Winter Tickets Sale! C++ Data Types C++ Input/Output C++ Arrays C++ Pointers C++ OOPs C++ STL C++ Interview Questions C++ Programs

160 Days of DSA

Share Your Experiences

reinterpret_cast in C++ | Type Casting operators

const_cast in C++ | Type Casting operators

Casting Operators in C++

C/C++ Ternary Operator - Some Interesting Observations

Types of Operator Overloading in C++

Default Assignment Operator and References in C++

reinterpret_cast in C++ | Type Casting operators

Last Updated : 30 May, 2022

reinterpret_cast is a type of casting operator used in C++.

- It is used to convert a pointer of some data type into a pointer of another data type, even if the data types before and after conversion are different.
- It does not check if the pointer type and data pointed by the pointer is same or not.

Syntax :

```
data_type *var_name = 
       reinterpret_cast <data_type *>
(pointer_variable);
```

Skip to content

Return Type

- It doesn’t have any return type. It simply converts the pointer type.

Parameters

- It takes only one parameter i.e., the source pointer variable (p in above example).

CPP

```cpp
// CPP program to demonstrate working of
// reinterpret_cast
#include <iostream>
using namespace std;

int main()
{
    int* p = new int(65);
    char* ch = reinterpret_cast<char*>(p);
    cout << *p << endl;
    cout << *ch << endl;
    cout << p << endl;
    cout << ch << endl;
```

Skip to content

```
    return 0;
}
```

Output:

```
65
A
0x1609c20
A
```

Purpose for using reinterpret_cast

1. reinterpret_cast is a very special and dangerous type of casting operator. And is suggested to use it using proper data type i.e., (pointer data type should be same as original data type).
2. It can typecast any pointer to any other data type.
3. It is used when we want to work with bits.
4. If we use this type of cast then it becomes a non-portable product. So, it is suggested not to use this concept unless required.
5. It is only used to typecast any pointer to its original type.

Skip to content

6. Boolean value will be converted into integer value i.e., 0 for false and 1 for true.

CPP

```
// CPP code to illustrate using structure
#include <bits/stdc++.h>
using namespace std;

// creating structure mystruct
struct mystruct {
    int x;
    int y;
    char c;
    bool b;
};

int main()
{
    mystruct s;

    // Assigning values
    s.x = 5;
    s.y = 10;
    s.c = 'a';
    s.b = true;

    // data type must be same during casting
    // as that of original
```

Skip to content

```
// converting the pointer of 's' to,
// pointer of int type in 'p'.
int* p = reinterpret_cast<int*>(&s);

cout << sizeof(s) << endl;

// printing the value currently pointed by *p
cout << *p << endl;

// incrementing the pointer by 1
p++;

// printing the next integer value
cout << *p << endl;

p++;

// we are casting back char * pointed
// by p using char *ch.
char* ch = reinterpret_cast<char*>(p);

// printing the character value
// pointed by (*ch)
cout << *ch << endl;

ch++;

/* since, (*ch) now points to boolean value,
so it is required to access the value using
same type conversion.so, we have used
data type of *n to be bool. */

bool* n = reinterpret_cast<bool*>(ch);
cout << *n << endl;
```

Skip to content

```
        // we can also use this line of code to
        // print the value pointed by (*ch).
        cout << *(reinterpret_cast<bool*>(ch));

        return 0;
    }
```

Output:

```
12
5
10
a
1
1
```

Program 2

CPP

```
// CPP code to illustrate the pointer reinterpret
#include <iostream>
using namespace std;
```

Skip to content

```
class A {
public:
    void fun_a()
    {
        cout << " In class A\n";
    }
};

class B {
public:
    void fun_b()
    {
        cout << " In class B\n";
    }
};

int main()
{
    // creating object of class B
    B* x = new B();

    // converting the pointer to object
    // referenced of class B to class A
    A* new_a = reinterpret_cast<A*>(x);

    // accessing the function of class A
    new_a->fun_a();
    return 0;
}
```

Output:

Skip to content

```
In class A
```

Related link :

https://www.geeksforgeeks.org/casting-operators-in-c-set-1-const_cast/
https://stackoverflow.com/questions/573294/when-to-use-reinterpret-cast” rel=”noopener” target=”_blank
http://forums.codeguru.com/showthread.php?482227-reinterpret_cast-lt-gt-and-where-can-it-be-used
https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.cbclx01/keyword_reinterpret_cast.htm
https://stackoverflow.com/questions/573294/when-to-use-reinterpret-cast

Learn in a distraction-free environment with refined, **high-quality content** and **35+ expert-led tech courses** to help you crack any interview. From programming languages and DSA to web development and data science, GeeksforGeeks

Skip to content

Premium has you covered!

Choose GeeksforGeeks Premium today and also get access to **Unlimited Article Summarization, 100% Ad free environment, A.I. Bot support** in all coding problems, and much more. Go Premium!

R Rohit... + Follow 26

Next Article

const_cast in C++ | Type Casting operators

Similar Reads

const_cast in C++ | Type Casting operators

C++ supports following 4 types of casting operators: 1. const_cast 2. static_cast 3. dynamic_cast 4....

4 min read

Casting Operators in C++

Casting operators are used for type casting in C++. They are used to convert one Skip to content other. C+...

5 min read

unordered_set operators in C++ STL

Unordered_set provides two operators in C++ STL. These are: Syntax: 1. (unordered_set &lhs ==...

5 min read

C++ Pointer Operators

Prerequisite: Pointers in C++ A pointer variable is a variable that stores the address of another variable or ...

