

Ravesli [Ravesli](#)


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


## Урок №11. cout, cin и endl

 [Юрий](#) |

- [Уроки C++](#)

|

 Обновл. 29 Авг 2020 |

 119808

[↑](#)  40

На этом уроке мы рассмотрим такие объекты, как `cout`, `endl` и `cin`, существующие в языке C++.

Оглавление:

1. [Объект `std::cout`](#)
2. [Объект `std::endl`](#)
3. [Объект `std::cin`](#)
4. [`std::cin`, `std::cout`, `<<` и `>>`](#)

### Объект `std::cout`

Как мы уже говорили на предыдущих уроках, **объект `std::cout`** (который находится в **библиотеке `iostream`**) используется для вывода данных на экран (в консольное окно). В качестве напоминания, вот наша программа «Hello, world!»:

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hello, world!";
6     return 0;
7 }
```

Для вывода нескольких предложений на одной строке **оператор вывода `<<`** нужно использовать несколько раз, например:

```
1 #include <iostream>
2
```

```
3 int main()
4 {
5     int a = 7;
6     std::cout << "a is " << a;
7     return 0;
8 }
```

Программа выведет:

a is 7

А какой результат выполнения следующей программы?

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi!";
6     std::cout << "My name is Anton.";
7     return 0;
8 }
```

Возможно, вы удивитесь, но:

Hi!My name is Anton.

## Объект std::endl

Если текст нужно вывести раздельно (на нескольких строках) — используйте std::endl. При использовании с std::cout, **std::endl** вставляет символ новой строки. Таким образом, мы перемещаемся к началу следующей строки, например:

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hi!" << std::endl;
6     std::cout << "My name is Anton." << std::endl;
7     return 0;
8 }
```

Результат:

Hi!  
My name is Anton.

## Объект std::cin

std::cin является противоположностью std::cout. В то время как std::cout выводит данные в консоль с помощью оператора вывода <<, **std::cin** получает данные от пользователя с помощью **оператора ввода** >>. Используя std::cin мы можем получать и обрабатывать пользовательский ввод:

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Enter a number: "; // просим пользователя ввести любое число
6     int a = 0;
7     std::cin >> a; // получаем пользовательское число и сохраняем его в переменную a
8     std::cout << "You entered " << a << std::endl;
9     return 0;
10 }
```

Попробуйте скомпилировать и запустить эту программу. При запуске вы увидите `Enter a number:`, а затем программа будет ждать, пока вы укажете число. Как только вы это сделаете и нажмете Enter, программа выведет `You entered`, а затем ваше число.

Например (я ввел 7):

```
Enter a number: 7
You entered 7
```

Это самый простой способ получения данных от пользователя. Мы будем его использовать в дальнейших примерах.

Если ваше окно закрывается сразу после ввода числа — смотрите [Урок №7](#) (там есть решение данной проблемы).

Если же ввести действительно большое число, то вы получите переполнение, так как переменная `a` может содержать числа только определенного размера/диапазона. Если число больше/меньше допустимых максимумов/минимумов, то происходит переполнение. Об этом мы детально поговорим на следующих уроках.

## `std::cin`, `std::cout`, `<<` и `>>`

Новички часто путают `std::cin` с `std::cout` и `<<` с `>>`. Вот простые способы запомнить их различия:

- ➔ `std::cin` и `std::cout` всегда находятся в левой стороне стейтмента;
- ➔ `std::cout` используется для вывода значения (**c**OUT = вывод);
- ➔ `std::cin` используется для получения значения (**c**IN = ввод);
- ➔ оператор вывода `<<` используется с `std::cout` и показывает направление, в котором данные движутся от r-value в консоль. `std::cout << 7;` (значение 7 перемещается в консоль);
- ➔ оператор ввода `>>` используется с `std::cin` и показывает направление, в котором данные движутся из консоли в переменную. `std::cin >> a;` (значение из консоли перемещается в переменную `a`).

