Lecture 7.1

Topics
1. C++ Class – Continued
2. Object Initialization – Constructor

____

1. C++ `Class` – Continued

Recall that

Class is an abstraction used to define a new data **type** through the **`class`** keyword. The general syntax of a **`class`** declaration is as follows,

```
class ClassName {
private:
  //private functions and data members of class
protected:
  //protected functions and data members
public:
  //public functions and data members of class
};
```

Or,

```
class ClassName {
public:
  //public functions and data members of class
protected:
  //protected functions and data members
private:
  //private functions and data members of class };
```

Another possible order of a class is as follows,

```
class ClassName {
  //private functions and data members of class
public:
  //public functions and data members of class
protected:
  //protected functions and data members
};
```

Example 1

```
/**
 *Program Name:   cis25L0711.cpp
 *Discussion:     Class with and without
 *                Data Encapsulation
 */

#include <iostream>
using namespace std;

class PrivateFraction {
private:
  int iNum;
  int iDenom;
};

class PublicFraction {
public:
  int iNum;
  int iDenom;
};

int main( void ) {
  int iX;

  Fraction prfrA; //What can we do with this
                  //frA object?

  int iY;

  PublicFraction pufrB; //What can we do with this
                        //pufrB object?

  int iZ;

  return 0;
}
```

The above example has two fraction classes with data members being placed in different groups – **`private`** and **`public`**. The class with only data and placed in **`private`** group will have its objects unusable; while the objects with all **`public`** data will allow its data to be modified directly and this is not recommended.

An object should have its data encapsulated that means no **direct** access to data to be permitted. Data should only be accessed (used or modified) through operations or function calls.

Every object should be created (constructed) with proper (initial) data values. This process is performed through the use of a special function called constructor. Let's look at constructor next.

2. Object Initialization -- Constructors

Whenever an object is created it should be initialized. In general, the initialization will assign private data with some initial values depending on how the object being created. This initialization is performed through a **`public`** function member called **constructor**.

One of the constructors is called whenever an object is created. There may be one, two, or more constructors for any class.

In general, a **class constructor** must

- have the same name as that of class in which it is a member, and
- not have return type (and value) but may have its own arguments.

Example 2

```
/**
 *Program Name:   cis25L0712.cpp
 *Discussion:       Class with CONSTRUCTOR for
 *                        Data Initialization
 */

#include <iostream>
using namespace std;
```

```
class Fraction {
public:
  Fraction();
private:
  int iNum;
  int iDenom;
};

Fraction::Fraction() {
  iNum = 0;
  iDenom = 1;
}

class PublicFraction {
public:
  PublicFraction();
public:
  int iNum;
  int iDenom;
};

PublicFraction::PublicFraction() {
  iNum = 0;
  iDenom = 1;
}

int main( void ) {
  int iX;

  Fraction frA; //What can we do with this
                //prfrA object?

  int iY;

  PublicFraction pufrB; //What can we do with this
                        //pufrB object?

  int iZ;

  return 0;
}
```

Let’s discuss the above example in class.