Enumeration in C++

Last Updated : 7 Oct, 2025

In C++, **enumeration** is a user-defined data type that consists of a set of named **integer** constants.

- It helps in assigning meaningful names to integer values to improve code readability and maintainability.
- It is mainly useful when we have a small set of possible values for an item like directions, days of week, etc.

```
#include <iostream>
using namespace std;

// Defining enum
enum direction
{
    EAST,
    NORTH,
    WEST,
    SOUTH
};

int main()
{

    // Creating enum variable
    direction dir = NORTH;

    cout << dir;
    return 0;
}
```

Output

```
1
```

Create Enum

An **enum** needs to be defined before we can create its variables.

```
enum enum_name {

    name1, name2, name3, ...

};
```

where **name1, name2, ….** are the names for the constant. They should be unique identifiers. By default, the first name in an enum is assigned the integer value 0, and the subsequent ones are incremented by 1.

After that, we can create a variable of this **enum** and assign it some name constant that was defined in it.

```
enum_name var;
```

Changing Values of Named Constant

We can manually assign values to enum members if needed.

```
enum enum_name {
    name1 = val1, name2 = val2, name3, ...
};
```

val1, val2 … values should be integer. It is also not compulsory to define the value of all constant names. If the current constant name value is x, then subsequent values will keep incrementing by one.

```
#include <iostream>
using namespace std;

// Defining enum
enum fruit
{
    APPLE,
    BANANA = 5,
    ORANGE
};

int main()
{

    // Creating enum variable
    fruit f = BANANA;
    cout << f << endl;

    // Changing the value
    f = ORANGE;
    cout << f;
    return 0;
}
```

Output

```
5

6
```

Enum Classes

C++11 introduced enum class, which provides better type safety. It helps in resolving name conflicts by providing scope to the constant names. It also requires manual typecasting to integer values from names.

Create enum Class

Enum class can be created just by adding the **class** keyword in the enum delcaration.

```
#include <iostream>
using namespace std;

// Define the enum class
enum class Day
{
    Sunday = 1,
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday
};

int main()
{

    // initializing
    Day today = Day::Thursday;

    // Print the enum
    cout << static_cast<int>(today);
    return 0;
}
```

Output

```
5
```

If we try to just assign the name of the constant to the enum variable, compiler gives an error.

```
#include <iostream>
using namespace std;

// Define the enum class
enum class Day
{
    Sunday = 1,
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday
};

int main()
{

    // initializing
    Day today = Thursday;

    // Print the enum
    cout << static_cast<int>(today);

    return 0;
}
```

Output

```
main.cpp: In function ‘int main()’:

main.cpp:12:17: error: ‘Thursday’ was not declared in this scope; did
you mean ‘Day::Thursday’?

   12 |     Day today = Thursday;
```

```
      |                 ^~~~~~~
      |                 Day::Thursday
main.cpp:6:28: note: ‘Day::Thursday’ declared here
    6 |                 Wednesday, Thursday, Friday,
      |
```