Chapter 3
Introduction to
Classes and Objects

OBJECTIVES

- What classes(类), objects(对象), member functions(成员函数) and data members(数据成员) are.
- How to define a class and use it to create an object.
- How to define member functions in a class to implement the class's behaviors(行为).
- How to declare data members in a class to implement the class's attributes(属性).
- How to call a member function of an object to make that member function perform its task.

OBJECTIVES

- The differences between data members of a class and local variables(局部变量) of a function.
- How to use a constructor(构造函数) to ensure that an object's data is initialized when the object is created.
- How to engineer a class to separate its interface(接口) from its implementation and encourage reuse.

Topics

- **3.1 Introduction**
- 3.2 Defining a Class with a Member Function(eg1)
- 3.3 Defining a Member Function with a Parameter(eg2)
- 3.4 Data Members, set Functions and get Functions(eg3)
- 3.5 Initializing Objects with Constructors(eg4)
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- 3.7 Separating Interface from Implementation(eg6)
- 3.8 Validating Data with set Functions(eg7)

3.1 Introduction

- 3.1.1 Concepts of Class and Object
- 3.1.2 七个例子简介

3.1.1 Concepts of Class and Object

- **Class(类):** 面向对象程序设计将数据(属性)和函数(行为)封装到称为类的软件包中, 类是实现数据封装(encapsulation)和抽象(abstraction)的工具.
- **Object(对象):** 客观世界中各种各样的实体, 它是类的实例。

3.1.1 Concepts of Class and Object

类的设计：

抽象

封装

class Car

数据成员(属性)：

颜色、排量、载荷、

速度、油量……

成员函数(行为)：

踩油门、刹车、转弯……

3.1.1 Concepts of Class and Object

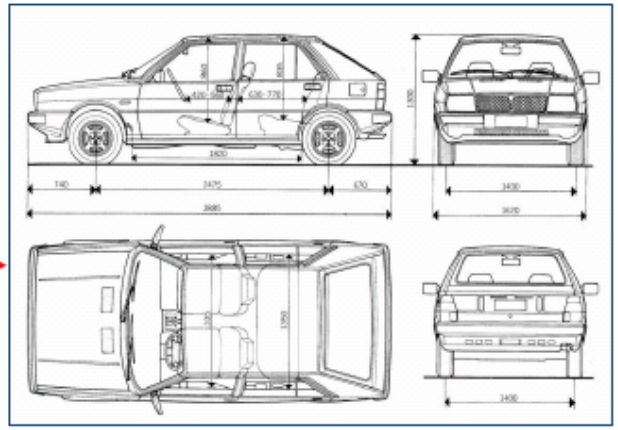
- 类的应用：对象的生成

class Car

数据成员：颜色等

成员函数：刹车等

相似



实例化，即
创建类的对象

生产

Car redCar

数据成员(属性)：
颜色 = 红色

……

Car blueCar

数据成员(属性)：
颜色 = 蓝色

3.1.1 Concepts of Class and Object

- 类的应用：对象的使用

Car theCar
数据成员： 颜色 = 红色 速度　　油量　　……
成员函数： 踩油门(加速) 刹车　　转弯　　……

① 设置对象属性，即数据成员赋值

② 发送消息给对象，即调用成员函数，以完成特定任务

3.1.2 七个例子简介

- 例子 **in Ch2**
- 基本的输入/输出；算术运算；基本判断语句. 除了**cin/cout**的使用外，不涉及类和对象的构建，所有任务都在**main**函数中完成

3.1.2 七个例子简介

❑例子 in Ch3

GradeBook

例子 3.1	拥有一个成员函数的类
例子 3.2	拥有一个带有参数的成员函数的类
例子 3.3	拥有数据成员+set/get成员函数的类
例子 3.4	拥有构造函数的类
例子 3.5	类的重用，将类置于单独的文件
例子 3.6	分离类的接口和实现
例子 3.7	增强功能-输入数据的有效性验证

Topics

- 3.1 Introduction
- **3.2 Defining a Class with a Member Function(eg1)**
- 3.3 Defining a Member Function with a Parameter(eg2)
- 3.4 Data Members, set Functions and get Functions(eg3)
- 3.5 Initializing Objects with Constructors(eg4)
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- 3.7 Separating Interface from Implementation(eg6)
- 3.8 Validating Data with set Functions(eg7)

3.4 Defining a Class With a Member Function (eg1)

- 如何定义带有一个成员函数的类?
- 如何创建对象?
- 如何调用对象的函数?

3.4 Defining a Class With a Member Function (eg1)

```
1  // Fig. 3.1: fig03_01.cpp
2  // Define class GradeBook with a member function displayMessage;
3  // Create a GradeBook object and call its displayMessage function.
4  #include <iostream>
5  using std::cout;
6  using std::endl;
7
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // function that displays a welcome message to the GradeBook user
13    void displayMessage()
14    {
15       cout << "Welcome to the Grade Book!" << endl;
16    } // end function displayMessage
17 }; // end class GradeBook
18
19 // function main begins program execution
20 int main()
21 {
22    GradeBook myGradeBook; //create a GradeBook object named myGradeBook
23    myGradeBook.displayMessage(); // call object's displayMessage function
24    return 0; // indicate successful termination
25 } // end main
```

① 文件包含、using声明

② 类定义

③ main函数, 类的使用

3.4 Defining a Class With a Member Function (eg1)

② 类定义

```
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // function that displays a welcome message to the GradeBook user
13    void displayMessage()
14    {
15       cout << "Welcome to the Grade Book!" << endl;
16    } // end function displayMessage
17 }; // end class GradeBook
```

Line 9: class GradeBook

- 定义类的关键词为class, 后跟用户自定义类的类名

GradeBook—用户定义类型(vs 基本数据类型)

- 类名命名规范: camel case, 自定义类的类名由若干名词组成, 每个名词首字母大写(G, B)

3.4 Defining a Class With a Member Function (eg1)

② 类定义

```
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // function that displays a welcome message to the GradeBook user
13    void displayMessage()
14    {
15       cout << "Welcome to the Grade Book!" << endl;
16    } // end function displayMessage
17 }; // end class GradeBook
```

Line 10-17:

- class' body, 类体
- 以左大括号 { 开始，以右大括号 } 和 分号 ; 结束

3.4 Defining a Class With a Member Function (eg1)

② 类定义

```
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // function that displays a welcome message to the GradeBook user
13    void displayMessage()
14    {
15       cout << "Welcome to the Grade Book!" << endl;
16    } // end function displayMessage
17 }; // end class GradeBook
```

Line 11: public:

- Access-Specifier Label, 访问权限指示符标签, 表示从该指示符标签至下一指示符标签或类定义结束括号之间的所有成员函数和数据成员均为此访问权限类型
- 其中public关键词, 为访问权限指示符, 表示可公共使用: public(公有), private(私有), protected(保护)