Оценить статью:



← [Урок №10. Переменные, Инициализация и Присваивание](#)

[Урок №12. Функции и оператор возврата return](#) →



## Комментариев: 40

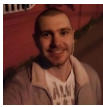


1. *Enceladus:*

[21 сентября 2019 в 19:50](#)

Читал, что болеубийца rainkiller и beksheikh пишут в Dev-C++, от себя хочу вставить пять копеек, пишу в нативном Eclipse CDT, под управлением Linux-овой осью. И по этому нет проблем типа преждевременного закрытия окна консоли 😊 Вообще, вижу, что Линуксовые ОС это идеальный полигон для изучения C++

[Ответить](#)



1. *Дмитрий Бушуев:*

[23 сентября 2019 в 11:29](#)

>>Вообще, вижу, что Линуксовые ОС это идеальный полигон для изучения C++

Если вы готовы разменять личную жизнь на линуксовые ОС, то — да. 😊

[Ответить](#)



2. *Ростик:*

[6 мая 2019 в 15:35](#)

как std::cin получает данные от пользователя с помощью оператора ввода?

[Ответить](#)



3. *beка:*

[25 марта 2019 в 14:23](#)

кто может мне помочь и объяснить мне по полочкам этот код

```
1 #include "pch.h"
2 #include <iostream>
3 #include <stdio.h>
4 #include <fstream>
5 using namespace std;
6 struct date
7 {
```

```
8     int day;
9     int month;
10    int year;
11 };
12 struct konf
13 {
14     int kod;
15     int kodpr;
16     date d;
17     int srok;
18     double vkys;
19     int nach;
20     double kal;
21     bool proizv;
22     date kons;
23 };
24 konf* Create(ifstream &f, int n)
25 {
26     int i;
27     char*b = new char[10];
28     konf* a = new konf[n];
29     for (i = 0; i < n; i++)
30     {
31         f >> a[i].kod >> a[i].kodpr;
32         f >> b;
33         a[i].d.day = (int(b[0]) - 48) * 10 + int(b[1]) - 48;
34         a[i].d.month = (int(b[3]) - 48) * 10 + int(b[4]) - 48;
35         a[i].d.year = (int(b[6]) - 48) * 1000 + (int(b[7]) - 48) * 100 + (int(
36             int(b[9] - 48);
37         f >> a[i].srok >> a[i].vkys >> a[i].nach >> a[i].kal >> a[i].proizv;
38         a[i].kons.day = a[i].d.day;
39         a[i].kons.month = a[i].d.month + a[i].srok;
40         a[i].kons.year = a[i].d.year;
41         while (a[i].kons.month > 12)
42         {
43             a[i].kons.month -= 12;
44             a[i].kons.year += 1;
45         }
46     }
47     return a;
48 }
49 bool eqals(date konfeti, date secondia)
50 {
51     if (konfeti.year > secondia.year)
52         return false;
53     else
54         if (konfeti.year < secondia.year)
55             return true;
56     else
57         if (konfeti.month > secondia.month)
58             return false;
59     else
```

```
60         if (konfeti.month < segondia.month)
61             return true;
62         else
63             if (konfeti.day >= segondia.day)
64                 return false;
65             else
66                 return true;
67     }
68 void Srok(konf* a, date dn, int n, ofstream &out, int &k)
69 {
70     int i;
71     k = 0;
72     for (i = 0; i < n; i++)
73         if (eqals(a[i].kons, dn))
74         {
75             out.write((char*)&a[i].kod, sizeof(a[i].kod));
76             out.write((char*)&a[i].kodpr, sizeof(a[i].kodpr));
77             out.write((char*)&a[i].kons, sizeof(a[i].kons));
78             k++;
79         }
80     cout << k << endl;
81 }
82 void Sort(konf* a, int n, ofstream &out, int &k)
83 {
84     int i, j;
85     konf t;
86     for (i = 0; i < n; i++)
87         for (j = 0; j < n - 1; j++)
88             if (a[j].kal < a[j + 1].kal)
89             {
90                 t = a[j];
91                 a[j] = a[j + 1];
92                 a[j + 1] = t;
93             }
94     for (i = 0; i < n; i++)
95     {
96         out.write((char*)&a[i].kod, sizeof(a[i].kod));
97         out.write((char*)&a[i].kodpr, sizeof(a[i].kodpr));
98         out.write((char*)&a[i].vkys, sizeof(a[i].vkys));
99         out.write((char*)&a[i].kal, sizeof(a[i].kal));
100     }
101     k = n;
102     cout << n << endl;
103 }
104 void Vkys(konf*a, int n, ofstream &out, int &k)
105 {
106     int i;
107     k = 0;
108     for (i = 0; i < n; i++)
109         if ((a[i].proizv) && (a[i].vkys > 80))
110         {
111             out.write((char*)&a[i], sizeof(a[i]));
```

```

112         k++;
113     }
114     cout << k << endl;
115 }
116 void Nachinka(konf*a, int n, ofstream &out, int &k)
117 {
118     int i;
119     k = 0;
120     for (i = 0; i < n; i++)
121         if ((a[i].vkys > 0) && (a[i].vkys < 50) && (a[i].nach % 3 == 0))
122         {
123             out.write((char*)&a[i].kod, sizeof(a[i].kod));
124             out.write((char*)&a[i].kodpr, sizeof(a[i].kodpr));
125             out.write((char*)&a[i].d, sizeof(a[i].d));
126             out.write((char*)&a[i].vkys, sizeof(a[i].vkys));
127             out.write((char*)&a[i].nach, sizeof(a[i].nach));
128             k++;
129         }
130     cout << k << endl;
131 }
132 void Prov1(istream &f, int n)
133 {
134     cout << "РѹСБPsPIРѹСБPeP° 1: " << endl;
135     konf* a = new konf[n];
136     for (int i = 0; i < n; i++)
137     {
138         f.read((char*)&a[i].kod, sizeof(a[i].kod));
139         f.read((char*)&a[i].kodpr, sizeof(a[i].kodpr));
140         f.read((char*)&a[i].kons, sizeof(a[i].kons));
141         cout << a[i].kod << ' ' << a[i].kodpr << ' ' << a[i].kons.day <<
142             '.' << a[i].kons.month << '.' << a[i].kons.year << endl;
143     }
144 }
145 void Prov2(istream &f, int n)
146 {
147     cout << "РѹСБPsPIРѹСБPeP° 2: " << endl;
148     konf* a = new konf[n];
149     for (int i = 0; i < n; i++)
150     {
151         f.read((char*)&a[i].kod, sizeof(a[i].kod));
152         f.read((char*)&a[i].kodpr, sizeof(a[i].kodpr));
153         f.read((char*)&a[i].vkys, sizeof(a[i].vkys));
154         f.read((char*)&a[i].kal, sizeof(a[i].kal));
155         cout << a[i].kod << ' ' << a[i].kodpr << ' ' << a[i].vkys << ' ' << a[i].kal << endl;
156     }
157 }
158 void Prov3(istream &f, int n)
159 {
160     cout << "РѹСБPsPIРѹСБPeP° 3: " << endl;
161     konf* a = new konf[n];
162     for (int i = 0; i < n; i++)
163     {

