

Ravesli [Ravesli](#)


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


Урок №90. Оператор доступа к членам через указатель

 [Юрий](#) |

- [Уроки C++](#)

|

 Обновл. 25 Сен 2020 |

 23762

[↑](#)  6

Обычно есть либо [указатель](#), либо [ссылка](#) на [структуру](#)/класс. Как мы уже знаем из предыдущих уроков, доступ к члену структуры осуществляется через **оператор выбора члена** (.) (или «*оператор доступа к члену*»):

```
1 struct Man
2 {
3     int weight;
4     double height;
5 };
6 Man man;
7
8 // Доступ к члену осуществляется через использование фактической переменной структуры M
9 man.weight = 60;
```

Этот синтаксис также работает и со ссылками:

```
1 struct Man
2 {
3     int weight;
4     double height;
5 };
6 Man man; // определяем переменную структуры Man
7
8 // Доступ к члену осуществляется через ссылку на переменную структуры Man
```

```
9 | Man &ref = man;  
10 | ref.weight = 60;
```

Однако, в случае с указателем, вам нужно его сначала разыменовать:

```
1 | struct Man  
2 | {  
3 |     int weight;  
4 |     double height;  
5 | };  
6 | Man man;  
7 |  
8 | // Доступ к члену осуществляется через указатель на переменную структуры Man  
9 | Man *ptr = &man;  
10 | (*ptr).weight = 60;
```

Обратите внимание, разыменование указателя должно находиться в круглых скобках, поскольку оператор выбора члена имеет более высокий [приоритет](#), чем оператор разыменования.

Поскольку синтаксис доступа к членам структур/классов с помощью указателя несколько неудобен, то C++ предоставляет второй **оператор выбора членов** (->) для осуществления доступа к членам через указатель. Следующие две строки идентичны:

```
1 | (*ptr).weight = 60;  
2 | ptr->weight = 60;
```

Это не только легче писать, но и этот способ так же менее подвержен ошибкам, поскольку здесь разыменование выполняется неявно, поэтому нет проблем с приоритетом, о котором нужно помнить. Следовательно, при доступе к членам структур или классов через указатель, всегда используйте оператор -> вместо оператора ..

Правило: При использовании указателя для доступа к значению члена структуры или класса используйте оператор «->» вместо оператора «.».

Оценить статью:

★★★★★ (221 оценок, среднее: 4,96 из 5)

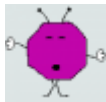


← [Урок №89. Ссылки и const](#)

[Урок №91. Цикл foreach](#) →



Комментариев: 6



1. *Александр:*

[22 февраля 2019 в 15:53](#)

тем, кто сомневается в удобстве синтаксиса -> можно предложить сравнить:

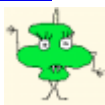
```
1 | head -> tail -> tail -> getData();
```

с

```
1 | ((*(*head).tail).tail).getData();
```

оставим за скобками идиотизм подобного вызова 😊 Но в принципе то подобное возможно! Что читается проще и наглядней? :)))

[Ответить](#)



1. *Алексей:*

[20 августа 2019 в 15:04](#)

Не видел структуры, но мне лично сложно такое читать.

C -> идет все ступенчато.

[Ответить](#)



2. *alex_1988:*

[9 февраля 2020 в 22:10](#)

В Microsoft Visual Studio можно писать через точку, среда разработки сама исправит на нужный оператор.

[Ответить](#)



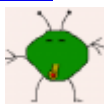
1. *alex_1988:*

[9 февраля 2020 в 22:19](#)

И добавлю что у тебя указан неверный пример. Для указателя оператор разыменования или выбора члена нужно укзывать только один раз, остальной доступ осуществляется только через точку:

```
1 | head->tail.tail.getData();
2 | (*head).tail.tail.getData();
```

[Ответить](#)



1. *boba:*

[20 мая 2020 в 15:03](#)

если каждый член указатель?



2. *Viktor:*

[20 сентября 2020 в 14:10](#)

Ты не прав. Это зависит от того, является ли член указателем

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