3.4 Defining a Class With a Member Function (eg1)

② 类定义

```
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // function that displays a welcome message to the GradeBook user
13    void displayMessage()
14    {
15       cout << "Welcome to the Grade Book!" << endl;
16    } // end function displayMessage
17 }; // end class GradeBook
```

Function header, 函数头

Line 13-16: 成员函数定义

- **void**, 表示无任何数据返回
- **displayMessage**: 函数名, 首字母小写, 后续单词首字母大写
- 小括号对用于指定输入参数, 空括号()表示无输入参数

3.4 Defining a Class With a Member Function (eg1)

```
19  // function main begins program execution
20  int main()
21  {
22     GradeBook myGradeBook; //create a GradeBook object named myGradeBook
23     myGradeBook.displayMessage(); // call object's displayMessage function
24     return 0; // indicate successful termination
25  } // end main
```

③ main函数, 类的使用

Line 20-25: main函数定义

- 目标: 通过调用GradeBook类的displayMessage函数输出文字“Welcome to the Grade Book! ”
- 22行,与基本数据类型变量的定义类似, 定义GradeBook类变量myGradeBook, 即生成了一个GradeBook类的对象
- 23行, 对象变量+点操作符dot operator .+成员函数名

```
1  // Fig. 3.1: fig03_01.cpp
2  // Define class GradeBook with a member function displayMessage;
3  // Create a GradeBook object and call its displayMessage function.
4  #include <iostream>
5  using std::cout;
6  using std::endl;
7
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // function that displays a welcome message to the GradeBook user
13    void displayMessage()
14    {
15       cout << "Welcome to the Grade Book!" << endl;
16    } // end function displayMessage
17 }; // end class GradeBook
18
19 // function main begins program execution
20 int main()
21 {
22    GradeBook myGradeBook; //create a GradeBook object named myGradeBook
23    myGradeBook.displayMessage(); // call object's displayMessage function
24    return 0; // indicate successful termination
25 } // end main
```

① 文件包含、using声明

② 类定义

③ main函数, 类的使用

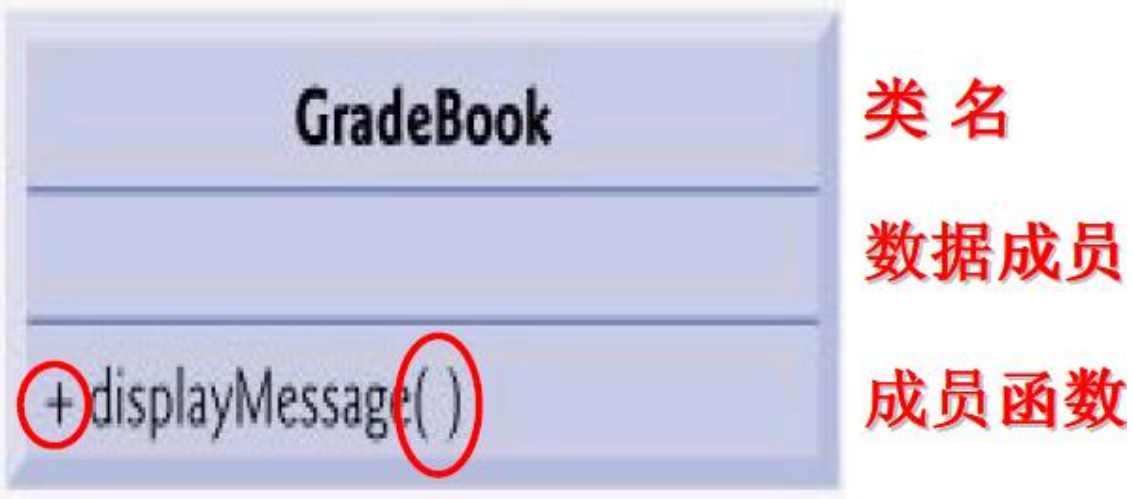
```
Welcome to the Grade Book!
```

3.4 Defining a Class With a Member Function (eg1)

- 基本步骤:
- 通过关键词class, 定义类GradeBook
- 生成GradeBook类的对象myGradeBook
- 通过点操作符, 调用myGradeBook对象的成员函数displayMessage()

3.4 Defining a Class With a Member Function (eg1)

- **GradeBook类的UML类图**



Topics

- 3.1 Introduction
- 3.2 Defining a Class with a Member Function(eg1)
- **3.3 Defining a Member Function with a Parameter(eg2)**
- 3.4 Data Members, set Functions and get Functions(eg3)
- 3.5 Initializing Objects with Constructors(eg4)
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- 3.7 Separating Interface from Implementation(eg6)
- 3.8 Validating Data with set Functions(eg7)

3.3 Defining a Member Function with a Parameter(eg2)

- 调用类的成员函数时, 如何传递额外的数据?
- 函数参数: **Parameter +Argument**

① 文件包含、using声明

```
#include <iostream>
using std::cout;
using std::cin;
using std::endl;

#include <string> // program uses C++ standard string class
using std::string;
using std::getline;
```

② 类定义

```
// GradeBook class definition
class GradeBook
{
public:
   // function that displays a welcome message to the GradeBook user
   void displayMessage( string courseName )
   {
      cout << "Welcome to the grade book for\n" << courseName << "!"
         << endl;
   } // end function displayMessage
}; // end class GradeBook
```

③ main函数，类的使用

```
// function main begins program execution
int main()
{
   string nameOfCourse; // string of characters to store the course name
   GradeBook myGradeBook; // create a GradeBook object named myGradeBook

   // prompt for and input course name
   cout << "Please enter the course name:" << endl;
   getline( cin, nameOfCourse ); // read a course name with blanks
   cout << endl; // output a blank line

   // call myGradeBook's displayMessage function
   // and pass nameOfCourse as an argument
   myGradeBook.displayMessage( nameOfCourse );
   return 0; // indicate successful termination
} // end main
```

P69 Fig3.3

3.3 Defining a Member Function with a Parameter(eg2)

```
4  #include <iostream>
5  using std::cout;
6  using std::cin;
7  using std::endl;
8
9  #include <string> // program uses C++ standard string class
10 using std::string;
11 using std::getline;
```

① 文件包含、using声明

Line 9-11: 使用C++标准字符串类

- 第9行, 包含include字符串类头文件
- 第10/11行, using声明, 使用string类和getline函数

3.3 Defining a Member Function with a Parameter(eg2)

```
14  class GradeBook
15  {
16  public:
17     // function that displays a welcome message to the GradeBook user
18     void displayMessage( string courseName )
19     {
20        cout << "Welcome to the grade book for\n" << courseName << "!"
21           << endl;
22     } // end function displayMessage
23  }; // end class GradeBook
```

② 类定义

第18行: ()内的为参数列表(parameter list)