```

```

164     f.read((char*)&a[i], sizeof(a[i]));
165     cout << a[i].kod << ' ' << a[i].kodpr << ' ' << a[i].d.day << '.' << a[i].d.month
166         << '.' << a[i].d.year << ' ' << a[i].srok << ' ' << a[i].vkys << ' ' << a[i].nach
167         << ' ' << a[i].kal << ' ' << a[i].proizv << ' ' << a[i].kons.day << ' ' << a[i].kons.month
168         << '.' << a[i].kons.year << endl;
169 }
170 }
171 void Prov4(ifstream &f, int n)
172 {
173     cout << "Результаты сортировки: " << endl;
174     konf* a = new konf[n];
175     for (int i = 0; i < n; i++)
176     {
177         f.read((char*)&a[i].kod, sizeof(a[i].kod));
178         f.read((char*)&a[i].kodpr, sizeof(a[i].kodpr));
179         f.read((char*)&a[i].d, sizeof(a[i].d));
180         f.read((char*)&a[i].vkys, sizeof(a[i].vkys));
181         f.read((char*)&a[i].nach, sizeof(a[i].nach));
182         cout << a[i].kod << ' ' << a[i].kodpr << ' ' << a[i].d.day <<
183             '.' << a[i].d.month << '.' << a[i].d.year << ' ' << a[i].vkys <<
184             << a[i].nach << endl;
185     }
186 }
187 int main()
188 {
189     setlocale(LC_ALL, "Russian");
190     ifstream in("in.txt");
191     ofstream out1("output1.bin", ios::binary);
192     ofstream out2("output2.bin", ios::binary);
193     ofstream out3("output3.bin", ios::binary);
194     ofstream out4("output4.bin", ios::binary);
195     int n, k1, k2, k3, k4;
196     cout << "Введите количество записей: ";
197     cin >> n;
198     konf* a = Create(in, n);
199     char* b = new char[10];
200     date d;
201     cout << "Введите дату (дд.мм.гггг): ";
202     cin >> b;
203     d.day = (int(b[0]) - 48) * 10 + int(b[1]) - 48;
204     d.month = (int(b[3]) - 48) * 10 + int(b[4]) - 48;
205     d.year = (int(b[6]) - 48) * 1000 + (int(b[7]) - 48) * 100 + (int(b[8]) - 48) +
206         int(b[9] - 48);
207     Srok(a, d, n, out1, k1);
208     Sort(a, n, out2, k2);
209     Vkys(a, n, out3, k3);
210     Nachinka(a, n, out4, k4);
211     in.close();
212     out1.close();
213     out2.close();
214     out3.close();
215     out4.close();

```



```

216     ifstream f1("output1.bin", ios::binary);
217     ifstream f2("output2.bin", ios::binary);
218     ifstream f3("output3.bin", ios::binary);
219     ifstream f4("output4.bin", ios::binary);
220     Prov1(f1, k1);
221     Prov2(f2, k2);
222     Prov3(f3, k3);
223     Prov4(f4, k4);
224 }

```

Ответить1. *YU:*[23 февраля 2020 в 11:51](#)

Возможно после прочтения 100 урока, я смогу ответить на твой вопрос.

Ответить2. *Мари:*[30 марта 2020 в 12:21](#)

Объяснить, пока, не могу, но юзание транслита в коде — это боль

Ответить4. *Георгий:*[1 февраля 2019 в 21:12](#)

Юрий, я работаю в VS 2015 и создаю пустой проект без заголовка. Если я ввожу в cin число больше чем int, у меня выводится 0 и ошибки переполнения нет, а если не больше int, то выводится введённое число. Если я пробую вывести на консоль не инициализированную переменную, то ничего не выводится, нет мусорного значения и пустой строки, только "Для продолжения ...". Это проблема с IDE или компилятором или что-то другое?

Ответить5. *Олег:*[21 августа 2018 в 18:41](#)

Здравствуйте. Я не давно начал изучать программирование в целом. Не могли бы вы объяснить почему переменной а мы присвоили значение 0

Ответить1. *Юрий:*[21 августа 2018 в 21:12](#)

Привет. Мы таким образом инициализировали а, обнулив её. В этом примере впринципе без этого можно обойтись, но в более сложных программах обнуление используется для

того, чтобы предотвратить случайное или "мусорное" значение, которое могло быть присвоено переменной а где-то в коде раньше.

