User-Defined Data Type

Different types of elements are there in user-defined data type. To support the definition of a full-fledged data type, all C++ had to do was extend the syntax of struct by permitting you to include functions and operators as members of a struct. C++ also made the structure tag or name-the symbol following the struct-a stand alone name, meaning that you can use that name without the struct prefix. Lets see the different kinds of user-define data type:

Structures and Classes

User defined data types like struct and union have been used by programmers in C. While these data types are valid in C++, a few more additional features are added to make them suitable for Object-Oriented Programming. In C, structures were usually used to hold only data, but in C++ structures can hold both data and functions. C++ also has another data type known as class which can contain both data and functions and can also be used to declare variables and those class variables are called as objects.

Enumerated Data Type

Another approach to defining your own data type is the enumerated data type which is similar to a structure or union. Its members are constants that are written as identifiers, though they have a signed value. These constants represent values that can be assigned to corresponding enumeration variables. The enum keyword automatically enumerates a list of words by assigning them values 0,1,2, and so on. This facility provides another means for creating symbolic constants. The syntax of an enum statement is:

```
enum typename {member1, member2, ....., member n};
```

where enum is a keyword; typename is a name that identifies the type being defined, and member1, member2, ..., member n represents the individual identifiers that may be assigned to variables of this type. The member names must differ from one another, and they must be distinct from other identifiers whose scope is the same as that of the enumeration.