2 min read

C++ Arithmetic Operators

Arithmetic Operators in C++ are used to perform arithmetic or mathematical operations on the operand...

3 min read

Input/Output Operators Overloading in C++

Operator Overloading is a part of Polymorphism, which enables the feature because of which we can directly...

2 min read

C++ Increment and Decrement Operators

Skip to content

Prerequisite: Operators in C++ What is a C++ increment Operator? The C++ increment operator is a unary...

4 min read

C++ Comparison Operators

Comparison operators are operators used for comparing two elements, these are mostly used with if-else...

3 min read

C++ Relational Operators

In C++ programming language, we sometimes require to compare values and expressions. This comparison...

4 min read

C++ Logical Operators

In C++ programming languages, logical operators are symbols that allow you to combine or modify conditio...

4 min read

Unary Operators In C++

Unary operators in C++ are those operators that work on a single value (operand). They perform operations...

8 min read

Skip to content

Assignment Operators In C++

In C++, the assignment operator forms the backbone of many algorithms and computational processes by...

7 min read

What are the Operators that Can be and Cannot be...

There are various ways to overload Operators in C++ by implementing any of the following types of functions: ...

4 min read

Increment (Decrement) operators require L-value...

What will be the output of the following program? #include<stdio.h> int main() { int i = 10; printf("%d", +...

1 min read

Order of operands for logical operators

The order of operands of logical operators &&, || are important in C/C++. In mathematics, logical AND, OR,...

1 min read

Overloading stream insertion (<>) operators in C++

In C++, stream insertion operator "<<" is used for output and extraction operator ">>" is used for input. We mus...

2 min read

Skip to content

How to sum two integers without using arithmetic...

Given two integers a and b, how can we evaluate the sum a + b without using operators such as +, -, ++, --,...

4 min read

Conditionally assign a value without using...

Given 4 integers a, b, y, and x, where x can assume the values of either 0 or 1 only. The following question is...

6 min read

Written version of Logical operators in C++

Can we use keywords in place of operators in C++ ? Yes, certainly, we can. The ANSI C++ Standard has propos...

2 min read

Comparing String objects using Relational Operato...

If strings are compared using relational operators then, their characters are compared lexicographically...

2 min read

Article Tags : C++ cpp-advanced cpp-pointer

Skip to content

Practice Tags : CPP

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:-
A-143, 9th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305) | Registered Address:- K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

GET IT ON Google Play
Download on the App Store

Company
About Us
Legal
Careers
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program

Explore
Job-A-Thon
Hiring Challenge
Hack-A-Thon
GfG Weekly Contest
Offline Classes (Delhi/NCR)
DSA in JAVA/C++
Master System Design
Master CP
GeeksforGeeks Videos
Geeks Community

Languages
Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial

DSA
Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
DSA Interview Questions
Competitive Programming

Data Science & ML
Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Web Technologies
HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
NodeJs
Bootstrap
Tailwind CSS

Python Tutorial
Python Programming Examples
Django Tutorial
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial

Computer Science
GATE CS Notes
Operating Systems
Computer Network
Database Management System

DevOps
Git
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design
High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp

School Subjects
Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar

Commerce
Accountancy
Business Studies
Economics
Management
HR Management
Finance
Income Tax

Skip to content

Python Interview Question

Software Engineering
Digital Logic Design
Engineering Maths

Interview Questions

Databases
SQL
MYSQL
PostgreSQL
PL/SQL
MongoDB

Preparation Corner
Company-Wise Recruitment Process
Resume Templates
Aptitude Preparation
Puzzles
Company-Wise Preparation
Companies
Colleges

Competitive Exams
JEE Advanced
UGC NET
UPSC
SSC CGL
SBI PO
SBI Clerk
IBPS PO
IBPS Clerk

More Tutorials
Software Development
Software Testing
Product Management
Project Management
Linux
Excel
All Cheat Sheets
Recent Articles

Free Online Tools
Typing Test
Image Editor
Code Formatters
Code Converters
Currency Converter
Random Number Generator
Random Password Generator

Write & Earn
Write an Article
Improve an Article
Pick Topics to Write
Share your Experiences
Internships

DSA/Placements
DSA - Self Paced Course
DSA in JavaScript - Self Paced Course
DSA in Python - Self Paced

Development/Testing
JavaScript Full Course
React JS Course
React Native Course
Django Web Development Course
Complete

Machine Learning/Data Science
Complete Machine Learning & Data Science Program - [LIVE]

Programming Languages
C Programming with Data Structures
C++ Programming Course

Clouds/Devops
DevOps Engineering
AWS Solutions Architect Certification
Salesforce Certified Administrator Course

GATE
GATE CS & IT Test Series - 2025
GATE DA Test Series 2025
GATE CS & IT Course - 2025
GATE DA Course 2025

Skip to content

C Programming Course Online - Learn C with Data Structures
Complete Interview Preparation
Master Competitive Programming
Core CS Subject for Interview Preparation
Mastering System Design: LLD to HLD
Tech Interview 101 - From DSA to System Design [LIVE]
DSA to Development [HYBRID]
Placement Preparation Crash Course [LIVE]
Full Stack Development - [LIVE]
JAVA Backend Development - [LIVE]
Complete Software Testing Course [LIVE]
Android Mastery with Kotlin [LIVE]
Data Analytics Training using Excel, SQL, Python & PowerBI - [LIVE]
Data Science Training Program - [LIVE]
Mastering Generative AI and ChatGPT
Java Programming Course
Python Full Course

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved