Class Constructors

When defining a class, the programmer may choose to write special member functions, called **constructors**. Constructors are called automatically when an instance of a class is declared, so they are used to initialize some or all class data fields. The following rules apply to constructors

1) They have the same name as the class.
2) They do not have return value declarations, not even void.
3) If a programmer does not define any constructors, then the compiler generates one automatically. When such a constructor is called, the data fields of the class instance created are uninitialized.
4) A class may have several constructors. The constructor that has no parameters is called the default constructor. If a programmer defines any constructors, then they would be wise to also define a default constructor.
5) You can't call constructors the way you call a normal function. They are invoked automatically under the following circumstances:
   a) When class instance is declared. For example, "String S;", where "String" is a class.
   b) When an instance of a class is passed by value to a function.
   c) When an instance of a class is returned as a function value.
6) In order to declare an array of class instances, one of the following must be true:

   No constructors have been explicitly defined by the programmer
   OR
   A default constructor has been explicitly defined by the programmer.

Example:

```
class Fraction
{
  long Numer;     //holds numerator of fraction
  long Denom;     //holds denominator

public:

Fraction(long Numerator,     // constructor that does initialization
         long Denominator)
{
  Numer = Numerator;
  Denom = Denominator;
}

Fraction()      // the default constructor
{
  Numer = 0;
  Denom = 1;
}
// Rest of class definition
};
```

There is more about constructors, which will come with dynamically created data.