Constants in C

The constants in C are the read-only variables whose values cannot be modified once they are declared in the C program. The type of constant can be an integer constant, a floating pointer constant, a string constant, or a character constant. In C language, the **const** keyword is used to define the constants.

What is a constant in C?

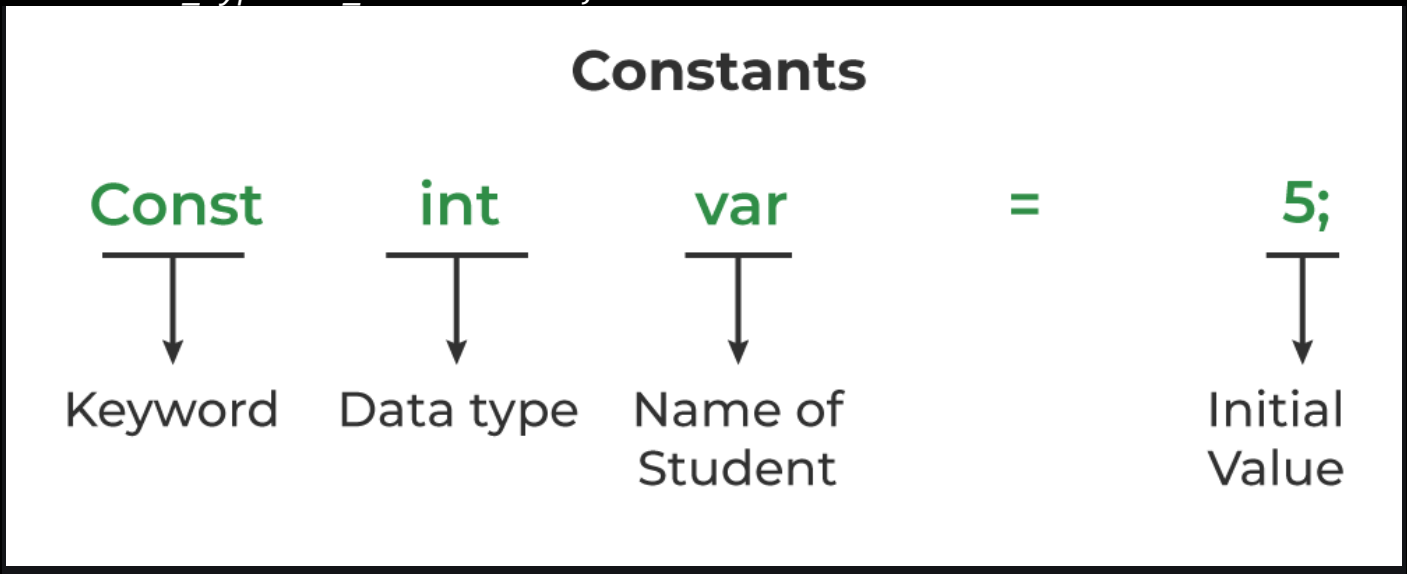
As the name suggests, a constant in C is a variable that cannot be modified once it is declared in the program. We can not make any change in the value of the constant variables after they are defined.

How to define a constant in C?

We define a constant in C language using the **const** keyword. Also known as a const type qualifier, the const keyword is placed at the start of the variable declaration to declare that variable as a constant.

Syntax to Define Constant

```
const data_type var_name = value;
```



Example of Constants in C

- C

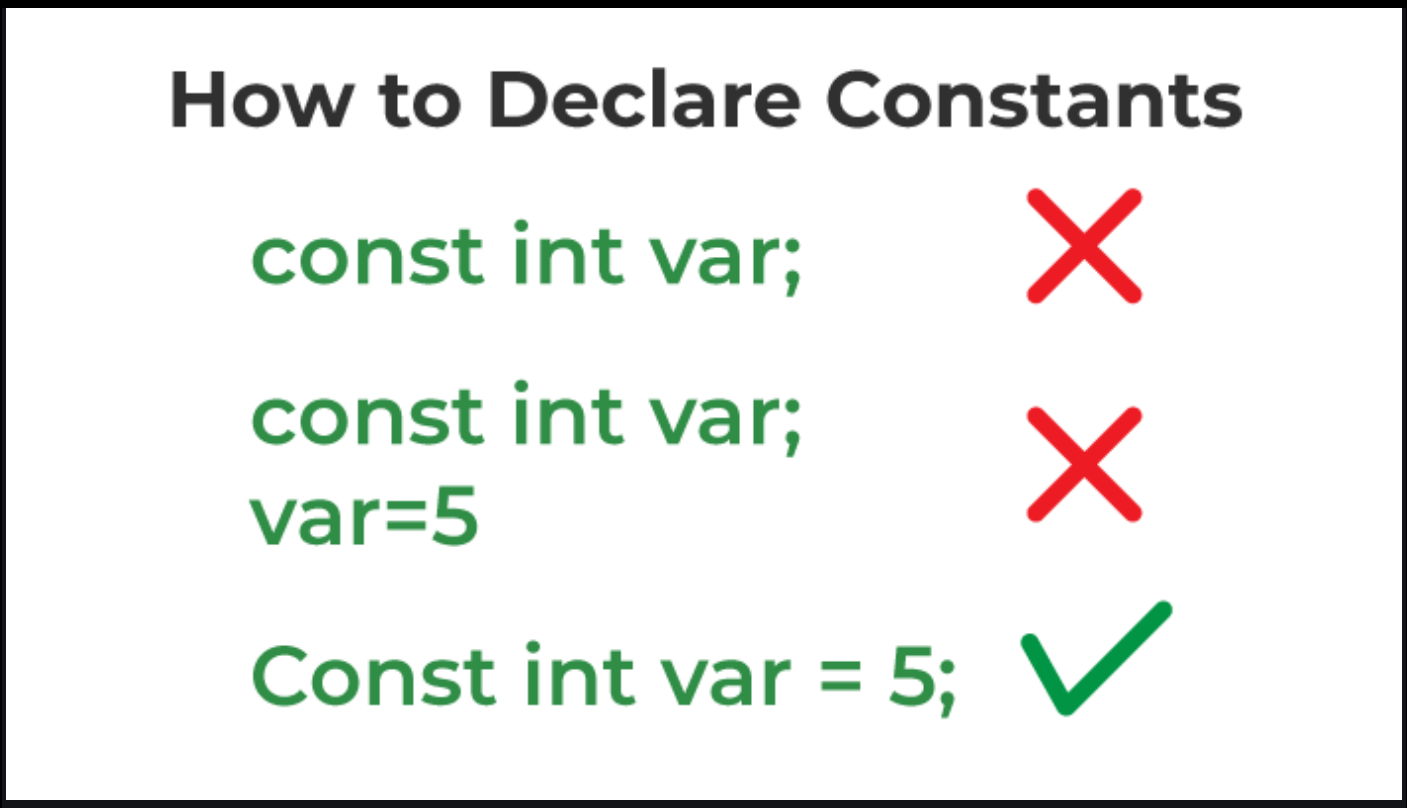
```c
// C program to illustrate constant variable definition

#include <stdio.h>


int main()

{
```

```c
    // defining integer constant using const keyword
    const int int_const = 25;

    // defining character constant using const keyword
    const char char_const = 'A';

    // defining float constant using const keyword
    const float float_const = 15.66;

    printf("Printing value of Integer Constant: %d\n",
           int_const);
    printf("Printing value of Character Constant: %c\n",
           char_const);
    printf("Printing value of Float Constant: %f",
           float_const);

    return 0;
}
```