[Ответить](#)



2. *Алексей:*

[21 января 2019 в 00:10](#)

Написав ноль в переменную, мы таким образом выделили место в памяти и присвоили переменной пустое значение. Если же это место уже было выделено и там находилось какое-то значение, присвоением мы его заменили на новое.

[Ответить](#)



6. *Sergey Groysman:*

[28 марта 2018 в 21:17](#)

Маленький вопрос вдогонку: на будущее, чтобы сократить место в комментарии, как вы вставляете PmScr со своими стейтментами?

[Ответить](#)



1. *Юрий:*

[29 марта 2018 в 18:09](#)

PmScr — это, я так понял, код? Код вставляется через специальный плагин в админке этого сайта, каждый код в комментарии я модерую самостоятельно. Вы можете вставлять всё, как есть, за экономию места — не переживайте.

Если же у вас слишком большой код, то воспользуйтесь [сервисом OnlineGDB](#) — там можно сохранять/компилировать свой код на C++. Вставили свой код, запустили его (проверили) и нажали на кнопку *Share* — вам выдадут ссылку, по которой вы сможете поделиться этим кодом. Тогда здесь в комментариях уже указываете только эту ссылку на ваш код и задаете свои вопросы.

[Ответить](#)



1. *master114:*

[3 мая 2018 в 14:29](#)

а копировать нужно

Share Code:

или

Embed Code:

и нужно ли добавлять какие-нибудь дополнительные теги к этим ссылкам чтобы код в комментариях отображался корректно?

[Ответить](#)



1. *Юрий:*  
[4 мая 2018 в 11:54](#)

Копировать ссылку нужно с Share Code. Добавлять в комментарии дополнительные теги не нужно — вставили ссылку, свой комментарий и всё.



7. *Sergey Groysman:*  
[28 марта 2018 в 21:11](#)

День добрый, Юрий.

Спасибо за ваши уроки.

Преамбула вопроса — при выполнении данной функции у меня вводимая переменная d отображается при её вводе и потом повторно уже в команде cout. Вопрос: как её убрать из видимости при вводе и оставить только в команде cout?

Спасибо.

```
1 int doubleNumber(int d)
2 {
3     return d * 2;
4 }
5 int main()
6 {
7     std::cout << "Please, Enter a number, press button ENTER and we DOUBLE you NUMB
8     int d;
9     std::cin >> d;
10    std::cout << "You are entered the number is: " << d << std::endl;
11    std::cout << "Your DOUBLE number is: " << doubleNumber(d) << std::endl;
12    return 0;
13 }
```

### [Ответить](#)



1. *Юрий:*  
[29 марта 2018 в 17:57](#)

Привет. Чтобы скрыть вводимое значение, нужно добавить заголовочный файл:

```
1 #include <Windows.h>
```

И прописать следующие строчки в функции main():

```
1 HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
2 DWORD mode = 0;
3 GetConsoleMode(hStdin, &mode);
4 SetConsoleMode(hStdin, mode & (~ENABLE_ECHO_INPUT));
```

Готовый код:

```
1 #include "stdafx.h"
```

```

2  #include <iostream>
3  #include <Windows.h>
4
5
6  int doubleNumber(int d)
7  {
8      return d * 2;
9  }
10 int main()
11 {
12     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
13     DWORD mode = 0;
14     GetConsoleMode(hStdin, &mode);
15     SetConsoleMode(hStdin, mode & (~ENABLE_ECHO_INPUT));
16
17     std::cout << "Please, Enter a number, press button ENTER and we DOUBLE
18     int d;
19     std::cin >> d;
20     std::cout << std::endl;
21     std::cout << "You are entered the number is: " << d << std::endl;
22     std::cout << "Your DOUBLE number is: " << doubleNumber(d) << std::endl;
23     return 0;
24 }

```

[Ответить](#)

1. *Sergey Groysman:*  
[29 марта 2018 в 20:06](#)

Спасибо.

