

НУЛЕВОЙ указатель в C

Уровень сложности : Средний • Последнее обновление : 14 Авг, 2019

На очень высоком уровне мы можем думать о NULL как о нулевом указателе, который используется в C для различных целей. Некоторые из наиболее распространенных вариантов использования NULL

: а) Инициализация переменной указателя, когда этой переменной указателя еще не присвоен допустимый адрес памяти.

б) Проверка наличия нулевого указателя перед обращением к любой переменной указателя. Таким образом, мы можем выполнять обработку ошибок в коде, связанном с указателем, например, в переменной указателя разыменования, только если она не равна NULL.

в) Передавать нулевой указатель аргументу функции, когда мы не хотим передавать какой-либо допустимый адрес памяти.

Примером а является

```
int * pInt = NULL;
```

Примером б является

```
if(pInt != NULL) /*We could use if(pInt) as well*/  
{ /*Some code*/}  
else  
{ /*Some code*/}
```

Примером с является

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

.

Start Your Coding Journey Now!

[Login](#)[Register](#)

неинициализированные и висячие указатели в конкретном контексте программы все неинициализированные, висячие или нулевые указатели недопустимы, но NULL-это конкретный недопустимый указатель, который упоминается в стандарте C и имеет определенные цели. Мы имеем в виду, что неинициализированные и висячие указатели недействительны, но они могут указывать на некоторый адрес памяти, который может быть доступен через непреднамеренный доступ к памяти.

```
#include <stdio.h>
int main()
{
    int *i, *j;
    int *ii = NULL, *jj = NULL;
    if(i == j)
    {
        printf("This might get printed if both i and j are same by chance.");
    }
    if(ii == jj)
    {
        printf("This is always printed coz ii and jj are same.");
    }
    return 0;
}
```

Конкретно упоминая НУЛЕВОЙ указатель, стандарт C дает механизм, с помощью которого программист C может использовать и проверять, является ли данный указатель законным или нет. Но что такое NULL и как он определяется? Строго говоря, NULL расширяется до определенной реализацией константы нулевого указателя, которая определяется во многих заголовочных файлах, таких как "*stdio.h*", "*stddef.h*", "*stdlib.h*" и т. Д. Давайте посмотрим, что говорят стандарты C о нулевом указателе. Из стандарта C11 пункт 6.3.2.3,

"Целочисленное постоянное выражение со значением 0 или такое выражение,

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

Start Your Coding Journey Now!

[Login](#)[Register](#)

называется стандартом C11. Для полноты картины отметим, что предыдущими стандартами для C были C99, C90 (также известный как ISO C) и C89 (также известный как ANSI C). Хотя фактический стандарт C11 можно приобрести у ISO, есть проект документа, который доступен в открытом доступе бесплатно.

Возвращаясь к нашему обсуждению, макрос NULL определяется как `((void *)0)` в заголовочных файлах большинства реализаций компилятора C. Но стандарт C говорит, что 0 также является константой нулевого указателя. Это означает, что следующее также совершенно законно в соответствии со стандартом.

```
int * ptr = 0;
```

Обратите внимание, что 0 в приведенном выше операторе C используется в контексте указателя и отличается от 0 как целое число. Это одна из причин, почему использование NULL предпочтительнее, поскольку в коде явно указано, что программист использует нулевой указатель, а не целое число 0. Еще одна важная концепция NULL заключается в том, что *"NULL расширяется до константы нулевого указателя, определенной реализацией"* Это утверждение также взято из пункта 7.19 C11. Это означает, что внутреннее представление нулевого указателя может быть ненулевым битовым шаблоном для передачи НУЛЕВОГО указателя. Вот почему NULL всегда не должен быть внутренне представлен как битовый шаблон всех нулей. Реализация компилятора может выбрать представление "константы нулевого указателя" в виде битового шаблона для всех 1 или чего-либо еще. Но опять же, как программисту на Си, нам не нужно сильно беспокоиться о внутреннем значении нулевого указателя, если только мы не участвуем в кодировании компилятора или даже ниже уровня кодирования. Сказав это, обычно NULL представляется как все биты, установленные только в 0. Чтобы узнать это на конкретной платформе, можно использовать следующее

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

.

Start Your Coding Journey Now!

[Login](#)[Register](#)

компилятора/платформы C. Вы можете попробовать несколько других вещей в приведенной выше программе, таких как `printf("%c", NULL)` или `printf("%s", NULL)` и даже `printf("%f", NULL)`. Выходы из них будут отличаться в зависимости от используемой платформы, но это было бы интересно, особенно использование `%fc NULL`!