- (), 0个参数
- (string courseName), 1个参数
- (string courseName, int a, int b), 多个参数

逗号,分割，每个参数都必须指定类型

3.3 Defining a Member Function with a Parameter(eg2)

定义了string类型变量

类的使用

```
26  int main()
27  {
28     string nameOfCourse; // string of characters to store the course name
29     GradeBook myGradeBook; // create a GradeBook object named myGradeBook
30
31     // prompt for and input course name
32     cout << "Please enter the course name:" << endl;
33     getline( cin, nameOfCourse ); // read a course name with blanks
34     cout << endl; // ou
35
36     // call myGradeBook's displayMessage function
37     // and pass nameOfCourse as an argument
38     myGradeBook.displayMessage( nameOfCourse );
39     return 0; // indicate successful termination
40  } // end main
```

替换为cin >> nameOfCourse;如何？

P69 Fig3.3

- 第33行: getline(cin, nameOfCourse);
  - 从标准输入读入一行字符串

3.3 Defining a Member Function with a Parameter(eg2)

```
cin >> nameOfCourse;
```

vs

```
getline( cin, nameOfCourse );
```

- **cin**: 读取字符直至**first white-space character**(第一个空白字符)出现
- **getline**: 读取字符直至**newline character**(换行符)出现, 将换行符之前(丢弃换行符)的所有字符(包括空格等空白字符)赋值给字符串变量

3.3 Defining a Member Function with a Parameter(eg2)

③ main函数，类的使用

```
26  int main()
27  {
28     string nameOfCourse; // string of characters to store the course name
29     GradeBook myGradeBook; // create a GradeBook object named myGradeBook
30
31     // prompt for and input course name
32     cout << "Please enter the course name:" << endl;
33     getline( cin, nameOfCourse ); // read a course name with blanks
34     cout << endl; // output a blank line
35
36     // call myGradeBook's displayMessage function
37     // and pass nameOfCourse as an argument
38     myGradeBook.displayMessage( nameOfCourse );
39     return 0; // indicate successful termination
40  } // end main
```

P69 Fig3.3

- 第38行, 调用对象的成员函数, 传入参数(Argument)
- 输入参数的类型与函数定义时的一致 (type-cast)

3.3 Defining a Member Function with a Parameter(eg2)

- 18行: 函数头, Parameter

  void displayMessage(string courseName)

- 38行: 成员函数调用, Argument

  myGradeBook.displayMessage(nameOfCourse)

18行: parameter, 形式参数(形参), 函数接受输入数据的途径, 可以是基本类型、类等

38行: argument, 实际参数(实参), 用于初始化形参的值, 可以是变量、常量、表达式等

3.3 Defining a Member Function with a Parameter(eg2)

```
13 // GradeBook class definition
14 class GradeBook
15 {
16 public:
17    // function that displays a welcome message to the GradeBook user
18    void displayMessage( string courseName )
19    {
20       cout << "Welcome to the grade book for\n" << courseName << "!"
21          << endl;
22    } // end function displayMessage
23 }; // end class GradeBook
24
25 // function main begins program execution
26 int main()
27 {
28    string nameOfCourse; // string of characters to store the course name
29    GradeBook myGradeBook; // create a GradeBook object named myGradeBook
30
31    // prompt for and input course name
32    cout << "Please enter the course name:" << endl;
33    getline( cin, nameOfCourse ); // read a course name with blanks
34    cout << endl; // output a blank line
35
36    // call myGradeBook's displayMessage function
37    // and pass nameOfCourse as an argument
38    myGradeBook.displayMessage( nameOfCourse );
39    return 0; // indicate successful termination
40 } // end main
```

string courseName = nameOfCourse;

3.3 Defining a Member Function with a Parameter(eg2)

- GradeBook类的UML类图

Topics

- 3.1 Introduction
- 3.2 Defining a Class with a Member Function(eg1)
- 3.3 Defining a Member Function with a Parameter(eg2)
- **3.4 Data Members, set Functions and get Functions(eg3)**
- 3.5 Initializing Objects with Constructors(eg4)
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- 3.7 Separating Interface from Implementation(eg6)
- 3.8 Validating Data with set Functions(eg7)

3.4 Data Members, set Functions and get Functions(eg3)

- 数据成员的定义、赋值和读取

程序解读

```
14 // GradeBook class definition
15 class GradeBook
16 {
17 public:
18    // function that sets the course name
19    void setCourseName( string name )
20    {
21       courseName = name; // store the course name in the object
22    } // end function setCourseName
23
24    // function that gets the course name
25    string getCourseName()
26    {
27       return courseName; // return the object's courseName
28    } // end function getCourseName
29
30    // function that displays a welcome message
31    void displayMessage()
32    {
33       // this statement calls get
34       // name of the course this
35       cout << "Welcome to the grade book for\n" << getCourseName() << "!"
36          << endl;
37    } // end function displayMessage
38 private:
39    string courseName; // course name for this GradeBook
40 }; // end class GradeBook
```

给courseName赋值

返回courseName值

输出getCourseName()函数返回的字符串值

```
14 // GradeBook class definition
15 class GradeBook
16 {
17 public:
18    // function that sets the course name
19    void setCourseName( string name )
20    {
21       courseName = name; // store the course name in the object
22    } // end function setCourseName
23
24    // function that gets the course name
25    string getCourseName()
26    {
```

Line 38-39: private:

- 私有访问权限指示符, 表示私有访问属性, 即仅能被本对象的成员函数访问 -- displayMessage() ?
- 数据成员私有, 即data hiding(数据隐藏)
- 类成员函数和数据成员的缺省访问属性均为私有属性

```
37    } // end function displayMessage
38 private:
39    string courseName; // course name for this GradeBook
40 }; // end class GradeBook
```

3.4 Data Members, set Functions and get Functions(eg3)

- courseName为类GradeBook的私有字符串型的数据成员
- Data Member vs Local Variable
  - Data Members (数据成员): 在类定义体内、在类成员函数外声明的变量, 每个对象都有自己的一份属性数据
  - Local Variable(局部变量): 在函数(类成员函数/全局函数)定义体内声明, 必须在声明后才能使用,并且只能在函数体内使用. 函数执行结束, 局部变量值即丢失(static…ch6)

```
#include <iostream>

// GradeBook class definition
class GradeBook
{
   int nSpirit;
public:
   void displayMessage()
   {
      cout << "Welcome to the grade book \n" << endl;
   }
};


int main()
{
   GradeBook myGradeBook;

   myGradeBook.displayMessage();
   return 0;
}
```

