

Ravesli [Ravesli](#)


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


Урок №86. Динамические массивы

 [Юрий](#) |

- [Уроки C++](#)

|

 Обновл. 17 Сен 2020 |

 86266

[↑](#)  45

Помимо **динамического выделения переменных** мы также можем динамически выделять и массивы. В отличие от **фиксированного массива**, где его размер должен быть известен во время компиляции, динамическое выделение массива в языке C++ позволяет нам устанавливать его длину во время выполнения программы.

Оглавление:

1. [Динамические массивы](#)
2. [Удаление динамического массива](#)
3. [Инициализация динамических массивов](#)
4. [Изменение длины массивов](#)
5. [Тест](#)

Динамические массивы

Для выделения динамического массива и работы с ним используются отдельные формы операторов new и delete: new[] и delete[].

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Enter a positive integer: ";
6     int length;
```

```
7   std::cin >> length;
8
9   int *array = new int[length]; // используем оператор new[] для выделения массива.
10
11  std::cout << "I just allocated an array of integers of length " << length << '\n';
12
13  array[0] = 7; // присваиваем элементу под индексом 0 значение 7
14
15  delete[] array; // используем оператор delete[] для освобождения выделенной массив
16  array = 0; // используйте nullptr вместо 0 в C++11
17
18  return 0;
19 }
```

Поскольку мы выделяем массив, то C++ понимает, что он должен использовать другую форму оператора `new` — форму для массива, а не для переменной. По факту вызывается оператор `new[]`, даже если мы и не указываем `[]` сразу после ключевого слова `new`.

Обратите внимание, поскольку память для динамических и фиксированных массивов выделяется из разных «резервуаров», то размер динамического массива может быть довольно большим. Вы можете запустить программу, приведенную выше, но уже выделить массив длиной 1 000 000 (или, возможно, даже 100 000 000) без проблем. Попробуйте!

Удаление динамического массива

При удалении динамических массивов также используется форма оператора `delete` для массивов — `delete[]`. Таким образом, мы сообщаем процессору, что ему нужно очистить память от нескольких переменных вместо одной. Самая распространенная ошибка, которую совершают новички при работе с динамическим выделением памяти, является использование `delete` вместо `delete[]` для удаления динамических массивов. Использование формы оператора `delete` для переменных при удалении массива приведет к таким неожиданным результатам, как повреждение данных, утечка памяти, сбой или другие проблемы.

Инициализация динамических массивов

Если вы хотите инициализировать динамический массив значением `0`, то всё довольно просто:

```
1 int *array = new int[length]();
```

До C++11 не было простого способа инициализировать динамический массив ненулевыми значениями (список инициализаторов работал только с фиксированными массивами). А это означает, что нужно перебрать каждый элемент массива и присвоить ему значение явным указанием:

```
1 int *array = new int[5];
2 array[0] = 9;
3 array[1] = 7;
```

```
4 array[2] = 5;  
5 array[3] = 3;  
6 array[4] = 1;
```

Немного утомляет, не правда ли?

Однако, начиная с C++11, появилась возможность инициализации динамических массивов через списки инициализаторов:

```
1 int fixedArray[5] = { 9, 7, 5, 3, 1 }; // инициализируем фиксированный массив  
2 int *array = new int[5] { 9, 7, 5, 3, 1 }; // инициализируем динамический массив
```

Обратите внимание, в синтаксисе динамического массива между длиной массива и списком инициализаторов оператора присваивания (=) нет.

В C++11 фиксированные массивы также могут быть инициализированы с использованием [uniform-инициализации](#):

```
1 int fixedArray[5] { 9, 7, 5, 3, 1 }; // инициализируем фиксированный массив в C++11  
2 char fixedArray[14] { "Hello, world!" }; // инициализируем фиксированный массив в C++11
```

Однако, будьте осторожны, так как в C++11 вы не можете инициализировать динамический массив символов [строкой C-style](#):

```
1 char *array = new char[14] { "Hello, world!" }; // не работает в C++11
```

Вместо этого вы можете динамически выделить [std::string](#) (или выделить динамический массив символов, а затем с помощью функции `strcpy_s()` скопировать содержимое нужной строки в этот массив).

Также обратите внимание на то, что динамические массивы должны быть объявлены с явным указанием их длины:

```
1 int fixedArray[] {1, 2, 3}; // ок: неявное указание длины фиксированного массива  
2  
3 int *dynamicArray1 = new int[] {1, 2, 3}; // не ок: неявное указание длины динамического  
4  
5 int *dynamicArray2 = new int[3] {1, 2, 3}; // ок: явное указание длины динамического ма
```

Изменение длины массивов

Динамическое выделение массивов позволяет задавать их длину во время выделения. Однако C++ не предоставляет встроенный способ изменения длины массива, который уже был выделен. Но и это ограничение можно обойти, динамически выделив новый массив, скопировав все элементы из старого массива, а затем удалив старый массив. Однако этот способ подвержен ошибкам (об этом чуть позже).

К счастью, в языке C++ есть массивы, размер которых можно изменять, и называются они векторами (`std::vector`). О них мы поговорим на соответствующем уроке.

Тест

Напишите программу, которая:

- спрашивает у пользователя, сколько имен он хочет ввести;
- просит пользователя ввести каждое имя;
- вызывает функцию для сортировки имен в алфавитном порядке (измените код сортировки методом выбора из [урока №77](#));
- выводит отсортированный список имен.

Подсказки:

- Используйте динамическое выделение `std::string` для хранения имен.
- `std::string` поддерживает сравнение строк с помощью операторов сравнения `<` и `>`.

Пример результата выполнения вашей программы:

```
How many names would you like to enter? 5
Enter name #1: Jason
Enter name #2: Mark
Enter name #3: Alex
Enter name #4: Chris
Enter name #5: John
```

Here is your sorted list:

```
Name #1: Alex
Name #2: Chris
Name #3: Jason
Name #4: John
Name #5: Mark
```

Ответ

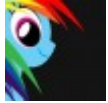
```
1 #include <iostream>
2 #include <string>
3 #include <utility> // для std::swap(). Если у вас не поддерживается C++11, то тогда #i
4
5 void sortArray(std::string *array, int length)
6 {
7     // Перебираем каждый элемент массива
8     for (int startIndex = 0; startIndex < length; ++startIndex)
9     {
10         // smallestIndex - индекс наименьшего элемента, с которым мы столкнулись
11         int smallestIndex = startIndex;
12
13         // Ищем наименьший элемент, который остался в массиве (начиная со startIndex+1,
```

