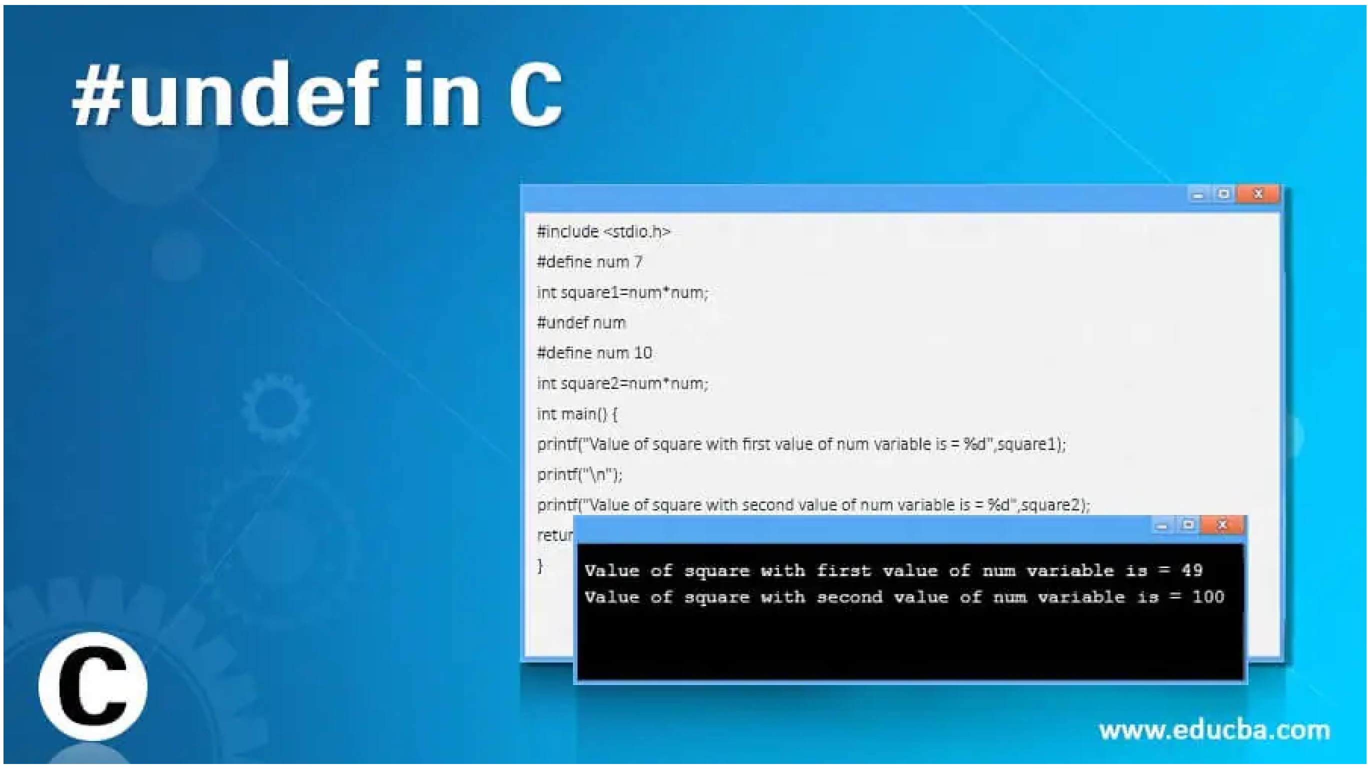
EDUCBA
(https://www.educba.com/software-development/)

← (https://www.educba.com/sharp-ifndef-in-c/)

→ (https://www.educba.com/hash-include-in-c/)



```
#include <stdio.h>
#define num 7
int square1=num*num;
#undef num
#define num 10
int square2=num*num;
int main() {
printf("Value of square with first value of num variable is = %d",square1);
printf("\n");
printf("Value of square with second value of num variable is = %d",square2);
retur
}
```

```
Value of square with first value of num variable is = 49
Value of square with second value of num variable is = 100
```

Introduction to #undef in C

Undef is a directive in C programming language that helps to remove all the definitions given macro name or any constant defined using #define directive. It is a part of the preprocessor directive since it is called by the compiler automatically before actual compilation

EDUCBA
(https://www.educba.com/software-development/)

Syntax

Preprocessors are a feature provided in C to process the source code written by the programmer before its actual compilation is done. Before the program is passed through a preprocessor compiler passes the code through the preprocessor where specific instructions such as directives are looked for in the C program known as preprocessor directives that can be easily understood by the preprocessor. These preprocessor directives are must begin with (#) sign.

The preprocessor is that part of the compiler which executes essential operations in the given code before the compiler actually compiles it. The transformations performed by the preprocessors are lexical which tells that the output of the preprocessor is in text form.

To define a macro we use below syntax

```
#define macro_name
```

Eg: #define PI 3.14

Thus when the above line is passed to the preprocessor, it assigns 3.14 value to PI vari can be used further anywhere in the program. Further, in case, we need to limit the scope for this macro_name within the program we can use the #undef directive to remove the declared

- Here macro_name refers to the name of the variable that we defined earlier and needs to be removed.
- Here # specifies that it is a preprocessor directive and is compiled using the preprocessor before the actual code is sent for the compilation to the compiler.

How does #undef work in C?

Preprocessors refer to the programs that are processed in our source code even before code enters the compiler for compilation. # under is such a command for the preprocessor.

