78};
6 |     const int numStudents = sizeof(students) / sizeof(students[0]);
7 |
8 |     int maxScore = 0; // отслеживаем самую высокую оценку
9 |     for (int person = 0; person < numStudents; ++person)
10 |         if (students[person] > maxScore)
11 |             maxScore = students[person];
12 |
13 |     std::cout << "The best score was " << maxScore << '\n';
14 |
15 |     return 0;
16 | }
```

Здесь уже используется переменная `maxScore` (не из цикла) для отслеживания самого большого значения массива. Сначала инициализируем `maxScore` значением `0`, что означает, что мы еще не видели никаких оценок. Затем перебираем каждый элемент массива и, если находим оценку, которая выше предыдущей, присваиваем её значение переменной `maxScore`. Таким образом, `maxScore` всегда будет хранить наибольшее значение из всех элементов массива.

Использование циклов с массивами

Циклы с массивами обычно используются для выполнения одной из 3-х следующих задач:

- ➔ Вычислить значение (например, среднее или сумму всех значений).
- ➔ Найти значение (например, самое большое или самое маленькое).
- ➔ Отсортировать элементы массива (например, по возрастанию или по убыванию).

При вычислении значения, переменная обычно используется для хранения промежуточного результата, который необходим для вычисления конечного значения. В примере, приведенном выше, где мы вычисляем средний балл, переменная `totalScore` содержит сумму значений всех рассмотренных элементов.

При поиске значения, переменная обычно используется для хранения наилучшего варианта (или индекса наилучшего варианта) из всех просмотренных. В примере, приведенном выше, где мы используем цикл для поиска наивысшей оценки, переменная `maxScore` используется для хранения наибольшего количества баллов из просмотренных ранее элементов массива.

Сортировка массива происходит несколько сложнее, так как в этом деле используются вложенные циклы (но об этом уже на следующем уроке).

Массивы и «ошибка неучтенной единицы»

Одной из самых сложных задач при использовании циклов с массивами является убедиться, что цикл выполняется правильное количество раз. Ошибку на единицу (или «*ошибку неучтенной единицы*») сделать легко, а попытка получить доступ к элементу, индекс которого больше, чем длина массива, может иметь самые разные последствия. Рассмотрим следующую программу:

```
1 #include <iostream>
2
3 int main()
4 {
5     int students[] = { 73, 85, 84, 44, 78 };
6     const int numStudents = sizeof(students) / sizeof(students[0]);
7
8     int maxScore = 0; // отслеживаем самую высокую оценку
9     for (int person = 0; person <= numStudents; ++person)
10         if (students[person] > maxScore)
11             maxScore = students[person];
12
13     std::cout << "The best score was " << maxScore << '\n';
14
15     return 0;
16 }
```

Здесь проблема состоит в неверном условии оператора if в цикле for! Объявленный массив содержит 5 элементов, проиндексированных от 0 до 4. Однако цикл внутри перебирает элементы от 0 до 5. Следовательно, на последней итерации в цикле for выполнится:

```
1         if (students[5] > maxScore)
2             maxScore = students[5];
```

Но ведь `students[5]` не определен! Его значением, скорее всего, будет простой мусор. И в итоге результатом выполнения цикла может быть ошибочный `maxScore`.

Однако представьте, что бы произошло, если бы мы ненароком присвоили значение элементу `students[5]`! Мы бы могли перезаписать другую переменную (или её часть) или испортить что-либо — эти типы ошибок очень трудно отследить!

Следовательно, при использовании циклов с массивами, всегда перепроверяйте условия в циклах, чтобы убедиться, что их выполнение не приведет к ошибке неучтенной единицы.

Тест

Задание №1

Выведите на экран следующий массив с помощью цикла:

```
1 int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
```

Подсказка: Используйте трюк с `sizeof` (из предыдущего урока) для определения длины массива.

Ответ №1