```
14     for (int currentIndex = startIndex + 1; currentIndex < length; ++currentIndex)
15     {
16         // Если текущий элемент меньше нашего ранее найденного наименьшего элемента
17         if (array[currentIndex] < array[smallestIndex])
18             // то тогда это новое наименьшее значение в этой итерации
19             smallestIndex = currentIndex;
20     }
21
22     // Меняем местами наш начальный элемент с найденным наименьшим элементом массива
23     std::swap(array[startIndex], array[smallestIndex]);
24 }
25 }
26
27 int main()
28 {
29     std::cout << "How many names would you like to enter? ";
30     int length;
31     std::cin >> length;
32
33     // Выделяем массив для хранения имен
34     std::string *names = new std::string[length];
35
36     // Просим пользователя ввести все имена
37     for (int i = 0; i < length; ++i)
38     {
39         std::cout << "Enter name #" << i + 1 << ": ";
40         std::cin >> names[i];
41     }
42
43     // Сортируем массив
44     sortArray(names, length);
45
46     std::cout << "\nHere is your sorted list:\n";
47     // Выводим отсортированный массив
48     for (int i = 0; i < length; ++i)
49         std::cout << "Name #" << i + 1 << ": " << names[i] << '\n';
50
51     delete[] names; // не забываем использовать оператор delete[] для освобождения памяти
52     names = nullptr; // используйте 0, если не поддерживается C++11
53
54     return 0;
55 }
```

Оценить статью:

[← Урок №85. Динамическое выделение памяти](#)[Урок №87. Указатели и const →](#)

Комментариев: 45



1. *Александр:*
[1 октября 2020 в 16:00](#)

вместо:

```
1 std::string *names = new std::string[length];
```

удобнее использовать:

```
1 auto *names = new std::string[length];
```

[Ответить](#)



2. *Vlad:*
[6 сентября 2020 в 22:34](#)

```
1 #include <iostream>
2
3 int main()
4 {
5     {
6         using namespace std;
7
8         cout << "<< Enter length: ";
9         int length; cin >> length;
10
11         string* names = new string[length];
12         for (int index = 0; index < length; index++)
13         {
14             cout << "<< Enter name #" << index + 1 << ": ";
15             cin >> names[index];
16         }
17
18         for (int iteration = 0; iteration < length - 1; iteration++)
19         {
```

```

20         for (int index = 0; index < length - iteration - 1; index++)
21         {
22             int next = index + 1;
23             if (names[index] > names[next])
24                 swap(names[index], names[next]);
25         }
26     }
27
28     for (int index = 0; index < length; index++)
29         cout << ">> Name #" << index + 1 << ": " << names[index] << "\n";
30
31     delete[] names;
32     names = nullptr;
33 }
34
35 return 0;
36 }

```

[Ответить](#)



3. *Onium:*

[4 июля 2020 в 23:23](#)

Странно, но у меня (Visual studio 2019, C++17) работает этот код и без `#include <utility>` и `#include <string>`. Почему так?

```

1  #include <iostream>
2
3  int main()
4  {
5      std::cout << "How many names would you like to enter? ";
6      int countOfNames;
7      std::cin >> countOfNames;
8      std::string *names = new std::string[countOfNames];
9
10     for (int count = 0; count < countOfNames; ++count)
11     {
12         std::cout << "Enter name #" << count+1 << ": ";
13         std::cin >> names[count];
14
15     }
16
17     for (int startIndex = 0; startIndex < countOfNames - 1; ++startIndex)
18     {
19         int smallestIndex = startIndex;
20
21         for (int currentIndex = startIndex + 1; currentIndex < countOfNames;

```

```

23         {
24             if (names[currentIndex] < names[smallestIndex])
25                 smallestIndex = currentIndex;
26         }
27         std::swap(names[startIndex], names[smallestIndex]);
28     }
29
30
31     std::cout << '\n';
32     std::cout << "Here is your sorted list: " << '\n';
33
34     for (int index = 0; index < countOfNames; ++index)
35         std::cout << "Name #" << index + 1 << ": " << names[index] << ' ' << '\n'
36
37     return 0;
38 }

```

[Ответить](#)

1. Роман:

[9 июля 2020 в 15:49](#)

Iostream автоматически подключает string, насчёт utility не знаю, но скорее всего аналогично

[Ответить](#)

2. Виталий:

[11 октября 2020 в 09:57](#)

у меня код тоже работает без подключения этих библиотек

[Ответить](#)

4. Ватрі:

[17 июня 2020 в 02:05](#)

Может сгодится:

```

1 #include <iostream>
2 #include <string>
3
4 std::string* fill_arr(int n)
5 {
6     std::string* arr_name = new std::string[n];
7
8     std::string temp;
9     for (int i = 0; i < n; ++i)

```



```
10     {
11         std::cout << "Enter name #" << i + 1 << ": ";
12         std::cin >> *(arr_name + i);
13     }
14
15     return arr_name;
16 }
17
18 void sort_arr(std::string* arr_str, int n)
19 {
20     for (int i = 0; i < n - 1; ++i)
21     {
22         for (int j = i + 1; j < n; ++j)
23         {
24             if (arr_str[i] > arr_str[j])
25             {
26                 std::swap(arr_str[i], arr_str[j]);
27             }
28         }
29     }
30 }
31
32 void show_arr(const std::string* arr_name, int n)
33 {
34     std::cout << std::endl;
35     std::cout << "Here is your sorted list:" << std::endl;
36     for (int i = 0; i < n; ++i)
37     {
38         std::cout << "Name #" << i + 1 << ": " << *(arr_name + i) << std::endl;
39     }
40 }
41
42 int main()
43 {
44     int n;
45     std::string* arr_name = nullptr;
46     std::cout << "How many names would you like to enter?: ";
47     std::cin >> n;
48
49     if (n > 1)
50     {
51         arr_name = fill_arr(n);
52         sort_arr(arr_name, n);
53         show_arr(arr_name, n);
54     }
55     else
56     {
57         arr_name = fill_arr(n);
58         show_arr(arr_name, n);
```

```

59     }
60
61     delete[] arr_name;
62     arr_name = nullptr;
63
64     return EXIT_SUCCESS;
65 }

```

[Ответить](#)1. *ARTYR:*[19 августа 2020 в 04:05](#)

Спасибо!

А вот почему это работает?