[Ответить](#)

1. *Юрий:*  
[29 марта 2018 в 21:28](#)

Пожалуйста 😊



8. *Анна:*  
[26 марта 2018 в 16:41](#)

Нигде не могу найти, в чем смысл данной строки? просто возврат каретки?

```
1 | cout << endl;
```

[Ответить](#)

1. *Юрий:*  
[26 марта 2018 в 17:02](#)

endl — это перенос на новую строку и очистка потока, аналог переноса на новую строку — 'n'. cout << endl, как отдельная строчка, используется просто для переноса на новую строку, двойной cout << endl используется для пробела между строками (результат виден в консоли). Т.е. можно в конце строки добавить 'n':

```
1 std::cout << "Hey!\n";
2 std::cout << "My name is Anton.";
```

Либо

```
1 std::cout << "Hey!";
2 std::cout << std::endl;
3 std::cout << "My name is Anton.";
```

Результат один и тот же, просто endl еще и очищает за собою поток данных.

[Ответить](#)



1. *Анна:*

[27 марта 2018 в 10:17](#)

Спасибо! Теперь все понятно 😊

[Ответить](#)



9. *beksheikh:*

[9 марта 2018 в 12:22](#)

Здравствуйте.объясните если возможно для чего надо ставить std?можно же и без него..работаю на дев с++.и еще для чего нужен std::cin(get)?..если хотите могу написать полный код.С уважением Я)).

[Ответить](#)



1. *Юрий:*

[9 марта 2018 в 13:07](#)

Привет, об std:: говорится в [уроке 24](#) и в [уроке 54](#). Следуйте порядку, в котором изложены уроки и всё узнаете — всё, что вам нужно знать.

Об std::cin(get) первый раз слышу, возможно std::cin.get()?

[Ответить](#)



1. *beksheikh:*

[9 марта 2018 в 19:29](#)

Да.)не правильно написал

[Ответить](#)



1. *Юрий:*  
[9 марта 2018 в 21:07](#)

`cin.get()` считывает из входного потока данных один символ, используется для "задержания" консольного окна.



2. *painkiller:*  
[29 марта 2018 в 14:15](#)

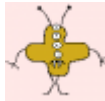
Тоже работаю в этой IDE — она более простая и понятная для новичка!

Чтобы не было никаких проблем с преждевременным закрытием окна консоли, нужно перед последней фигурной скобкой (}) писать следующее:

`cin.ignore();` — последнее нажатие ENTER игнорируется;

`cin.get();` — ждёт этого нажатия от Вас;

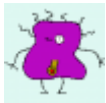
[Ответить](#)



1. *Данила:*  
[21 ноября 2018 в 21:21](#)

сделал как вы написали — ничего не вышло.

[Ответить](#)



10. *Иван:*  
[5 марта 2018 в 20:59](#)

Здравствуй. Подскажи как поместить в одну строку два ввода `cin`?  
 Заранее спасибо.

[Ответить](#)



1. *Юрий:*  
[6 марта 2018 в 21:21](#)

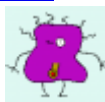
Привет. Объясни кодом, на примере, что ты конкретно хочешь. Разместить два стейтмента `cin` можно на одной строке

```
1 | cin >> x; cin >> y;
```

Между ними можно вставить `cout` и разместить всё также на одной строке:

```
1 | cin >> x; cout >> x; cin >> y;
```

[Ответить](#)



1. *Иван:*  
[6 марта 2018 в 21:40](#)

Мне нужно, чтобы 2 ввода, в консоли, на одной строчке были. Хочу, чтобы в консоли было примерно так:

Калькулятор

Введите цифру:  $5 * 6 = 30$

### Ответить

**R**

1. *Юрий:*  
[7 марта 2018 в 23:39](#)

Во-первых, вы пробовали гуглить или искать ответ на StackOverflow?