- The source code written by the user is first sent for preprocessing to the preprocessors which generates an expanded source file with the same name as that of the program. This expanded file is further sent for the compilation to the compiler to generate an object code of the library functions and once this object code is linked to the various library functions being used , an executable (.exe) file is generated.
- When #undef macro_name command is found by the preprocessor recognized using # symbol preprocessor checks for the macro with the same name. When it finds such a name it removes the macro from the memory so that it can be used again. If a macro being used is already assigned, a compile-time error is thrown.
- One can also use #ifdef ..#endIf directive that helps to check if a particular macro name exist or not otherwise if we access a macro name on which #undef has already been run compile-time error is thrown as we can see in below example -2.

Types of Preprocessor

There are various preprocessor directives that can be defined which can be categorized below 4 main categories:

- Macros
- File Inclusion
- Conditional Compilation
- Other directives

Examples to Implement #undef in C

Below are the examples mentioned:

Example #1

Let's see what happens when we declare a num variable with value 7 and then undefine it using undef directive. Then again define it with value 10 and see how the value of variable

(https://www.educba.com/software-development/)

```
#include <stdio.h>
#define num 7
int square1=num*num;
#undef num
#define num 10
int square2=num*num;
int main() {
printf("Value of square with first value of num variable is =
%d",square1);
printf("\n");
printf("Value of square with second value of num variable is =
%d",square2);
return 0;
}
```

Output:

Example #2

In this example, we will see what happens when one tries to access a constant or macro defined using #define but have been removed using #undef directive.

Code:

```
#include <stdio.h>
```

(https://www.educba.com/software-development/)

```
printf("Value of constant num that has been removed using #undef
directive  = %d",num1);
return 0;
}
```

Output:

Example #3

In this example, we will see how we can implement define and undef directive for declaring a macro name and constant in ones program. We will use #ifdef directive to check if a particular macro exist or not and handle the situations such as failing due to call to non-existing macros.

Code:

```
#include <stdio.h>
#define StudentId 12
#undef StudentId
int main()
{
#ifdef StudentId

printf("Student with roll_no %d exists \n", StudentId);
#endif
printf("Learning preprocessor directives is fun\n");
```

EDUCBA
(https://www.educba.com/software-development/)

Example #4

This is another cause for the above example where #ifdef directive returns true and the statement gets executed.

Code:

```
#include <stdio.h>
#define StudentId 12
//#undef StudentId
int main()
{
#ifdef StudentId
printf("Student with roll_no %d exists \n", StudentId);
#endif
printf("Learning preprocessor directives is fun\n");
return 0;
}
```

Output:

QUIZ

Conclusion

EDUCBA
(https://www.educba.com/software-development/)

Recommended Articles

This is a guide to #undef in C. Here we discuss an introduction to #undef in C, syntax, how does it work with examples. You can also go through our other related articles to learn more –

1. Preprocessor Directives in C (https://www.educba.com/preprocessor-directives-in-c/)
2. Memory Allocation In C (https://www.educba.com/memory-allocation-in-c/?source=leftnav)
3. Regular Expression In C (https://www.educba.com/regular-expression-in-c/?source=leftnav)
4. Address Operator In C (https://www.educba.com/address-operator-in-c/?source=leftnav)

EDUCBA
(https://www.educba.com/software-development/)

About Us

Blog (https://www.educba.com/blog/?source=footer)

Who is EDUCBA? (https://www.educba.com/about-us/?source=footer)

Sign Up (https://www.educba.com/software-development/signup/?source=footer)

Corporate Training (https://www.educba.com/corporate/?source=footer)

Certificate from Top Institutions (https://www.educba.com/educbalive/?source=footer)

Contact Us (https://www.educba.com/contact-us/?source=footer)

Verifiable Certificate (https://www.educba.com/software-development/verifiable-certificate/?source=footer)

Reviews (https://www.educba.com/software-development/reviews/?source=footer)

Terms and Conditions (https://www.educba.com/terms-and-conditions/?source=footer)

Privacy Policy (https://www.educba.com/privacy-policy/?source=footer)

Apps

iPhone & iPad (https://itunes.apple.com/in/app/educba-learning-app/id1341654580?mt=8)

Android (https://play.google.com/store/apps/details?id=com.educba.www)

QUIZ

EDUCBA
(https://www.educba.com/software-development/)

Python Tutorials (https://www.educba.com/software-development/software-development-tutorials/python-tutorial/?source=footer)

All Tutorials (https://www.educba.com/software-development/software-development-tutorials/?source=footer)

Certification Courses

All Courses (https://www.educba.com/software-development/courses/?source=footer)

Software Development Course - All in One Bundle (https://www.educba.com/software-development/courses/software-development-course/?source=footer)

Become a Python Developer (https://www.educba.com/software-development/courses/python-certification-course/?source=footer)

Java Course (https://www.educba.com/software-development/courses/java-course/?source=footer)

Become a Selenium Automation Tester (https://www.educba.com/software-development/courses/selenium-training-certification/?source=footer)

Become an IoT Developer (https://www.educba.com/software-development/courses/iot-course/?source=footer)

ASP.NET Course (https://www.educba.com/software-development/courses/asp-net-course/?source=footer)

VB.NET Course (https://www.educba.com/software-development/courses/vb-net-course/?source=footer)

PHP Course (https://www.educba.com/software-development/courses/php-course/?source=footer)

QUIZ