```
1 | std::cin >> *(arr_name + i);
```

это как arr_name[i]

[Ответить](#)5. *Павел:*[31 мая 2020 в 19:43](#)

Хмм я сделал как и в ответе, но добавил элемент тестирования кода. Еще ни все проверки додумал (к примеру у нас тут целое число, а я использую вместо int, double, т.к. стараюсь сделать проверку максимально универсальной).

function.cpp

```

1 | #include<iostream>
2 | using namespace std;
3 |
4 | double input()
5 | {
6 |     double a;
7 |     cin >> a;
8 |
9 |
10 |     while (a <= 0) {
11 |
12 |         if (cin.fail())
13 |         {
14 |             cin >> a;
15 |             cin.clear();
16 |             cin.ignore(32767, '\n');
17 |             cout << "Вы ввели некорректное значение, не являющемся целочисленным :

```

```

18         }
19         else if (cin.good())
20             cout << "Вы ввели значение меньше или равное 0.\nПовторите ввод: ";
21             cin >> a;
22
23     }
24
25     return a;
26 }

```

function.h

```

1 #ifndef FUNCTION_H
2 #define FUNCTION_H
3
4 double input();
5
6 #endif

```

ConsoleApplication.cpp

```

1 #ifndef FUNCTION_H
2 #define FUNCTION_H
3
4 double input();
5
6 #endif

```

[Ответить](#)6. *Степан:*[10 мая 2020 в 04:12](#)

Люди почему-то любят всё усложнять с сортировкой, а достаточно использовать `std::sort` для этого:

```

1 #include "stdafx.h"
2 #include <iostream>
3 #include <Windows.h>
4 #include <string>
5 #include <utility>
6 #include <algorithm>
7
8
9 int main()
10 {
11     using namespace std;
12     // РУССКИЙ ЯЗЫК
13     SetConsoleCP(1251);
14     SetConsoleOutputCP(1251);

```

```
15
16     static int x = 0;
17
18     cout << "Введите кол-во имён: " << endl;
19     cin >> x;
20
21     string *names = new string[x];
22
23     for (int i = 0; i < x; i++)
24     {
25         cout << "Введите имя № " << i + 1 << endl;
26         cin >> names[i];
27     }
28
29     cout << "Сортировка... \n\n";
30
31     sort(names, names + x);
32
33     for (int i = 0; i < x; i++)
34     {
35         cout << "Имя № " << i << ": " << names[i] << endl;
36     }
37
38     return 0;
39 }
```

[Ответить](#)



7. Yerda:

[23 апреля 2020 в 14:34](#)

```
1  #include <iostream>
2  #include <string>
3
4  int getNumber() {
5      std::cout << "How many names would you like to enter? ";
6      int chislo;
7      std::cin >> chislo;
8      return chislo;
9  }
10
11 std::string getName(int x) {
12
13     std::cout << "Enter name #" << x+1<<" ";
14     std::string name;
15     std::cin >> name;
16     return name;
17 }
```

```

18
19 }
20
21 void sortAtoZ(std::string names[],int x) {
22     std::string key;
23     for (int iii = 0; iii < x-1; ++iii){
24         for (int jjj = 0; jjj < x-iii-1; ++jjj)
25             {
26                 if (names[jjj][0] > names[jjj+1][0])
27                     {
28                         key = names[jjj];
29                         names[jjj] = names[jjj + 1];
30                         names[jjj + 1] = key;
31                     }
32             }
33     }
34 }
35
36 int main()
37 {
38     int n = getNumber();
39     std::string* names = new std::string[n];
40     for (int iii = 0; iii < n; iii++)
41     {
42         names[iii]=getName(iii);
43     }
44
45
46
47     sortAtoZ(names, n);
48
49     Старался использовать функции, где возможно
50
51     std::cout << "Here is your sorted list: " << std::endl;
52     for (int iii = 0; iii < n; iii++)
53     {
54         std::cout << "Name #" << iii+1 << ": " << names[iii] << std::endl;
55     }
56
57     return 0;
58 }

```

[Ответить](#)



8.  Богдан:

[12 марта 2020 в 20:52](#)

Я просто використав функцію `std::sort()` для сортування по алфавіту. Чесно, і в готовій відповіді мені не зрозуміло, як порівнюються букви знаками більше менше. Вони якось неявно конвертуються в тип `int` і зв'язуються по коду ASCII?

```
1 #include<iostream>
2 #include <string>
3 #include<algorithm> //для std::sort
4
5 int main()
6 {
7     std::cout << "How many names you wanna enter?: ";
8     int SIZE;
9     std::cin >> SIZE;
10
11     // створення динамічного масиву типу string
12     std::string *name = new std::string[SIZE];
13     for (int i = 0; i < SIZE; i++)
14     {
15         std::cout << "enter name #" << i + 1 << ": "; //i+1 бо індексація масиву
16         std::cin >> name[i];
17     }
18
19     std::cout << std::endl;
20
21     // виведення введеного масиву
22     for (int i = 0; i < SIZE; i++)
23     {
24         std::cout << " name #" << i + 1 << ": " << name[i] << std::endl;
25     }
26     std::cout << std::endl;
27
28     // функція сортування даних
29     std::sort(name, name+SIZE);
30
31     std::cout << std::endl;
32     // виведення відсортованого масиву
33     for (int i = 0; i < SIZE; i++)
34     {
35         std::cout << " name #" << i + 1 << ": " << name[i] << std::endl;
36     }
37
38     std::cout << std::endl;
39
40     delete[] name;
41     name = nullptr;
42
43     system("pause");
44     return 0;
45 }
```

| }

[Ответить](#)

1. Павел:

[31 мая 2020 в 19:04](#)

Да всё верно. за каждой буквой стоит значение в цифровом эквиваленте, вот они то и сравниваются))

[Ответить](#)

1. Богдан:

[9 июня 2020 в 20:38](#)

дякую)

[Ответить](#)

9. Арбузик❤❤❤:

[7 февраля 2020 в 11:08](#)

А чем я хуже?

```

1  #include <iostream>
2  #include <string>
3  #include <utility>
4  using namespace std;
5
6  int main(int argc, char *argv[])
7  {
8      int n;
9      cout << "Enter value of names: ";
10     cin >> n;
11     string* arr = new string[n];
12     for (int i = 0; i < n; ++i)
13     {
14         cout << "Enter a name: ";
15         cin >> arr[i];
16     }
17     for (int i = 0; i < n - 1; ++i)
18     {
19         int min = i;
20         for (int j = i + 1; j < n; ++j)
21         {
22             for (int k = 0; k < arr[j].size(); ++k)
23             {

```

```

24         if (arr[min][k] > arr[j][k])
25         {
26             min = j;
27             break;
28         } else if (arr[min][k] < arr[j][k])
29             break;
30     }
31 }
32 swap(arr[i], arr[min]);
33 }
34 cout << "Sortirated: " << endl;
35 for (int i = 0; i < n; ++i)
36     cout << arr[i] << endl;
37 return 0;
38 }

```

[Ответить](#)



10. *Inviser666:*

[30 января 2020 в 17:49](#)

и так сойдёт)

```

1  #include <iostream>
2  #include <utility>
3  #include <cstdlib>
4  #include <string>
5
6  int main() {
7      setlocale(LC_ALL, "rus");
8      std::cout << "Введите количество имён: ";
9      int array_length;
10     std::cin >> array_length;
11     std::cin.clear();
12     std::cin.ignore(32767, '\n');
13     std::string* array = new std::string[array_length];
14
15     //запрос ввода имён
16     for (int index = 0; index < array_length; index++) {
17         std::string name;
18         std::cout << "Введите имя #" << index + 1 << ": ";
19         std::getline(std::cin, name);
20         array[index] = name;
21     }
22
23     //сортировка
24     for (int startIndex = 0; startIndex < array_length - 1; startIndex++) {
25         for (int currentIndex = startIndex + 1; currentIndex < array_length; curr

```



```

26         if (array[currentIndex]<array[startIndex]) {
27             std::swap(array[startIndex], array[currentIndex]);
28         }
29     }
30
31 }
32
33 //ВЫВОД ИМЕН
34 std::cout << std::endl;
35 std::cout << "Отсортированный список имён:" << std::endl;
36 for (int index = 0; index < array_length; index++) {
37     std::cout <<"Имя #"<<index+1<<": "<< array[index] << std::endl;
38 }
39 delete[] array;
40 array = nullptr;
41 system("pause");
42 }

```

Ответить



11. андр:

[18 января 2020 в 15:26](#)

Поначалу хотел сделать через массив char, потом понял что это дикая головная боль и сделал как все

```

1 // lesson 86.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчива
2 //
3
4 #include <iostream>
5 #include "lesson 86.h"
6
7 void sortArray(std::string* arrayToSort, int arrayLength) // descend by default
8 {
9     for (int i = 0; i < arrayLength - 1; i++)
10     {
11         for (std::string* ptr = arrayToSort; ptr < arrayToSort + arrayLength - 1
12             {
13                 if (ptr[1] < ptr[0])
14                     std::swap(ptr[1], ptr[0]);
15             }
16         }
17     }
18 }
19
20 std::string getName()
21 {
22     std::cout << "enter Name: \n";
23     std::string name;

```

```
24     std::cin >> name;
25     return name;
26 }
27
28 int main()
29 {
30     int count;
31     std::cout << "How many names do you want to enter?\n";
32     std::cin >> count;
33     std::string* names = new std::string[count];
34     for (int i = 0; i < count; i++)
35     {
36         names[i] = getName();
37     }
38     sortArray(names, count);
39     std::cout << "Sorted array is: ";
40     for (int i = 0; i < count; i++)
41     {
42         std::cout << names[i] << ", ";
43     }
44     std::cout << "\n";
45     delete[] names;
46 }
```

[Ответить](#)



12. *Марат:*

[15 января 2020 в 05:00](#)

Мое решение!

```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  string* getArr(int size)
7  {
8      string *arr = new string[size];
9      return arr;
10 }
11
12 void setNames(string* arr, int size)
13 {
14     for (int i(0); i < size; ++i)
15     {
16         cout << "Enter name #" << i + 1 << ": ";
17         cin >> arr[i];
18     }
19 }
```

```
18     }
19 }
20
21 void sortStrArr(string* arr, int size)
22 {
23     for (int start(0); start < size - 1; ++start)
24     {
25         int small = start;
26         for (int current = start; current < size - 1; ++current)
27         {
28             if (arr[current] < arr[small])
29             {
30                 small = current;
31             }
32         }
33         swap(arr[start], arr[small]);
34     }
35 }
36
37 void printStrArr(string* arr, int size)
38 {
39     cout << "Here is your sorted list:\n";
40     for (int i(0); i < size; ++i)
41     {
42         cout << "Name #" << i + 1 << ": " << arr[i] << endl;
43     }
44 }
45
46 int main()
47 {
48     setlocale(0, " ");
49     int names;
50     cout << "How many names would yiu like to enter? ";
51     cin >> names;
52     string* arr = getArr(names);
53     setNames(arr, names);
54     sortStrArr(arr, names);
55     printStrArr(arr, names);
56     delete[] arr;
57     arr = nullptr;
58     system("pause");
59     return 0;
60 }
```

[Ответить](#)



13. *Анастасия Лузинсан:*

[30 ноября 2019 в 21:49](#)

Всё просто и без усложнений. До последнего думала, что обычное сравнение `std::string` ни к чему хорошему не приведет — что не может быть всё так просто. Оказалось, очень даже может. Запустилось с первого раза 😊

```
1 #include <iostream>
2 #include <string>
3
4 void sort(std::string *names, int count)
5 {
6     for (int i = 0; i < count-1; i++)
7     {
8         for (int w = i + 1; w < count; w++)
9         {
10             if (names[i] > names[w])
11                 std::swap(names[i], names[w]);
12         }
13     }
14 }
15
16 int main()
17 {
18     int length(1);
19
20     do{
21         std::cout << "How many names will you enter? ";
22         std::cin >> length;
23
24
25         if (std::cin.fail())
26         {
27             std::cin.clear();
28             std::cin.ignore(32767, '\n');
29             std::cout << "Enter a number greater than zero!\n\n";
30         }
31
32     } while (length <= 0);
33
34     std::string *names = new std::string[length];
35
36     for (int i = 0; i < length; i++)
37     {
38         std::cout << "\nEnter name #" << i+1 << " ";
39         getline(std::cin, names[i]);
40     }
41
42     sort(names, length);
43
44     std::cout << "\nHere is your sorted list:\n";
45 }
```

```

46     for (int i = 0; i < length; i++)
47     {
48         std::cout << "Name #" << i + 1 << " ";
49         std::cout << names[i];
50         std::cout << '\n';
51     }
52
53     delete[] names;
54     names = nullptr;
55     return 0;
56 }

```

Ответить



14. Константин:

21 ноября 2019 в 09:29

Получилось!