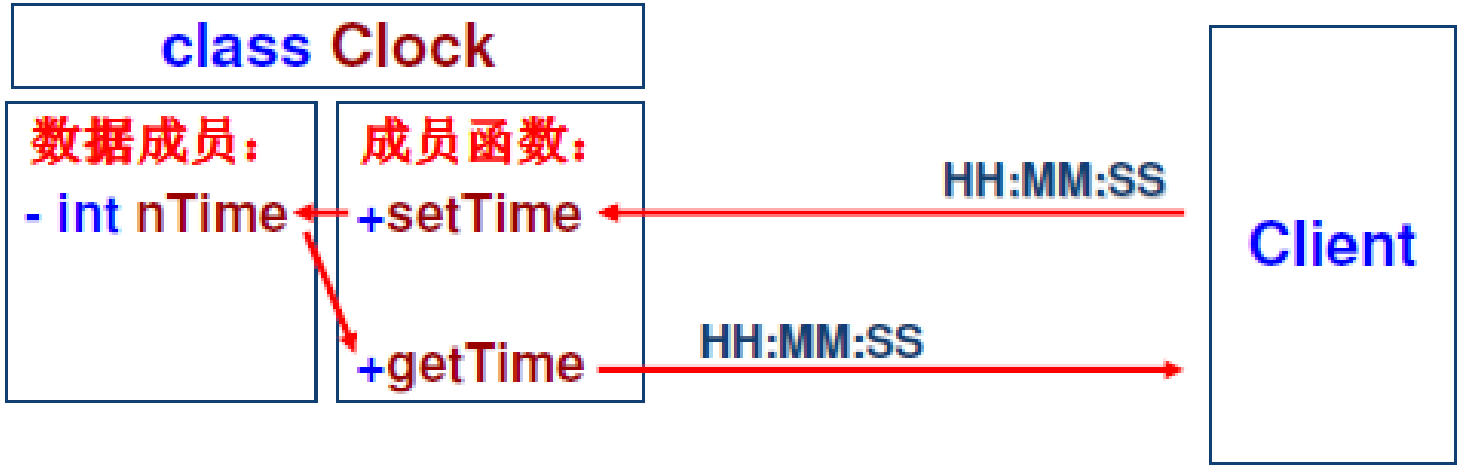
成员变量，私有

成员函数中的局部变量

全局函数中的局部变量

3.4 Data Members, set Functions and get Functions(eg3)

- 数据成员可以是任意访问权限,公有public、私有private、保护protected等
- 从软件工程角度考虑, 建议设置为私有属性,结合set/get函数实现对数据赋值/读取



```
14 // GradeBook class definition
15 class GradeBook
16 {
17 public:
18    // function that sets the course name
19    void setCourseName( string name )
20    {
21       courseName = name; // store the course name in the object
22    } // end function setCourseName
23
24    // function that gets the course name
25    string getCourseName()
26    {
27       return courseName; // return the object's courseName
28    } // end function getCourseName
29
30    // function that displays a welcome message
31    void displayMessage()
32    {
33       // this statement calls get
34       // name of the course this
35       cout << "Welcome to the grade book for\n" << getCourseName() << "!"
36          << endl;
37    } // end function displayMessage
38 private:
39    string courseName; // course name for this GradeBook
40 }; // end class GradeBook
```

给courseName赋值

返回courseName值

输出getCourseName()函数返回的字符串值

3.4 Data Members, set Functions and get Functions(eg3)

```
42 // function main begins progra
43 int main()
44 {
45    string nameOfCourse; // str
46    GradeBook myGradeBook; // c
47
48    // display initial value of courseName
49    cout << "Initial course name is: " << myGradeBook.getCourseName()
50       << endl;
```

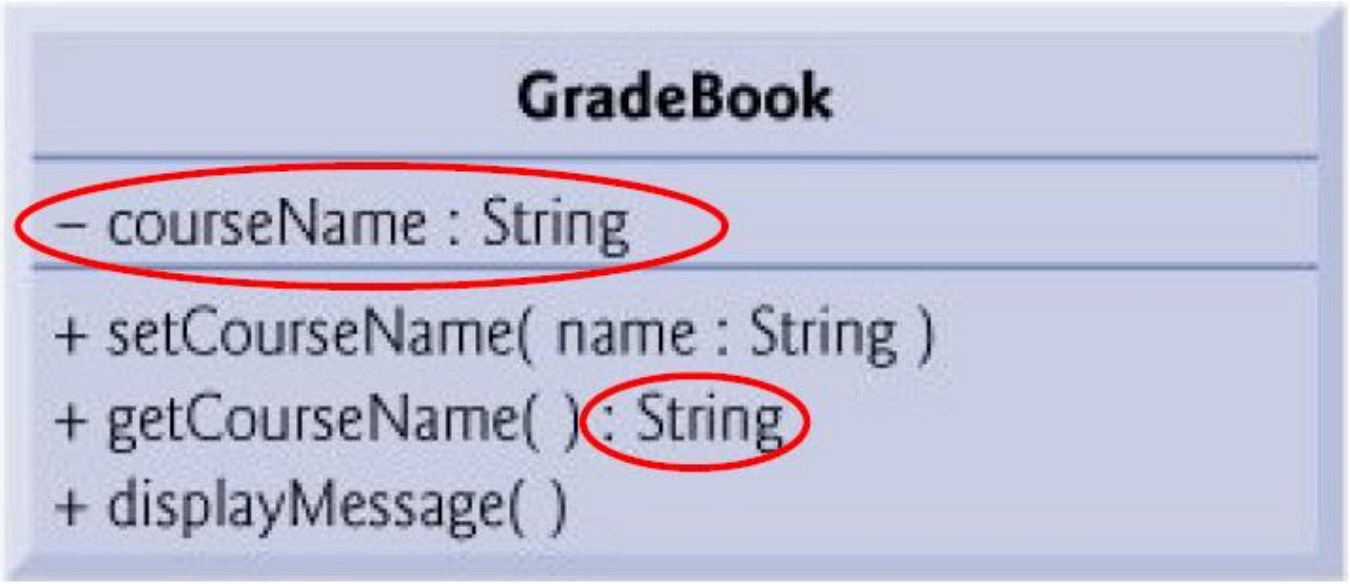
```
Initial course name is:

Please enter the course name:
CS101 Introduction to C++ Programming

Welcome to the grade book for
CS101 Introduction to C++ Programming!
```

Line 49:

- courseName未赋值时，①输出什么内容？
- 成员变量courseName的初始值，string类变量初始值为空字符串“”

```
58    myGradeBook.displayMessage(); // display message with new course name
59    return 0; // indicate successful termination
60 } // end main
```

3.4 Data Members, set Functions and get Functions(eg3)



Q & A

- 1. 简述Parameter和Argument概念.
- 2. 指出错误:

  string test(int num, string s1, s2){ …… }
- 3. 简述Local Variable和Member Variable概念.