Можем ли мы использовать *оператор `sizeof()`* для NULL в C? Ну, использование `sizeof(NULL)` разрешено, но точный размер будет зависеть от платформы.

```
#include<stdio.h>
int main()
{
    printf("%lu", sizeof(NULL));
    return 0;
}
```

Поскольку NULL определяется как `((void*)0)`, мы можем думать о NULL как о специальном указателе, и его размер будет равен любому указателю. Если размер указателя платформы равен 4 байтам, то результат работы вышеприведенной программы будет равен 4 байтам. Но если размер указателя на платформе равен 8 байтам, то выход вышеприведенной программы будет равен 8.

А как насчет разыменования NULL? Что произойдет, если мы используем следующий код на языке Си

```
#include<stdio.h>
int main()
{
    int * ptr = NULL;
    printf("%d", *ptr);
    return 0;
}
```

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

.

Start Your Coding Journey Now!

[Login](#)[Register](#)

поговорим о типе `void`. Согласно стандартному пункту 6.2.5 C11, *"Тип `void` содержит пустой набор значений; это неполный тип объекта, который не может быть завершен"*. Даже в пункте 6.5.3.4 C11 упоминается, что *"Оператор `sizeof` не должен применяться к выражению, имеющему тип функции или неполный тип, к заключенному в скобки имени такого типа или к выражению, обозначающему член битового поля"*. Это неполный тип, размер которого не имеет никакого смысла в программах на языке Си, но реализации (такие как `gcc`) могут выбрать `sizeof(void)` как 1, так что плоская память, указанная указателем `void`, может рассматриваться как нетипизированная память, то есть последовательность байтов. Но вывод следующего не обязательно должен быть одинаковым на всех платформах.

```
#include<stdio.h>
int main()
{
    printf("%lu", sizeof(void));
    return 0;
}
```

В `gcc` вышеизложенное приведет к выводу 1. А как насчет `sizeof(void *)`? Здесь C11 упомянул руководящие принципы. Из пункта 6.2.5 следует, что *"Указатель на `void` должен иметь те же требования к представлению и выравниванию, что и указатель на символьный тип"*. Вот почему результат следующего будет таким же, как и любой размер указателя на машине.

```
#include<stdio.h>
int main()
{
    printf("%lu", sizeof(void *));
    return 0;
}
```

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

Start Your Coding Journey Now!


[Login](#)[Register](#)

Пожалуйста, поставьте лайк/Tweet/G+1, если вы найдете вышесказанное полезным. Кроме того, пожалуйста, оставьте нам комментарий для получения дальнейших разъяснений или информации. Мы хотели бы помочь и научиться 😊

```
If (Coding)
{
  C foundation course = true;
  Focus = 100;
}
cout << "Success";
```

Wait no more!

Start Learning



Like 94



[Структуры данных](#) [Алгоритмы](#) [Подготовка к собеседованию](#) [Тематическая практика](#) [C+](#)

РЕКОМЕНДУЕМЫЕ СТАТЬИ

Страница : 1 2 3

01 **Что такое указатель на нулевой указатель**
28, 19 августа

05 **Передача NULL в printf в C**
23, 16 марта

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

Start Your Coding Journey Now!

[Login](#)[Register](#)

U4 **Указатель на массив | Array
Pointer**
12, Jun 17

U8 **Как объявить указатель на
функцию?**
01, Sep 09

Article Contributed By :



GeeksforGeeks

Vote for difficulty

Current difficulty : [Medium](#)

[Easy](#)[Normal](#)[Medium](#)[Hard](#)[Expert](#)

Improved By : [nishant2raj](#)

Article Tags : [C-Pointers](#), [cpp-pointer](#), [C Language](#)

[Improve Article](#)[Report Issue](#)

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

.

Start Your Coding Journey Now!

[Login](#)[Register](#)

Company

- About Us
- Careers
- In Media
- Contact Us
- Privacy Policy
- Copyright Policy

News

- Top News
- Technology
- Work & Career
- Business
- Finance
- Lifestyle

Web Development

- Web Tutorials
- Django Tutorial
- HTML
- CSS
- JavaScript
- Bootstrap

Learn

- Algorithms
- Data Structures
- SDE Cheat Sheet
- Machine learning
- CS Subjects
- Video Tutorials

Languages

- Python
- Java
- CPP
- Golang
- C#
- SQL

Contribute

- Write an Article
- Improve an Article
- Pick Topics to Write
- Write Interview Experience
- Internships
- Video Internship

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)

Start Your Coding Journey Now!

[Login](#)[Register](#)

Мы используем файлы cookie, чтобы обеспечить вам наилучший опыт просмотра нашего веб-сайта. Используя наш сайт, вы подтверждаете, что прочитали и поняли нашу [Политику использования файлов cookie](#) и [Политику конфиденциальности](#)