```

1  #include <iostream>
2  #include <string>
3  #include <utility>
4
5  int main()
6  {
7      int quNames; // переменная количества имен, которые будут введены
8      do // запрос и ввод количества имен с защитой ввода
9      {
10         std::cout << "How many names will you enter? ";
11         std::cin >> quNames;
12         if (std::cin.fail())
13         {
14             std::cin.clear();
15             std::cin.ignore(32767, '\n');
16             continue;
17         }
18
19         while (quNames <= 0); // количество имен не должно быть нулевым или отрицате
20
21         //объявляем динамический массив типа std::string для хранения вводимых имен
22         std::string* nameList = new std::string[quNames];
23
24         // приступаем к вводу имен
25         std::cout << "\nNow enter names\n\n";
26         std::cin.ignore(32767, '\n');
27         for (int index = 0; index < quNames; ++index)
28         {
29             std::cout << "Enter the name #" << index + 1 << ": ";

```

```
30     std::getline(std::cin, nameList[index]);
31 }
32 // приступаем к сортировке имен
33 std::string minName;
34 for (int startIndex = 0; startIndex < quNames-1; ++startIndex)
35 {
36     int smallestIndex=startIndex;
37     for (int currentIndex = startIndex+1; currentIndex < quNames; ++currentIndex)
38     {
39         if (nameList[currentIndex] < nameList[smallestIndex])
40             smallestIndex=currentIndex;
41     }
42     std::swap(nameList[startIndex], nameList[smallestIndex]) ;
43 }
44
45 // выводим массив имён
46 std::cout << "\nHere is your sorted list:\n";
47 for (int index = 0; index < quNames; ++index)
48     std::cout << nameList[index] << '\n';
49
50 // удаляем динамический массив и висячий указатель
51 delete[] nameList;
52 nameList = nullptr;
53
54 return 0;
55 }
```

[Ответить](#)

15. zashiki:

[10 сентября 2019 в 09:20](#)

Почему имя1<имя2 сортирует по алфавиту?

Типа первую букву рассматривают как символ с ASCII числовым кодом?

[Ответить](#)

1. Дмитрий:

[26 сентября 2019 в 17:30](#)

Тот же вопрос

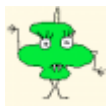
[Ответить](#)

1. Марат:

[15 января 2020 в 04:54](#)

В данном случае у класса `std::string` перегружен оператор сравнения.

[Ответить](#)



16. Алексей:

[20 августа 2019 в 13:16](#)

Подплянул, ибо некоторые вещи подзабыл.

На самом деле понимал как, просто сортировку не помню, а тут все очень просто. Добавление имен хотел через функцию, количество тоже, майн будет посвободнее.

[Ответить](#)



17. Алексей:

[1 июля 2019 в 21:11](#)

```
1 #include <iostream>
2 #include <utility>
3 #include <string>
4
5 void compare(std::string *array_0, int lenght)
6 {
7     for (int startIndex = 0; startIndex < lenght; ++startIndex)
8     {
9         int endIndex{ lenght - startIndex - 1 };
10        bool flag = false;
11        for (int currentIndex = 0; currentIndex < endIndex; ++currentIndex)
12        {
13            if (array_0[currentIndex] > array_0[currentIndex + 1])
14            {
15                std::swap(array_0[currentIndex], array_0[currentIndex + 1]);
16                flag = true;
17            }
18        }
19        if (!flag)
20            break;
21    }
22 }
23
24 int main()
25 {
26     int z;
27     std::cout << "How many would like to enter? ";
28     std::cin >> z;
29     std::string *array = new std::string[z];
30     for (int iteration = 0; iteration < z; iteration++)
31     {
```

```

32     std::cout << "Enter name #" << iteration + 1 << ": ";
33     std::cin >> array[iteration];
34 }
35 compare(array, z);
36 std::cout << "Here is your compared result: \n";
37 for (int newIteration = 0; newIteration < z; ++newIteration)
38 {
39     std::cout << "Name #" << newIteration + 1 << ": " << array[newIteration] << " ";
40 }
41 delete[] array;
42 array = nullptr;
43 system("pause");
44 return 0;
45 }

```

Не знаю, почему написано что, именно через метод сортировки. Мне, из представленных вами методов, больше всего нравится полный "пузырчатый" метод. Наверно это мой "программистский" почерк. Теперь, если я взломаю Пентагон, вы знаете как меня определить 😊

[Ответить](#)



18. Анастасия:
[27 июня 2019 в 22:48](#)

Я долго промучилась. Сначала вообще какая-то ерунда получилась, пришлось вспомнить урок об отладке программ...

Вроде как создание динамического массива выдавало исключение. Я его обработала, как учили в предыдущем уроке, и (о чудо!) оно перестало происходить.

Потом я использовала `getline` для записи строки из `cin`, но почему-то первое имя он при этом пропускал. Сделала, как в ответе, заработало нормально. Потом вернула `getline`, не знаю, что изменилось, но этот косяк исчез.

Вот мой код:

```

1  #include <iostream>
2  #include <Windows.h> // для кириллицы
3  #include <string>     // для std::string
4  #include <algorithm> // для std::swap. В C++11 используйте заголовок <utility>
5
6  using std::cout; using std::cin; using std::endl;
7
8  unsigned short lengthRequest(); // запрашивает у пользователя количество имён, к
9  void namesRequest(unsigned short namesNumber, std::string* namesArray); // запро
10 void namesSort(unsigned short namesNumber, std::string* namesArray); // Сортиров
11 void namesOutput(unsigned short namesNumber, std::string* namesArray); // Вывод
12
13 int main()
14 {
15     SetConsoleCP(1251); SetConsoleOutputCP(1251); // для ввода-вывода на кириллиц

```