Q & A

```
1.  class GradeBook
2.  {
3.      string classno;
4.   public:
5.      void displayMessage()
6.     {
7.          string str = "Welcome to the Grade Book for class ";
8.         cout << str << classno << endl;
9.      }
10.  };
11.  int main()
12. {
13.     GradeBook myGradeBook;
14.     myGradeBook.classno = "090113";
15.     myGradeBook.displayMessage();
16.     return 0;
17. }
```

Topics

- 3.1 Introduction
- 3.2 Defining a Class with a Member Function(eg1)
- 3.3 Defining a Member Function with a Parameter(eg2)
- 3.4 Data Members, set Functions and get Functions(eg3)
- **3.5 Initializing Objects with Constructors(eg4)**
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- 3.7 Separating Interface from Implementation(eg6)
- 3.8 Validating Data with set Functions(eg7)

3.5 Initializing Objects with Constructors(eg4)

- 如何在对象生成时初始化数据成员？
- **Constructor（构造函数）**

  Default Constructor（缺省构造函数）

程序解读

P77 Fig3.7

```
13    class GradeBook
14    {
15    public:
16       // constructor initializes courseName with string supplied as argument
17       GradeBook( string name )
18       {
19          setCourseName( name ); // call set function to initialize courseName
20       } // end GradeBook constructor
```

Line 17: GradeBook(string name)

- 构造函数(Constructor)是特殊的成员函数
  - 与类同名, 可有参数列表
  - 无返回类型 (包括void), 没有return语句
  - 通常声明为公有函数
  - 用于初始化类对象, 在对象定义时自动调用
  - 没有参数的构造函数称为缺省构造函数(Default Constructor)

3.5 Initializing Objects with Constructors(eg4)

- 当对象创建时，C++必须要构造函数来对对象进行正确初始化，构造函数会在对象创建时自动隐性地调用。

之前的例子均未定义构造函数，为何能正常工作？

3.5 Initializing Objects with Constructors(eg4)

- 当且仅当没有构造函数时, 编译器会提供default constructor(缺省构造函数)
  - 基本数据类型成员变量, 不进行初始化
  - 抽象数据类型成员变量, 则自动隐性调用该类的缺省构造函数

- 例子: 上一节string类型数据成员courseName自动初始化为空字符串

3.5 Initializing Objects with Constructors(eg4)

- **Conclusions:**
- 没有参数的构造函数都称为缺省构造函数
- 缺省构造函数来源
  - 来源1: 用户自己定义的没有参数的构造函数
  - 来源2: 用户没有定义任何构造函数时, 编译器自动提供

3.5 Initializing Objects with Constructors(eg4)

```
46  int main()
47  {
48     // create two GradeBook objects
49     GradeBook gradeBook1( "CS101 Introduction to C++ Programming" );
50     GradeBook gradeBook2( "CS102 Data Structures in C++" );
51
52     // display initial value of courseName for each GradeBook
53     cout << "gradeBook1 created for course: " << gradeBook1.getCourseName()
54        << "\ngradeBook2 created for course: " << gradeBook2.getCourseName()
55        << endl;
56     return 0; // indicate successful termination
57  } // end main
```

```
gradeBook1 created for course: CS101 Introduction to C++ Programming
gradeBook2 created for course: CS102 Data Structures in C++
```

3.5 Initializing Objects with Constructors(eg4)

- **GradeBook(string courseName){ …… }**

  使用非缺省(有参)构造函数定义对象

  - **GradeBook book(“C++ Course”);**

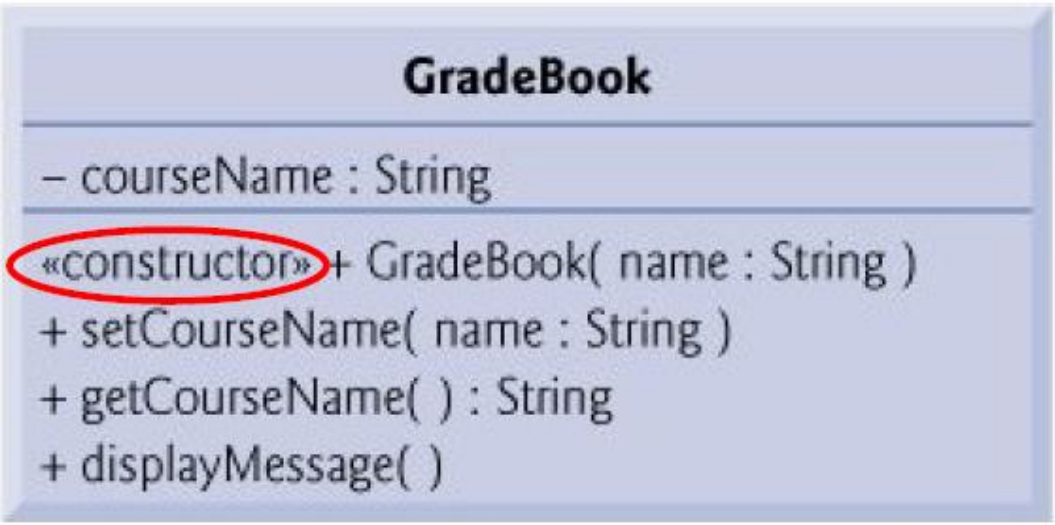
- **GradeBook(){ …… }**

  使用缺省(无参)构造函数定义对象

  - **GradeBook book;**
  - ~~**GradeBook book();**~~

二义性，函数声明 or 对象定义？

3.5 Initializing Objects with Constructors(eg4)



- 在构造函数名之前加上«constructor»
- 构造函数通常搁在类图第三栏的第一行

Topics

- 3.1 Introduction
- 3.2 Defining a Class with a Member Function(eg1)
- 3.3 Defining a Member Function with a Parameter(eg2)
- 3.4 Data Members, set Functions and get Functions(eg3)
- 3.5 Initializing Objects with Constructors(eg4)
- **3.6 Placing a Class in a Separate File for Reusability(eg5)**
- 3.7 Separating Interface from Implementation(eg6)
- 3.8 Validating Data with set Functions(eg7)

3.6 Placing a Class in a Separate File for Reusability(eg5)

- Source file：源文件
  - .cpp .cxx .cc .C
- 本节之前所有程序均为单独的.cpp文件(fig03_07.cpp)
- Header file：头文件– 被引用
  - 无后缀: 标准类库或者对C语言头文件的封装
  - .h: C头文件和用户定义头文件

```
#include <iostream>
```

3.6 Placing a Class in a Separate File for Reusability(eg5)

- Reusability(重用),如何重用GradeBook类?

```
// Fig. 3.10: fig03_10.cpp
#include <iostream>
using std::cout;
using std::endl;

// function main begins program execution
int main()
{
    cout << "I want to use class GradeBook!" << endl;

    // create two GradeBook objects
    GradeBook gradeBook1( "CS101 Introduction to C++ Programming" );
    GradeBook gradeBook2( "CS102 Data Structures in C++" );

    gradeBook1.displayMessage();
    gradeBook2.displayMessage();

    return 0; // indicate successful termination
} // end main
```

3.6 Placing a Class in a Separate File for Reusability(eg5)

- **Step 1:**

  error C2065: 'GradeBook' : undeclared identifier

- 编译器只知道基本数据类型，作为用户自定义类型的GradeBook，对编译器而言是未知的。
- 必须include定义GradeBook类的文件：
  - fig03_07.cpp → fig03_07.h
  - + #include .. fig03_07.h ..

3.6 Placing a Class in a Separate File for Reusability(eg5)

- include: “fig03_07.h” or <fig03_07.h> ?