```
1 #include <iostream>
2
3 int main()
4 {
5     int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
6     const int arrayLength = sizeof(array) / sizeof(array[0]);
7
8     for (int index=0; index < arrayLength; ++index)
9         std::cout << array[index] << " ";
10    return 0;
11 }
```

Задание №2

Используя массив из задания №1:

Попросите пользователя ввести число от 1 до 9. Если пользователь введет что-либо другое — попросите его снова ввести число и так до тех пор, пока он не введет корректное значение из заданного диапазона. Как только пользователь введет число от 1 до 9, выведите массив на экран. Затем найдите в массиве элемент с числом, которое ввел пользователь, и выведите его индекс.

Для обработки некорректного ввода используйте следующий код:

```
1 // Если пользователь ввел некорректное значение
2 if (std::cin.fail())
3 {
4     std::cin.clear();
5     std::cin.ignore(32767, '\n');
6 }
```

Ответ №2

```
1 #include <iostream>
2
3 int main()
4 {
5     // Сначала принимаем корректный пользовательский ввод
6     int number = 0;
7     do
8     {
9         std::cout << "Enter a number between 1 and 9: ";
10        std::cin >> number;
11
12        // Если пользователь ввел некорректное значение
13        if (std::cin.fail())
14            std::cin.clear();
15
16        std::cin.ignore(32767, '\n');
17    }
```

```
18 } while (number < 1 || number > 9);
19
20 // Далее выводим массив на экран
21 int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
22 const int arrayLength = sizeof(array) / sizeof(array[0]);
23
24 for (int index=0; index < arrayLength; ++index)
25     std::cout << array[index] << " ";
26
27 std::cout << "\n";
28
29 // Затем ищем в массиве число, которое ввел пользователь и выводим его индекс
30 for (int index=0; index < arrayLength; ++index)
31 {
32     if (array[index] == number)
33     {
34         std::cout << "The number " << number << " has index " << index << "\n";
35         break; // так как каждый элемент в массиве уникальный, то нет надобности п
36     }
37 }
38
39 return 0;
40 }
```

Задание №3

Измените следующую программу так, чтобы вместо `maxScore` с наибольшим значением, переменная `maxIndex` содержала индекс элемента с наибольшим значением:

```
1 #include <iostream>
2
3 int main()
4 {
5     int students[] = { 73, 85, 84, 44, 78};
6     const int numStudents = sizeof(students) / sizeof(students[0]);
7
8     int maxScore = 0; // отслеживаем самую высокую оценку
9
10    for (int person = 0; person < numStudents; ++person)
11        if (students[person] > maxScore)
12            maxScore = students[person];
13
14    std::cout << "The best score was " << maxScore << '\n';
15
16    return 0;
17 }
```

Ответ №3

```
1 #include <iostream>
2
3 int main()
4 {
5     int students[] = { 73, 85, 84, 44, 78};
6     const int numStudents = sizeof(students) / sizeof(students[0]);
7
8     int maxIndex = 0; // отслеживаем индекс самого большого значения
9
10    for (int person = 0; person < numStudents; ++person)
11        if (students[person] > students[maxIndex])
12            maxIndex = person;
13
14    std::cout << "The best score was " << students[maxIndex] << '\n';
15
16    return 0;
17 }
```

Оценить статью:



(199 оценок, среднее: 4,92 из 5)



[← Урок №75. Фиксированные массивы](#)



[Урок №77. Сортировка массивов методом выбора →](#)

Комментариев: 16



1. Юрий:
[28 ноября 2020 в 15:11](#)

Задание №3

```
1 #include <iostream>
2
3 int main()
4 {
5     setlocale(LC_ALL, "rus");
6     int students[] = { 73, 85, 84, 44, 78 };
7     const int numStudents = sizeof(students) / sizeof(students[0]);
```

```

8
9     int maxScore = 0; // отслеживаем самую высокую оценку
10    int maxIndex = 0; // содержала индекс элемента с наибольшим значением
11
12    for (int person = 0; person < numStudents; ++person)
13        if (students[person] > maxScore)
14        {
15            maxScore = students[person];
16            maxIndex = person+1; //+1 неучтенная единица
17        }
18    std::cout << "Наибольшее значение " << maxScore << " было в элементе № " << m
19
20    return 0;
21 }

```

[Ответить](#)



2. Yana:

[29 июня 2020 в 21:42](#)

Знаю, что без цикла, но имеет право жить это решение второго задания? Меня пугает предупреждение C6385. Позже постараюсь решить с циклом, когда пойму(

```

1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
7      int total = sizeof(array) / sizeof(array[0]);
8      int x;
9      while (true)
10     {
11         cout << "Select a number of array 1 to 9: ";
12         cin >> x;
13         if (std::cin.fail() || x>total || x<=0)
14         {
15             std::cin.clear();
16             std::cin.ignore(32767, '\n');
17             cout << "Please, try again!\n";
18         }
19         else if (x > 0 || x <= total)
20             break;
21     }
22     cout << "Your number of array "<<x<<" have : "<<array[x-1];
23 }

```

[Ответить](#)



3. Onium:

[28 июня 2020 в 23:36](#)

Задание 3

```
1 #include <iostream>
2
3 int main()
4 {
5     int students[] = { 73, 85, 84, 99, 78 };
6     const int numStudents = sizeof(students) / sizeof(students[0]);
7
8     int maxScore = 0; // отслеживаем самую высокую оценку
9     int maxIndex = 0;
10    for (int person = 0; person < numStudents; ++person)
11    {
12        if (students[person] > maxScore)
13            maxScore = students[person];
14        if (students[person] == maxScore)
15        {
16            maxIndex = person;
17        }
18    }
19
20    std::cout << "The best score was " << maxScore << '\n';
21    std::cout << "The best index was " << maxIndex << '\n';
22
23    return 0;
24 }
```

[Ответить](#)



4. Onium:

[28 июня 2020 в 23:19](#)

Второе задание

```
1 #include <iostream>
2
3 int getNumber()
4 {
5     int x;
6     do
7     {
8         std::cout << "Enter a number between 1 and 9: ";
9         std::cin >> x;
10    }
```

```
11         if (std::cin.fail())
12         {
13             std::cin.clear();
14             std::cin.ignore(32767, '\n');
15         }
16     } while (x < 1 || x > 9);
17     return x;
18 }
19
20 int main()
21 {
22     int y = getNumber();
23     int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
24     int numbers = sizeof(array) / sizeof(array[0]);
25
26     if(y > 1 || y < 9)
27     for (int index = 0; index < numbers; ++index)
28     {
29         std::cout << array[index] << " ";
30     }
31
32     for (int index = 0; index < numbers; ++index)
33     {
34         if (array[index] == y)
35         {
36             std::cout << "The " << y << " has index " << index << "\n";
37             break;
38         }
39     }
40     return 0;}
```

[Ответить](#)



5. Павел:

[12 мая 2020 в 17:36](#)

Вопрос по задаче 2. Почему если пишу "\n Не угадал! \n" при выполнении отступ будет 1 пример:

Введите число от 1 до 9: 45

Не угадал!

Введите число от 1 до 9:

А если пишу "\n Не угадал! \n\n" то всё получается как задумал(cout<<"\n Не угадал! "<<endl; не канает, т.к. получается результат пример 1):

Введите число от 1 до 9: 45

Не угадал!

Введите число от 1 до 9:

Сама прога:

```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6
7  int main()
8  {
9      setlocale(LC_ALL, "Russian");
10
11     int x;
12
13     do
14     {
15         cout << "Введите число от 1 до 9: ";
16         cin >> x;
17
18         if (std::cin.fail())
19         {
20             cout << "Не верное значение.\n";
21             cin.clear();
22             cin.ignore(32767, '\n');
23         }
24         else if (x < 1 || x > 9)
25             cout << "\n" << " Не угадал!" << "\n\n";
26
27     } while (x < 1 || x > 9);
28
29     int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
30     const int numstudent = sizeof(array) / sizeof(array[0]);
31
32     for (int person = 0; person < numstudent; ++person)
33         if (x == array[person])
34             cout << person << " ";
35
36     return 0;
37 }
```

[Ответить](#)



1. *Антон:*

[29 августа 2020 в 15:41](#)

Когда ты вводишь значение через `std::cin` автоматически добавляется абзац, соответственно на самом деле это выглядит как `\n\n` Не угадал `\n\n`

[Ответить](#)



6. Вячеслав:

[29 марта 2019 в 20:23](#)

третье задание:

```
1 #include "pch.h"
2 #include <iostream>
3
4 int main()
5 {
6     setlocale(0, "");
7     using namespace std;
8     //инициализируем массив 5-ю элементами
9     int array[] = { 73, 85, 84, 44, 78 };
10    //вычисляем длину массива по формуле массив / один элемент
11    const int lengthArray = sizeof(array) / sizeof(array[0]);
12    //инициализируем переменную максимальный индекс значением 0
13    int maxIndex = 0;
14    //цикл для вычисления максимального индекса массива
15    for (int index = 0; index < lengthArray; ++index)
16        if (array[index] > maxIndex)
17            maxIndex = array[index];
18    cout << "Индекс наибольшего элемента maxIndex : " << maxIndex << '\n';
19    return 0;
20 }
```

[Ответить](#)



7. Вячеслав:

[27 марта 2019 в 22:26](#)

вот первое задание:

```
1 #include "pch.h"
2 #include <iostream>
3
4 int main()
5 {
6     using namespace std;
7     //объявляем массив
8     int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
9     //вычисляем длину массива по формуле массив / на элемент
10    const int lengthOfArray = sizeof(array) / sizeof(array[0]);
```

```
11 //цикл for для вывода индекса массива
12     for (int index = 0; index < lengthOfArray; ++index)
13         cout << array[index] << " ";
14     return 0;
15 }
```

[Ответить](#)



8. Алена:

[7 октября 2018 в 16:32](#)

Такая проверка ввода излишне?

```
1  #include "pch.h"
2  #include <iostream>
3
4  int getUserNumber()
5  {
6      while (true)
7      {
8          std::cout << "Введите число от 1 до 9: ";
9          int userNumber;
10         std::cin >> userNumber;
11         if (std::cin.fail())
12         {
13             std::cin.clear();
14             std::cin.ignore(32676, '\n');
15             std::cout << "Error. Try again. ";
16         }
17         else
18         {
19             std::cin.ignore(32676, '\n');
20             if (userNumber >= 1 && userNumber <= 9)
21                 return userNumber;
22             else
23                 std::cout << "Error. Try again. ";
24         }
25     }
26 }
27
28 int main()
29 {
30     setlocale(LC_ALL, "Rus");
31     int userNumber = getUserNumber();
32     int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
33     const int lengthArray = sizeof(array) / sizeof(array[0]);
34     for (int index = 0; index < lengthArray; ++index)
35         std::cout << array[index] << " ";
```

```

36     for (int index = 0; index < lengthArray; ++index)
37         if (array[index] == userNumber)
38             std::cout << "\nИндекс элемента в массиве - " << index;
39     return 0;
40 }

```

[Ответить](#)



1. *Павел:*

[12 мая 2020 в 17:51](#)

А какой смысл в дальнейшем выполнении и прочих else. Протестируй код без него и увидишь, что он и так отлично работает, однако прочие куски не несут никакой логической нагрузки и попросту вредят.

[Ответить](#)



9. *Роман:*

[12 сентября 2018 в 13:29](#)

Мне кажется не обязательно делать проверку if(std::cin.fail()).

И можно совместить вывод массива и поиск индекса.

