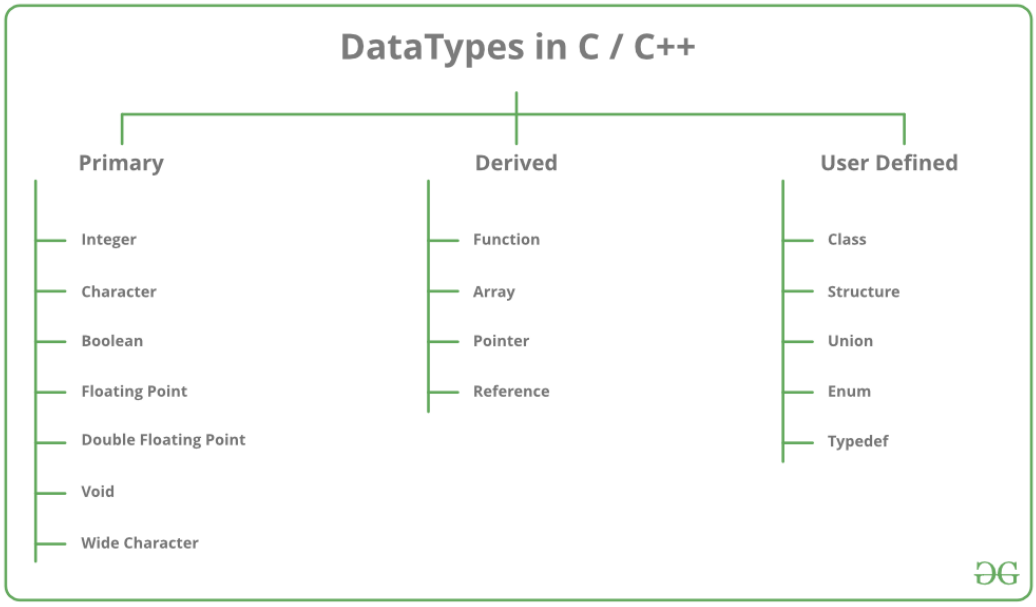
C++ Object as Data types

All variables use data-type during declaration to restrict the type of data to be stored. Therefore, we can say that data types are used to tell the variables the type of data it can store. Whenever a variable is defined in C++, the compiler allocates some memory for that variable based on the data type with which it is declared. Every data type requires a different amount of memory.

C++ supports a wide variety of data types and the programmer can select the data type appropriate to the needs of the application. Data types specify the size and types of value to be stored. However, storage representation and machine instructions to manipulate each data type differ from machine to machine, although C++ instructions are identical on all machines.

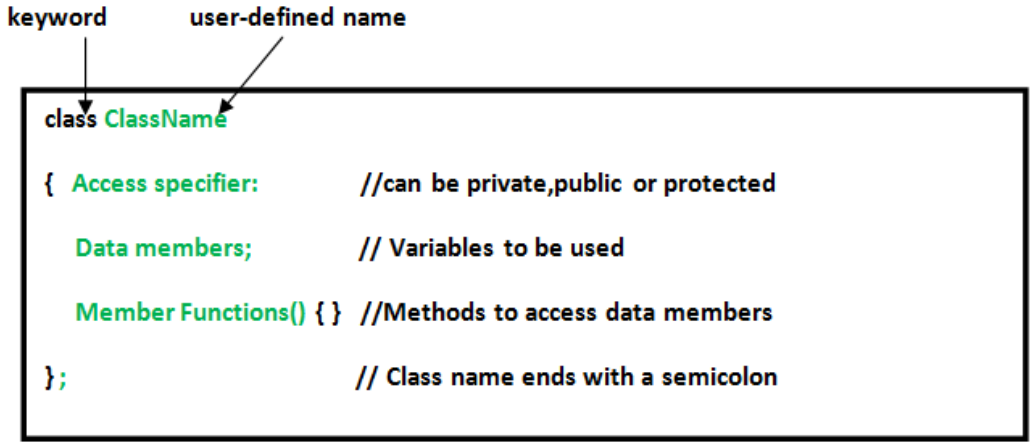
C++ supports the following data types:

1. Primary or Built in or Fundamental data type
2. Derived data types
3. User defined data types



Defining Class and Declaring Objects

A class is defined in C++ using keyword class followed by the name of class. The body of class is defined inside the curly brackets and terminated by a semicolon at the end.




```
// C++ program to demonstrate
#include <bits/stdc++.h>
using namespace std;
class Geeks
{
   // Access specifier
   public:

   // Data Members
   string geekname;

   // Member Functions()
   void printname()
   {
      cout << "Geekname is: " << geekname;
   }
};

int main() {

   // Declare an object of class geeks
   Geeks obj1;

   // accessing data member
   obj1.geekname = "Abhi";

   // accessing member function
   obj1.printname();
   return 0;
}
```

Output:

Geekname is: Abhi