Output

```
Printing value of Integer Constant: 25
Printing value of Character Constant: A
Printing value of Float Constant: 15.660000
```

One thing to note here is that we have to **initialize the constant variables at declaration**. Otherwise, the variable will store some garbage value and we won't be able to change it. The following image describes examples of incorrect and correct variable definitions.



Types of Constants in C

The type of the constant is the same as the data type of the variables. Following is the list of the types of constants

- Integer Constant
- Character Constant
- Floating Point Constant
- Double Precision Floating Point Constant
- Array Constant
- Structure Constant

We just have to add the const keyword at the start of the variable declaration.

Properties of Constant in C

The important properties of constant variables in C defined using the const keyword are as follows:

1. Initialization with Declaration

We can only initialize the constant variable in C at the time of its declaration. Otherwise, it will store the garbage value.

2. Immutability

The constant variables in c are immutable after its definition, i.e., they can be initialized only once in the whole program. After that, we cannot modify the value stored inside that variable.

- C

```c
// C Program to demonstrate the behaviour of constant
// variable
#include <stdio.h>

int main()
{
    // declaring a constant variable
    const int var;
    // initializing constant variable var after declaration
    var = 20;

    printf("Value of var: %d", var);
    return 0;
}
```

Output

```
In function 'main':
10:9: error: assignment of read-only variable 'var'
10 |     var = 20;
   |         ^
```

Difference between Constants and Literals

The constant and literals are often confused as the same. But in C language, they are different entities and have different semantics. The following table list the differences between the constants and literals in C:

Constant	Literals
Constants are variables that cannot be modified once declared.	Literals are the fixed values that define themselves.

Constant	Literals
Constants are defined by using the const keyword in C. They store literal values in themselves.	They themselves are the values that are assigned to the variables or constants.
We can determine the address of constants.	We cannot determine the address of a literal except string literal.
They are lvalues.	They are rvalues.
Example: const int c = 20.	Example: 24,15.5, ‘a’, “Geeks”, etc.

Defining Constant using #define Preprocessor

We can also define a constant in C using #define preprocessor. The constant defined using #define are macros that behaves like a constant. These constants are not handled by the compiler, they are handled by the preprocessor and are replaced by their value before complication.

Syntax of Constant in C using #define

```
#define const_name value
```

Example

- C

```c
// C Program to define a constant using #define
#include <stdio.h>
#define pi 3.14

int main()
{

    printf("The value of pi: %.2f", pi);
    return 0;
}
```

Output

```
The value of pi: 3.14
```

FAQs on C Constants

1. Define C Constants.

Constants in C are the immutable variables whose values cannot be modified once they are declared in the C program.

2. What is the use of the const keyword?

The const keyword is the qualifier that is used to declare the constant variable in C language.

3. Can we initialize the constant variable after the declaration?

No, we cannot initialize the constant variable once it is declared.

4. What is the right way to declare the constant in C?

The right way to declare a constant in C is to always initialize the constant variable when we declare.

5. What is the difference between constant defined using const qualifier and #define?

The following table list the differences between the constants defined using const qualifier and #define in C:

Constants using const	**Constants using #define**
They are the variables that are immutable	They are the macros that are replaced by their value.
They are handled by the compiler.	They are handled by the preprocessor.
Syntax: **const *type name = value*;**	Syntax: **#define *name value***

6. Is there any way to change the value of a constant variable in C?

Yes, we can take advantage of the loophole created by pointers to change the value of a variable declared as a constant in C. The below C program demonstrates how to do it.

- C

```c
// C Program to change the value of constant variable
#include <stdio.h>

int main()
{

    // defining an integer constant
    const int var = 10;

    printf("Initial Value of Constant: %d\n", var);

    // defining a pointer to that const variable
    int* ptr = &var;
    // changing value
    *ptr = 500;
    printf("Final Value of Constant: %d", var);
    return 0;
}
```

Output

```
Initial Value of Constant: 10
Final Value of Constant: 500
```