Во-вторых, вы можете делать вводы через `cin` на одной строчке в консоли, просто оставляя пробел между значениями. Например:

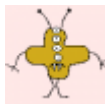
```
1 int x, y;  
2  
3 std::cin >> x;  
4 std::cin >> y;
```

В консоли вы можете вводить значения так:

7 8

и нажимать Enter.

В итоге у вас будет  $x = 7$  и  $y = 8$ . Что делать, чтобы выводить результат умножения этих двух значений на этой же строке, где вы ввели 7 и 8 — я не знаю. Вы должны обязательно нажать Enter, чтобы завершить ввод чисел для `cin`, а когда вы нажимаете Enter, то вы автоматически переходите на следующую строку. Решения, как это всё обойти, я не нашел.



2. *Данила:*  
[23 ноября 2018 в 15:31](#)

вот попробуйте так в строке ставить знак арифметического действия надо самому —

```
1 int main()  
2 {  
3     std::cout << "enter a ,y"; // ВВОД ЧИСЕЛ  
4  
5     int a = 0;  
6     std::cin >> a;  
7     int y = 0;  
8     std::cin >> y;  
9     std::cout << a + y; // ЭТО ОТВЕТ  
10  
11 }
```

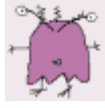


2. *Алексей:*

[21 января 2019 в 00:19](#)

А почему в "cout" направление стрелок не в ту сторону ">>" ?

[Ответить](#)



1. *Виталий:*

[24 февраля 2019 в 16:17](#)

Здравствуйте Юрий!

Скажите, будут ли эти программы работать на Андроиде( cppdroid) ?



11. *Кату:*

[3 марта 2018 в 14:42](#)

У меня не отображаются цифры и слова после Enter a number, а когда я поменяла cin на cout всё появилось. Не могли бы вы мне подсказать, что с этим делать?

[Ответить](#)



1. *Юрий:*

[3 марта 2018 в 16:50](#)

Я не очень понял ваш вопрос. На будущее, если у вас возникают вопросы по вашей программе — прикрепляйте код.

Cin нужен для ввода данных (слов, цифр, букв, предложений), cout для вывода. Если вы только указываете cin в коде, то выводиться ничего не будет. Если вы указываете только cout, то выводиться будет то, что вы указали вместе с cout. То есть с помощью cin вывести вы не сможете ничего, так как он для этого не предназначен.

[Ответить](#)



12. *Александр:*

[22 января 2018 в 22:23](#)

Здравствуйте.

Подскажите, после команд

`std::cin >>`

`std::cout <<`

для чего нужны кавычки, ведь и так понятно что,

`std::cin` — это ввод

`std::cout` — это вывод

Это просто такой синтаксис или эти команды также используются с другими символами и соответственно имеют другое предназначение ?



Начинаю изучение программирования с нуля, поэтому не хочу оставлять пробелов в понимании.

[Ответить](#)



1. *Юрий:*

[22 января 2018 в 22:59](#)

Привет, кавычки "" являются частью синтаксиса, если вы их не укажете, то получите синтаксические ошибки. "" используются для вывода символов с std::cout в консольное окно, кавычки с std::cin не используются.

[Ответить](#)



1. *Александр:*

[23 января 2018 в 07:58](#)

Я не однозначно сформулировал вопрос, я имел ввиду не эти кавычки "", а вот эти << и >>.

Это просто такой синтаксис или команды

std::cin

std::cout

могут использоваться с другими символами и соответственно иметь другое значение ?

[Ответить](#)



1. *Юрий:*

[23 января 2018 в 11:33](#)

Скобки показывают направление потока данных, >> — поток данных идёт внутрь, << — поток данных идёт изнутри. Скобки являются частью синтаксиса. Cin и cout всегда используются со скобками, другого значения, кроме как ввода/вывода данных у cin/cout нет.

## 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.

Отправить комментарий






TELEGRAM  КАНАЛ

Электронная почта



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 «Same Game»](#)
-  [Урок №4. Установка IDE \(Интегрированной Среды Разработки\)](#)

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