- 搜索路径
  - 双引号：用户程序目录 → 标准库目录
  - 尖括号：标准库目录
- 结论
  - #include <标准库头文件>
  - #include “用户定义头文件”

- #include “fig03_07.h”

3.6 Placing a Class in a Separate File for Reusability(eg5)

- **Step 2:**

  error C2084: function 'int __cdecl main(void)' already has a body

- 一个程序只能有一个main函数
  - - main函数 from “fig03_07.h”

❖ fig03_07.h

- 单独的类文件
- 没有main函数，不能独立编译为可执行程序

```
2  // GradeBook cl
3  #include <iostr
4  using std::cout
5  using std::endl
6
7  #include <strin
8  using std::stri
9
10 // GradeBook class definition
11 class GradeBook
12 {
13 public:
14    // constructor initializes courseName with string supplied as argument
15    GradeBook( string name )
16    {
17       setCourseName( name ); // call set function to initialize courseName
18    } // end GradeBook constructor
19
20    // function to set the course name
21    void setCourseName( string name )
22    {
23       courseName = name; // store the course name in the object
24    } // end function setCourseName
25
26    // function to get the course name
27    string getCourseName()
28    {
29       return courseName; // return object's courseName
30    } // end function getCourseName
31
32    // display a welcome message to the GradeBook user
33    void displayMessage()
34    {
35       // call getCourseName to get the courseName
36       cout << "Welcome to the grade book for\n" << getCourseName()
37          << "!" << endl;
38    } // end function displayMessage
39 private:
40    string courseName; // course name for this GradeBook
41 }; // end class GradeBook
```

```cpp
// Fig. 3.10: fig03_10.cpp
#include <iostream>
using std::cout;
using std::endl;

// Step 2
#include "fig03_07.h"

// function main begins program execution
int main()
{
    cout << "I want to use class GradeBook!" << endl;

    // create two GradeBook objects
    GradeBook gradeBook1( "CS101 Introduction to C++ Programming" );
    GradeBook gradeBook2( "CS102 Data Structures in C++" );

    gradeBook1.displayMessage();
    gradeBook2.displayMessage();
```

❖ fig03_10.cpp

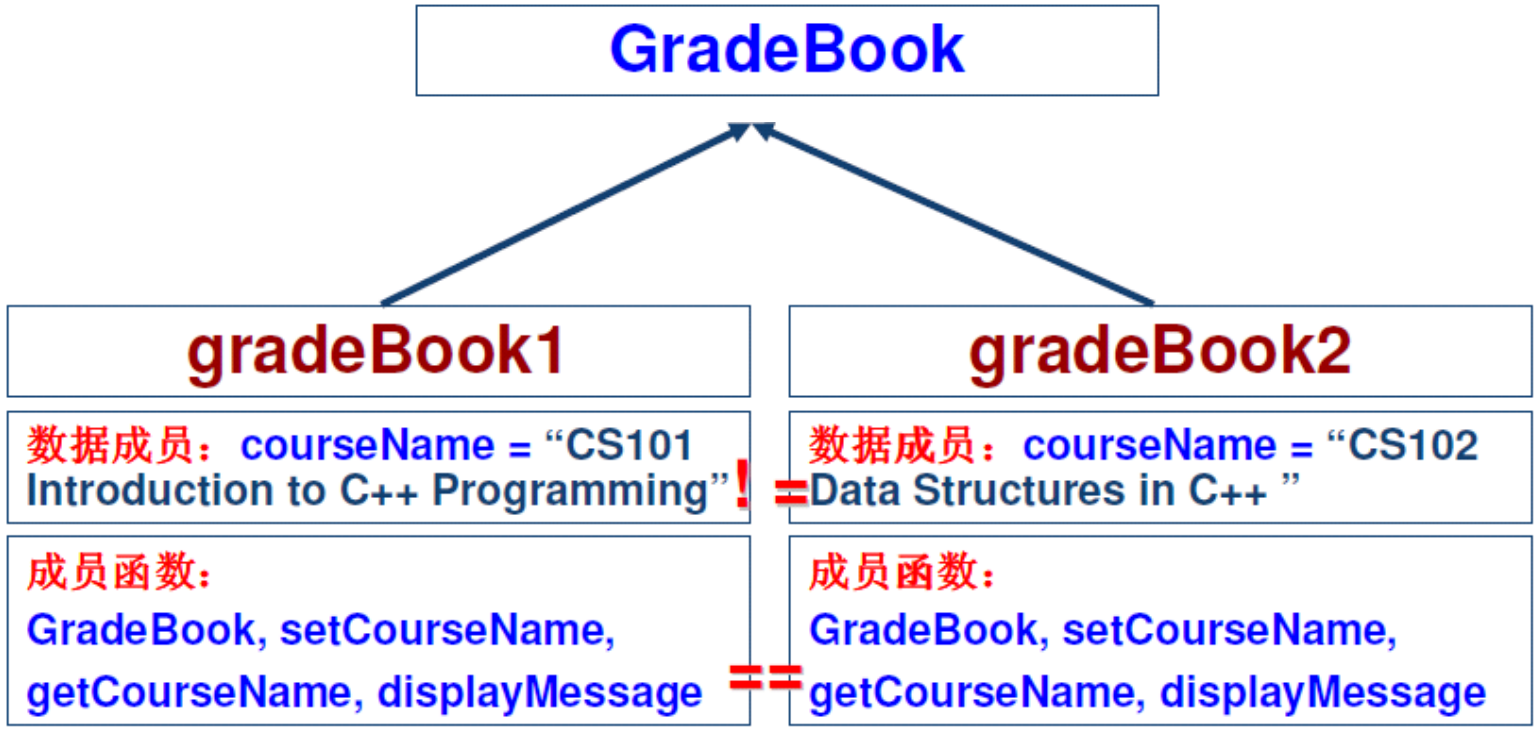
- 通过#include “fig03_07.h”, 使用 GradeBook 类
- Client Program, Driver Program(客户/驱动程序)

3.6 Placing a Class in a Separate File for Reusability(eg5)

- **Step 3:**

```
I want to use class GradeBook!
Welcome to the grade book for
CS101 Introduction to C++ Programming!
Welcome to the grade book for
CS102 Data Structures in C++!
Press any key to continue
```

3.6 Placing a Class in a Separate File for Reusability(eg5)



3.6 Placing a Class in a Separate File for Reusability(eg5)

- 结论: **Object** 对象的大小仅仅取决于数据成员所需要占用的内存数量, 而与成员函数无关!

