Union in C++

A union is a type of structure that can be used where the amount of memory used is a key factor.

- Similarly to the structure, the union can contain different types of data types.
- Each time a new variable is initialized from the union it overwrites the previous in C language but in C++ we also don't need this keyword and uses that memory location.
- This is most useful when the type of data being passed through functions is unknown, using a union which contains all possible data types can remedy this problem.
- It is declared by using the keyword "**union**".

Below is the C++ program illustrating the implementation of union:

```
// C++ program to illustrate the use
// of the unions
#include <iostream>
using namespace std;

// Defining a Union
union GFG {
    int Geek1;
    char Geek2;
    float Geek3;
};

// Driver Code
int main()
{
    // Initializing Union
    union GFG G1;

    G1.Geek1 = 34;

    // Printing values
    cout << "The first value at "
        << "the allocated memory : " << G1.Geek1 << endl;
    G1.Geek2 = 34;

    cout << "The next value stored "
        << "after removing the "
        << "previous value : " << G1.Geek2 << endl;

    G1.Geek3 = 34.34;

    cout << "The Final value value "
        << "at the same allocated "
        << "memory space : " << G1.Geek3 << endl;
    return 0;
}
```

Output

```
 The first value at the allocated memory : 34
The next value stored after removing the previous value : "
The Final value value at the same allocated memory space : 34.34
```

Explanation: In the above code, Geek2 variable is assigned an integer (34). But by being of char type, the value is transformed through coercion into its char equivalent (“). This result is correctly displayed in the Output section.