```

1  #include<iostream>
2
3  int main()
4  {
5      std::cout << "Enter a number at 1 to 9: ";
6
7      int num;
8      do
9      {
10         std::cin >> num;
11         std::cin.clear();
12         std::cin.ignore(32767, '\n');
13     } while (num < 1 || num > 9);
14
15     int array[] = { 7, 5, 6, 4, 9, 8, 2, 1, 3 };
16
17     int index;
18     int const length = sizeof(array) / sizeof(array[0]);
19     for (int i = 0; i < length; i++)
20     {
21         std::cout << array[i] << " ";
22         if (num == array[i])
23             index = i;
24     }
25

```

```

26 |         std::cout << "\nYour number has index " << index;
27 |     }

```

[Ответить](#)



10. *Torgu:*

[5 июля 2018 в 18:59](#)

Все время сталкиваюсь с одной и той же проблемой, но только сейчас решил спросить. Зачастую не получается использовать уникальную инициализацию в циклах/условиях. Конкретно есть следующий отрывок кода из второго задания:

```

1 | int index;
2 | for (int count{ 0 }; count < lengthArray; ++count)
3 | {
4 |     if (array[count] == num)
5 |         index{ count };
6 | }

```

Строчка "index{ count }" выдает ошибку "требуется точка с запятой", и только стоит мне изменить на прямую инициализацию "index = count", так все идеально работает. На уникальную инициализацию существуют какие-то ограничения?

[Ответить](#)



1. *Torgu:*

[20 июля 2018 в 18:16](#)

UPD: если кому-то интересно, понял в чем проблема на опыте: { } — инициализация, а не присвоение значения, то есть данные скобки можно использовать только один раз для конкретно переменных — при ее инициализации. А я пытался использовать второй раз

[Ответить](#)



11. *Алибек:*

[17 мая 2018 в 14:49](#)

2 задача:

```

1 | #include "stdafx.h"
2 | #include <iostream>
3 | int main()
4 | {
5 |     int array[] = { 7,5,6,4,9,8,2,1,3 };
6 |     int arrayLen = sizeof(array) / sizeof(array[0]);
7 |     while (true)
8 |     {
9 |         std::cout << "Enter a number: ";

```

```
10     int num;
11     std::cin >> num;
12     if (std::cin.fail())
13     {
14         std::cin.clear();
15         std::cin.ignore(32767, '\n');
16     }
17     else if (num < 1 || num > 9)
18     {
19         continue;
20     }
21     else
22     {
23         int index=-1;
24         for (int i = 0; i < arrayLen; i++)
25         {
26             int x = array[i];
27             std::cout << x<<'\t';
28             if (num == x)
29             {
30                 index = i;
31             }
32         }
33         std::cout << std::endl;
34         std::cout << "Index of " << num << " is " << index << '\n';
35         break;
36     }
37 }
38 return 0;
39 }
```

[Ответить](#)

12. Валерий:

[26 августа 2017 в 11:00](#)

Много же тебе еще переводить! Хватит ли духу до конца? Вообще нужное дело!

[Ответить](#)

R

1. Юрий:

[26 августа 2017 в 11:46](#)

Уже можно сказать рефлекс выработался 😊 Сам не знаю, пока будет свободное время — буду переводить.

[Ответить](#)

Добавить комментарий

Ваш E-mail не будет опубликован. Обязательные поля помечены *

Имя *

Email *

Комментарий






☐ Сохранить моё Имя и E-mail. Видеть комментарии, отправленные на модерацию

☐ Получать уведомления о новых комментариях по электронной почте. Вы можете [подписаться](#) без комментирования.

[TELEGRAM](#)  [КАНАЛ](#)

[ПАБЛИК](#) 

ТОП СТАТЬИ

-  [Словарь программиста. Сленг, который должен знать каждый кодер](#)
-  [Урок №1. Введение в программирование](#)
-  [70+ бесплатных ресурсов для изучения программирования](#)
-  [Урок №1: Введение в создание игры «SameGame» на C++/MFC](#)
-  [Урок №4. Установка IDE \(Интегрированной Среды Разработки\)](#)

- [Ravesli](#)
- - [О проекте/Контакты](#) -
- - [Пользовательское Соглашение](#) -
- - [Все статьи](#) -
- Copyright © 2015 - 2020