```
16
17 // запрашиваем у пользователя количество имён
18 unsigned short namesNumber{lengthRequest()};
19
20 //выделяем память под массив (он же указатель) строк введённой пользователем
21 std::string* namesArray = new (std::nothrow)std::string[namesNumber];
22 // (std::nothrow) - специальная константа, которая в случае неудачи с выделени
23
24 //обработка случая, когда память выделить не удалось:
25 if (namesArray == nullptr)
26 {
27     cout << "Не удалось выделить память под массив!";
28 }
29 else //память выделили, приступаем к остальным пунктам:
30 {
31     //заполняем массив именами:
32     namesRequest(namesNumber, namesArray);
33
34     //сортируем массив методом выбора:
35     namesSort(namesNumber, namesArray);
36
37     //выводим результат:
38     cout << "Отсортированный массив имён:" << endl;
39     namesOutput(namesNumber, namesArray);
40
41     //освобождаем память, выделенную под массив строк:
42     delete[] namesArray;
43     namesArray = nullptr;
44 }
45 return 0;
46 }
47
48 // функция запрашивает у пользователя количество имён, которое и возвращает
49 unsigned short lengthRequest()
50 {
51     unsigned short namesNumber{ 0 };
52     cout << "Введите необходимое количество имён: ";
53     cin >> namesNumber;
54     cout << endl;
55     return namesNumber;
56 }
57
58 // запрос у пользователя имён, заполнение массива
59 void namesRequest(unsigned short namesNumber, std::string* namesArray)
60 {
61     cin.ignore(1000, '\n'); //очищаем cin
62     // запрос у пользователя имён, заполнение массива
63     for (unsigned short i = 0; i < namesNumber; i++)
64
```

```

65     {
66         cout << "Введите имя №" << i + 1 << ": ";
67         //записываем введенную строку в массив:
68         std::getline(cin, namesArray[i]);
69     }
70     cout << endl;
71 }
72
73 // Сортировка массива методом выбора
74 void namesSort(unsigned short namesNumber, std::string* namesArray)
75 {
76     // Перебираем каждый элемент массива
77     // (кроме последнего, он уже будет отсортирован к тому времени, когда мы до
78     for (int startIndex = 0; startIndex < (namesNumber - 1); ++startIndex)
79     {
80         // В переменной minimumIndex хранится индекс наименьшего значения, которое
81         // Начинаем с того, что наименьший элемент в этой итерации - это первый
82         int minimumIndex = startIndex;
83
84         // Затем ищем элемент поменьше в остальной части массива
85         for (int currentIndex = (startIndex + 1); currentIndex < namesNumber; ++currentIndex)
86         {
87             // Если мы нашли элемент, который больше нашего наименьшего элемента
88             if (namesArray[currentIndex] < namesArray[minimumIndex])
89                 // то запоминаем его
90                 minimumIndex = currentIndex;
91         }
92         // minimumIndex теперь индекс наименьшего элемента
93
94         // Меняем местами наше начальное наименьшее число с тем, которое мы обнаружили
95         std::swap(namesArray[startIndex], namesArray[minimumIndex]);
96     }
97 }
98
99 // Вывод элементов массива
100 void namesOutput(unsigned short namesNumber, std::string* namesArray)
101 {
102     // запрос у пользователя имён, заполнение массива
103     for (unsigned short i = 0; i < namesNumber; i++)
104     {
105         cout << "Имя №" << i + 1 << ": " << namesArray[i] << endl;
106     }
107 }

```

Ответить



1. *Nikita:*

[25 августа 2019 в 10:54](#)

Изменил в своей программе `cin` на `getline`, и тоже стало пропускать ввод первого элемента...
Почему так, не разобралась?

[Ответить](#)



2. *Nikita:*

[25 августа 2019 в 11:01](#)

Всё, понял. Не хватало строки

```
1 | std::cin.ignore(32767, '\n');
```

[Ответить](#)



19. *Денис:*

[15 июня 2019 в 02:52](#)

Честно? Подсмотрел! Но не списывал 😊 Не знал как `string` сортировать. Оказалось всё проще, чем я напридумывал с двумерными массивами:)

Добавил проверку ввода имени с большой буквы. А так всё как обычно...

```
1 | #include <iostream>
2 | #include <cstring>
3 | #include <algorithm>
4 |
5 | using namespace std;
6 |
7 | void sortingArray(string* array, int value) //Сортировка имён
8 | {
9 |     for (int startIndex = 0; startIndex < value; ++startIndex)
10 |     {
11 |         int smallIndex = startIndex;
12 |         for (int currentIndex = startIndex + 1; currentIndex < value; ++currentIndex)
13 |         {
14 |             if (array[currentIndex] < array[smallIndex])
15 |                 smallIndex = currentIndex;
16 |         }
17 |         swap(array[startIndex], array[smallIndex]);
18 |     }
19 | }
20 |
21 |
22 | int main()
23 | {
24 |     cout << "Enter number of names: ";
```

```

25     int value;
26     cin >> value;
27
28     string* name = new string[value];
29     string a_name = "a"; //Создаём переменную для проверки ввода с большой буквы
30     for (int cicle = 0; cicle < value; ++cicle) //Ввод имён
31     {
32         nameinput:
33         cout << "You enter the " << cicle + 1 << "th name out of " << value << ":
34         cin >> name[cicle];
35
36         if (name[cicle] >= a_name) //Если имя ввести с маленькой буквы, оно не буд
37         {
38             cout << "\nError: You entered the name with the first small letter. T
39             goto nameinput;
40         }
41     }
42
43     sortingArray(name, value); //Сортируем имена
44     cout << "Array successfully sorted!\n\n";
45     for (int i = 0; i < value; ++i) //Выводим отсортированный массив имён
46     {
47         cout << i + 1 << "th Name: " << name[i] << "\n";
48     }
49
50     system("pause");
51     return 0;
52 }

```

Ответить



20. alexk:

[17 января 2019 в 09:47](#)

В Вашем листинге решения ТЕСТа 34-й стейтмент выглядит вот так:

```
1 | std::string *names = new std::string[length];
```

т.е., насколько я понял, вводится динамический массив типа string.

А вот уже 40-я строка выглядит вот так:

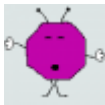
```
1 | std::cin >> names[i];
```

И вот мне НЕПОНЯТНО: откуда программа знает какой объем в байтах(или в символах) нужно программе "попросить" у ОС для names[i], чтобы "хватило" места на введение конкретного очередного имени ?

А вдруг перед компом сидит какой-нибудь ИНДИЕЦ ?! ... У них-то имена ОЧЕННО длинные !? ...



...

[Ответить](#)1. *Александр:*[22 февраля 2019 в 14:57](#)

Вы можете легко проверить через `sizeof`, что любая `std::string` занимает одинаковое количество памяти... Соответственно и выделение памяти под массив строк — не проблема.

Как такое может происходить? `std::string` состоит из нескольких частей, одна из которых — динамический массив символов (c-string). `std::string` по сути содержит только указатель на этот массив, соответственно и память выделяется только на этот указатель 😊

[Ответить](#)21. *Владимир:*[4 декабря 2018 в 02:54](#)

```

1  #include <iostream>
2  #include <stdint.h>
3  #include <string>
4
5  using std::cout;
6  using std::cin;
7  using std::string;
8
9  uint16_t inputCountOfNames() //Ввод количеств
10 {
11     cout << "How many names you want to enter?\n";
12
13     uint16_t count;
14     int16_t temp;
15
16     while (true)
17     {
18         cin >> temp;
19
20         if ((cin.fail()) || (temp < 1))
21         {
22             cin.clear();
23             cin.ignore(50000, '\n');
24
25             cout << "Input error. Unknown, too high or less than 1 input value.
26         }
27         else
28         {
29             cin.ignore(50000, '\n');
30             count = temp;
31

```

```
32         cout << '\n';
33
34         return count;
35     }
36 }
37
38 }
39
40 string *inputNames(uint16_t totalNamesCount) //Ввод имён;
41 {
42     string *strForNames{new string[totalNamesCount]};
43
44     for (uint16_t i{}; i < totalNamesCount; ++i)
45     {
46         cout << "Enter " << i + 1 << " name: ";
47
48         while (true)
49         {
50             cin >> strForNames[i];
51
52             if (cin.fail())
53             {
54                 cin.clear();
55                 cin.ignore(50000, '\n');
56                 cout << "Input error. The letters limit is exceeded. Try again.\n";
57             }
58
59             else break;
60         }
61     }
62     cout << '\n';
63
64     return strForNames;
65 }
66
67 string *bubbleSort(string *unsortedNames, uint16_t countOfNames) //Сортирует в а.
68 {
69     cout << "Bubble sorting is started...\n";
70
71     for (uint16_t i{}; i < countOfNames - 1; ++i)
72     {
73         string temp;
74
75         uint16_t iterations = countOfNames - i;
76
77         for (uint16_t j{}; j < iterations - 1; ++j)
78         {
79             if (unsortedNames[j] > unsortedNames[j + 1])
80
```

```
81         {
82             temp = unsortedNames[j];
83             unsortedNames[j] = unsortedNames[j + 1];
84             unsortedNames[j + 1] = temp;
85         }
86     }
87 }
88 cout << "Bubble sorting is complited.\n";
89 cout << '\n';
90
91 string *sortedNames = unsortedNames;
92 return sortedNames;
93 }
94
95 void printNames(string *names, uint16_t countOfNames)
96 {
97     cout << "Here is your names's list: \n";
98
99     for (uint16_t i{}; i < countOfNames; ++i)
100     {
101         cout << names[i] << '\n';
102     }
103     cout << '\n';
104 }
105
106 int main()
107 {
108     uint16_t count;
109     count = inputCountOfNames();
110
111     string *names;
112     names = inputNames(count);
113     printNames(names, count);
114
115     names = bubbleSort(names, count);
116     printNames(names, count);
117
118
119     delete[] names;
120     names = nullptr;
121     return 0;
122 }
```

//К чёрту Swap()

//Число имён;

//Список имён;

Ответить



22. Игорь:

23 сентября 2018 в 01:42

```
1 #include <iostream>
2 #include <utility>
3 int inputNumber();
4 void inputNames(std::string *uk, int n);
5 void printNames(std::string *uk, int n);
6 void sortingByChoice(std::string *uk, int n);
7 int main()
8 {
9     int n = inputNumber();
10    std::string *uk = new std::string[n];
11    inputNames(uk, n);
12    printNames(uk, n);
13    sortingByChoice(uk, n);
14    printNames(uk, n);
15
16    return 0;
17 }
18 int inputNumber()
19 {
20     int numberOfNames;
21     std::cout<<"How many names do you want to enter: ";
22     std::cin>>numberOfNames;
23
24     return numberOfNames;
25 }
26
27 void inputNames(std::string *uk, int n)
28 {
29     int i;
30     for(i = 0; i < n; i++)
31     {
32         std::cout<<"Enter name #"<<i + 1<<": ";
33         std::cin>>*uk;
34         uk++;
35     }
36     std::cout<<std::endl;
37 }
38 void printNames(std::string *uk, int n)
39 {
40     int i;
41     for(i = 0; i < n; i++)
42         std::cout<<"Name # "<<i + 1<<": "<<*uk++<<std::endl;
43
44     std::cout<<std::endl;
45 }
46 void sortingByChoice(std::string *uk, int n)
47 {
48     for (int startIndex = 0; startIndex < n - 1; ++startIndex)
49     {
```



```

50 // В переменной smallestIndex хранится индекс наименьшего значения, которое
51 // Начинаем с того, что наименьший элемент в этой итерации - это первый элемент
52 int smallestIndex = startIndex;
53
54 // Затем ищем элемент поменьше в остальной части массива
55 for (int currentIndex = startIndex + 1; currentIndex < n; ++currentIndex)
56 {
57     // Если мы нашли элемент, который меньше нашего наименьшего элемента
58     if (uk[currentIndex] < uk[smallestIndex])
59         // запоминаем его
60         smallestIndex = currentIndex;
61 }
62
63 // smallestIndex теперь наименьший элемент
64 // меняем местами наше начальное наименьшее число с тем, которое
65 std::swap(uk[startIndex], uk[smallestIndex]);
66 }
67 }

```

[Ответить](#)



23. *Mike:*

[14 сентября 2018 в 15:17](#)

Огромное Вам спасибо за данный ресурс. Вот мой вариант:

```

1  #include "stdafx.h"
2  #include <iostream>
3  #include <string>
4
5  void sort(std::string *p, int n)
6  {
7      for(int i=1; i<n; i++)
8          for(int x=1; x<n; x++)
9              if (p[x][0] < p[x - 1][0])
10                 {
11                     std::string o;
12                     o = p[x - 1];
13                     p[x - 1] = p[x];
14                     p[x] = o;
15                 }
16 }
17
18 int main()
19 {
20     std::cout << "How many names does it need?" << std::endl;
21     int n;
22     std::cin >> n;

```

```
23     std::cin.ignore(999, '\n');
24     std::string *names = new std::string[n];
25
26     for (int i = 0; i < n; i++)
27     {
28         std::cout << "Enter #" << i+1 << " name: ";
29         std::getline(std::cin, names[i]);
30     }
31     std::cout << "" << std::endl;
32
33     sort(names, n);
34
35     std::cout << "Here they are sort:" << std::endl;
36     std::cout << "" << std::endl;
37     for (int i = 0; i < n; i++)
38         std::cout << "#" << i+1 << " name: " << names[i] << std::endl;
39
40     delete[] names;
41     names = 0;
42
43
44     return 0;
45 }
```

[Ответить](#)24. *Oleksiy:*[4 сентября 2018 в 12:36](#)

Сложное решение, учитывая, что мы не проходили инициализацию указателя на массив через `std::string *`, а также никогда не делали ввод данных в массив через `std::cin>>array[i]`

[Ответить](#)1. *Mike:*[15 сентября 2018 в 19:41](#)

Согласен — чем проще, тем лучше. Читаю параллельно с этим курсом Герберта Шилдта, видимо намудрил оттуда.

[Ответить](#)25. *Данила:*[14 августа 2018 в 00:30](#)

Здравствуйте, а почему нельзя было решить тест так?

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

```
2  #include <string>
3  #include <algorithm>
4
5  int main()
6  {
7      std::cout << "How many names would you like to enter ? ";
8      int length;
9      std::cin >> length;
10     std::cin.ignore(32767, '\n');
11
12     std::string *array = new std::string[length];
13     for (int i = 0; i < length; ++i)
14     {
15         std::cout << "Name #" << i+1 << ": ";
16         std::getline(std::cin, array[i]);
17     }
18     std::sort(array, array + length);
19     for (int i = 0; i < length; ++i)
20     {
21         std::cout << array[i] << std::endl;
22     }
23     int u;
24     std::cin >> u;
25     return 0;
26 }
```

[Ответить](#)

26. *Саня:*

[9 августа 2018 в 13:25](#)

Не получается выполнять сортировку кириллицы.

в main добавлено setlocale(LC_ALL, "rus");

Вывод и ввод кириллицы все ОК. Но после сортировки выводятся непонятные символы и не сортируются судя по адресам в памяти.

Что нужно сделать?

[Ответить](#)

1. *Роман:*

[11 августа 2018 в 16:25](#)

нет, как раз таки с вводом проблемы, и кракозябры сразу же после ввода

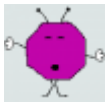
[Ответить](#)

27. *Михаил:*

[8 июля 2018 в 19:01](#)

Подскажите пожалуйста, если ввести часть имён с заглавной буквы, а часть нет сортировка не пойдёт по алфавиту. Можно ли сделать так, чтобы в любом случае сортировка шла как надо?

[Ответить](#)



1. *Александр:*

[22 февраля 2019 в 15:00](#)

Можно написать чекер с "правильной" проверкой и скормить его сортировке

```
1 bool checker(char a, char b) {...правила сравнения...}
2
3 std::sort(str.begin(), str.end(), checker);
```

[Ответить](#)



28. *Герман:*

[9 июня 2018 в 17:17](#)

Спасибо!!!

[Ответить](#)



29. *Денис:*

[1 июня 2018 в 13:21](#)

Чем дальше обучение, тем меньше комментариев. Вставлю свои пять копеек:

```
1 #include <algorithm>
2 #include <iostream>
3
4 int howMany()
5 {
6     int j;
7     std::cout << "How many names would you like to enter? ";
8     std::cin >> j;
9     return j;
10 }
11
12 void sortArray(std::string *array, int length)
13 {
14     for (int i = length; i >= 0; i--)
15     {
16         for (int j = 0; j < i-1; j++)
17         {
18             if (array[j] > array[j+1])
```

```

19         std::swap(array[j], array[j+1]);
20     }
21 }
22 std::cout << '\n' << "Here is your sorted list:" << '\n';
23 for (int l = 0; l < length; l++)
24     std::cout << "Name #" << l+1 << ": " << array[l] << '\n';
25     delete[] array;
26     array = 0;
27 }
28
29 int main()
30 {
31     int j = howMany();
32     std::string *array = new std::string[j];
33     for (int i = 0; i < j; i++)
34     {
35         std::cout << "Enter name #" << i+1 << ": ";
36         std::string st;
37         std::cin >> st;
38         array[i] = st;
39     }
40     sortArray(array, j);
41     return 0;
42 }

```

[Ответить](#)

1.  Юрий:
[1 июня 2018 в 18:40](#)

Неплохо 😊

[Ответить](#)

30.  Герман:
[29 мая 2018 в 19:49](#)

Уважаемый автор, не затруднит подсказать, как в тестовом задании изменить код, что-бы можно было вводить в массив имена с пробелом (getline у меня не прокатывает)?

[Ответить](#)

1.  Юрий:
[9 июня 2018 в 15:42](#)

Возможен такой вариант функции main():

```
1 | int main()
```

```
2  {
3      std::cout << "How many names would you like to enter? ";
4      int length;
5      std::cin >> length;
6
7      std::cin.ignore(32767, '\n');
8
9      // выделяем массив для хранения имен
10     std::string *names = new std::string[length];
11
12     // просим пользователя ввести все имена
13     for (int i = 0; i < length; ++i)
14     {
15         std::cout << "Enter name #" << i + 1 << ": ";
16         std::getline(std::cin, names[i]);
17     }
18
19     // сортируем массив
20     sortArray(names, length);
21
22     std::cout << "\nHere is your sorted list:\n";
23     // выводим отсортированный массив
24     for (int i = 0; i < length; ++i)
25         std::cout << "Name #" << i + 1 << ": " << names[i] << '\n';
26
27     delete[] names; // не забудьте использовать оператор delete для освобожд
28     names = nullptr; // используйте 0, если не поддерживается C++11
29
30     return 0;
31 }
```

Здесь используется `getline` и `cin.ignore`.

[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