Enum
=====

1. In C++, when you define an enum without specifying an underlying type explicitly, the enum is treated as an integer type by default.

2. This means that each enumerator in the enum gets assigned an integer value starting from 0 and incremented by 1 for subsequent enumerators.

3. Enumerators are replaced with their integer values during compilation, so they don't consume additional memory.

==

```
#include <iostream>
enum Color
{
    red,blue,pink,green
    // 0 1 2 3 are assigned by default in
    // the sequential order to enum element
};
int main()
{
    Color c1=red;//red -> 0
    Color c2=green;//green-> 3
    Color c3=pink;//pink -> 2
   //The variable c1,c2,c3 is treated as an int by the compiler
   //because enums are essentially integer types with named values.
    std::cout << c1 << std::endl;//0
    std::cout << c2 << std::endl;//3
    std::cout << c3 << std::endl;//2

    return 0;
}
```

==

```
#include <iostream>
enum Color
{
    red,blue,pink,green
    // 0 1 2 3 are assigned by default in
    // the sequential order to enum element
};
```

```
int main()
{
   for(int i=red;i<=green;i++)
   {
       std::cout << i << std::endl;//0 1 2 3
   }
    return 0;
}
```
===
```
#include<iostream>
#include<vector>
using namespace std;
// Bring all elements of std namespace into the current scope
enum Section
{
    START,MID,END
};
int main()
{
    vector<Section> v;
    //v is of type Section, which is of enum type
    v.push_back(Section::START);//0
    v.push_back(Section::MID);//1
    v.push_back(Section::END);//2
    for(auto &e:v)
    {
        cout <<e<<endl;// 0 1 2
    }
    cout<<Section::START<<endl;//0
    for (const auto &e : v)
    {
        switch (e)
        {
            case Section::START:
                cout << "START section" << endl; // cout without std::
                break;
            case Section::MID:
                cout << "MID section" << endl; // cout without std::
                Break;
            case Section::END:
                cout << "END section" << endl; // cout without std::
                break;
        }
    }
    return 0;
}
```
Output:

```
======
0
1
2
0
START section
MID section
END section
```
==