Topics

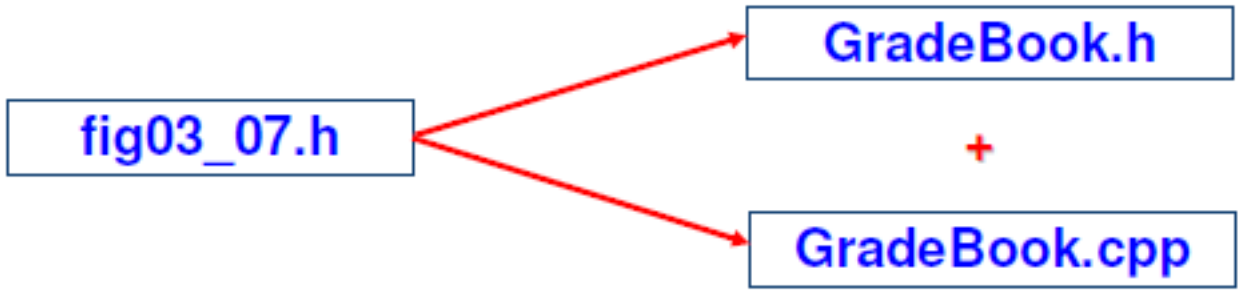
- 3.1 Introduction
- 3.2 Defining a Class with a Member Function(eg1)
- 3.3 Defining a Member Function with a Parameter(eg2)
- 3.4 Data Members, set Functions and get Functions(eg3)
- 3.5 Initializing Objects with Constructors(eg4)
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- **3.7 Separating Interface from Implementation(eg6)**
- 3.8 Validating Data with set Functions(eg7)

3.7 Separating Interface from Implementation(eg6)

- 上一节解决了GradeBook类的重用问题, 但是将所有实现代码的细节都暴露给了客户程序!
- 本节目标:
  - 保证类能重用
  - 类的客户程序知道如何调用类成员函数(哪些函数可用? 需要提供哪些参数? 返回类型是什么?)
  - 隐藏这些类成员函数的实现(知识产权, 代码更新)

3.7 Separating Interface from Implementation(eg6)

- **Interface(接口) of a Class：**
  - **What services a class's clients can use and how to request those services, but not how the class carries out the services.**
- **方法：在类定义外部给出成员函数代码**



3.7 Separating Interface from Implementation(eg6)

- （1） **GradeBook.h**
- （2）GradeBook.cpp

```
5.    #include <string> // class GradeBook uses C++ standard string class
6.    using std::string;
7.
8.    // GradeBook class definition
9.    class GradeBook
10.   {
11.   public:
12.     GradeBook( string ); // constructor that initializes courseName
13.     void setCourseName( string ); // function that sets the course name
14.     string getCourseName(); // function that gets the course name
15.     void displayMessage(); // function that displays a welcome message
16.   private:
17.     string courseName; // course name for this GradeBook
18.   }; // end class GradeBook
```

P83 Fig3.11

- **GradeBook.h:**

  Defining a class's Interface(接口) with Function Prototypes(函数原型) + Data Member

```
5.    #include <string> // class GradeBook uses C++ standard string class
6.    using std::string;
7.
8.    // GradeBook class definition
9.    class GradeBook
10.   {
11.   public:
12.      GradeBook( string ); // constructor that initializes courseName
13.      void setCourseName( string ); // function that sets the course name
14.      string getCourseName(); // function that gets the course name
15.      void displayMessage(); // function that displays a welcome message
16.   private:
17.      string courseName; // course name for this GradeBook
18.   }; // end class GradeBook
```

P83 Fig3.11

- 第5-6行
  - 仅保留了所需要的头文件和using声明
  - - <iostream>

```
5.  #include <string> // class GradeBook uses C++ standard string class
6.  using std::string;
7.
8.  // GradeBook class definition
9.  class GradeBook
10. {
11. public:
12.    GradeBook( string ); // constructor that initializes courseName
13.    void setCourseName( string ); // function that sets the course name
14.    string getCourseName(); // function that gets the course name
15.    void displayMessage(); // function that displays a welcome message
16. private:
17.    string courseName; // course name for this GradeBook
18. }; // end class GradeBook
```

P83 Fig3.11

- 第12-15行
  - **Function prototype (函数原型)**：关于函数的函数名、返回数据类型、参数类型的声明
  - 即函数头（参数名是否保留可选）+ 分号；

3.7 Separating Interface from Implementation(eg6)

```
GradeBook( string name )
{
   setCourseName( name );
}

void setCourseName( string name )
{
   courseName = name;
}
```

```
GradeBook( string name )

void setCourseName( string name )
```

保留函数头

加分号

函数原型

```
GradeBook( string name );

void setCourseName( string name );
```

```
GradeBook( string );

void setCourseName( string );
```

去参数名（可选）

```
5.  #include <string> // class GradeBook uses C++ standard string class
6.  using std::string;
7.
8.  // GradeBook class definition
9.  class GradeBook
10. {
11. public:
12.    GradeBook( string ); // constructor that initializes courseName
13.    void setCourseName( string ); // function that sets the course name
14.    string getCourseName(); // function that gets the course name
15.    void displayMessage(); // function that displays a welcome message
16. private:
17.    string courseName; // course name for this GradeBook
18. }; // end class GradeBook
```

P83 Fig 3.11

- 第17行
  - 保留数据成员(Data Member)
  - 编译时，告诉编译器类对象所需要的内存容量

3.7 Separating Interface from Implementation(eg6)

- 总结:
- 将函数定义替换为函数原型
- 删掉不需要的头文件包含声明和using声明（可选）
- 得到类接口(Interface)文件

3.7 Separating Interface from Implementation(eg6)

- （1） GradeBook.h
- （2）GradeBook.cpp

P83 Fig3.12

```
4.  #include <iostream>
5.  using std::cout;
6.  using std::endl;
7.
8.  #include "GradeBook.h" // include definition of class GradeBook
9.
10. // constructor initializes courseName with string supplied as argument
11. GradeBook::GradeBook( string name )
12. {
13.    setCourseName( name ); // call set function to initialize courseName
14. } // end GradeBook constructor
15.
16. // function to set the course name
17. void GradeBook::setCourseName( string name )
18. {
19.    courseName = name; // store the course name in the object
20. } // end function setCourseName
21.
22. // function to get the course name
23. string GradeBook::getCourseName()
24. {
25.    return courseName; // return object's courseName
26. } // end function getCourseName
27.
```

❖ **GradeBook.cpp**: 创建单独包含成员函数定义的源文件

3.7 Separating Interface from Implementation(eg6)

```
4.  #include <iostream>
5.  using std::cout;
6.  using std::endl;
7.
8.  #include "GradeBook.h" // include definition of class GradeBook
```

- 第4-6行
  - 必需的include和using声明
- 第8行：#include “GradeBook.h”
  - 确保编译时，成员函数头与头文件中的函数原型相符
  - 确保定义成员函数时，知道(可以使用)类的其它成员函数和数据成员

```
11    GradeBook::GradeBook( string name )
12    {
13       setCourseName( name ); // call set function to initialize courseName
14    } // end GradeBook constructor
15
16    // function to set the course name
17    void GradeBook::setCourseName( string name )
18    {
19       courseName = name; // store the course name in the object
20    } // end function setCourseName
21
22    // function to get the course name
23    string GradeBook::getCourseName()
24    {
25       return courseName; // return object's courseName
26    } // end function getCourseName
27
28    // display a welcome message to the GradeBook user
29    void GradeBook::displayMessage()
```

- 第11-34行
  - 成员函数定义, 11、17、23、29为新的函数头

3.7 Separating Interface from Implementation(eg6)

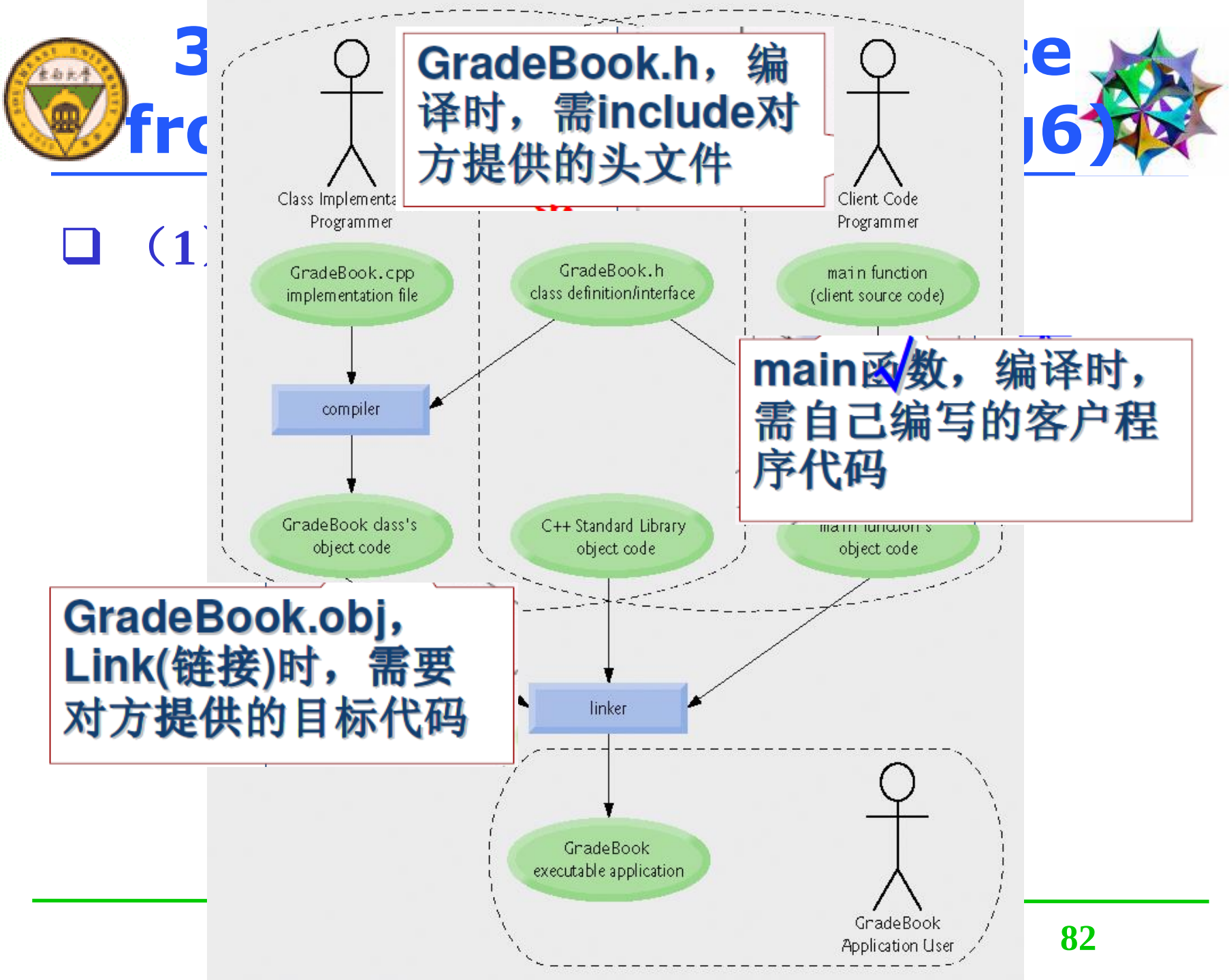
```
11    GradeBook::GradeBook( string name )
12    {
13       setCourseName( name ); // call set function to initialize courseName
14    } // end GradeBook constructor
15
16    // function to set the course name
17    void GradeBook::setCourseName( string name )
18    {
19       courseName = name; // store the course name in the object
20    } // end function setCourseName
```

- GradeBook::
  - :: binary scope resolution operator, 二元作用域操作符
  - 告诉编译器该函数是GradeBook类的成员函数

3.7 Separating Interface from Implementation(eg6)

- （1） GradeBook.h
- （2）GradeBook.cpp
- （3）fig03_13.cpp

  与上一节介绍的客户程序完全一致


GradeBook.h，编译时，需include对方提供的头文件
Class Implementa
Programmer
Client Code
Programmer
GradeBook.cpp
implementation file
GradeBook.h
class definition/interface
main function
(client source code)
compiler
main函数，编译时，需自己编写的客户程序代码
GradeBook class's
object code
C++ Standard Library
object code
object code
GradeBook.obj，Link(链接)时，需要对方提供的目标代码
linker
GradeBook
executable application
GradeBook
Application User

Topics

- 3.1 Introduction
- 3.2 Defining a Class with a Member Function(eg1)
- 3.3 Defining a Member Function with a Parameter(eg2)
- 3.4 Data Members, set Functions and get Functions(eg3)
- 3.5 Initializing Objects with Constructors(eg4)
- 3.6 Placing a Class in a Separate File for Reusability(eg5)
- 3.7 Separating Interface from Implementation(eg6)
- **3.8 Validating Data with set Functions(eg7)**

3.8 Validating Data with set Functions(eg7)

- 如何限定课程名不多于25个字符?
  - 对输入数据进行有效性验证
  - string类的length、substr成员函数的使用
  - if 语句的使用

3.8 Validating Data with set Functions(eg7)

- int string::length()
  - 函数名length, 返回类型int, 参数无
  - 返回string对象字符串的长度, 即包含多少字符
- string string::substr(int nPos, int nLength)
  - 函数名substr, 返回类型string, 输入两个int
  - 从原字符串中截取从nPos个字符开始的最多nLength长度的子字符串
  - 首字符为0起始

```
string aa = “hello world”;
string bb = aa.substr(3, 7);
```

lo worl

程序解读

```
18  void GradeBook::setCourseName( string name )          P87 Fig3.16
19  {
20     if ( name.length() <= 25 ) // if                   characters
21        courseName = name; // sto                       the object
22
23     if ( name.length() > 25 ) // if name has more th
24     {
25        // set courseName to first 25 characters of
26        courseName = name.substr( 0, 25 ); // start
27
28        cout << "Name \"" << name << "\" exceeds
29           << "Limiting courseName to first 25 chara
30     } // end if
31  } // end function setCourseName
```

```
if (condition)
    statement;
```

```
if (condition){
    statement_1;
    statement_2;
      ......
    statement_n;
}
```

- 第20-30行:

  如果长度不大于25，则保持不变;

  如果大于25，则截取前25个字符

```
18    void GradeBook::setCourseName( string name )          P87 Fig3.16
19    {
20       if ( name.length() <= 25 ) // if name has 25 or fewer characters
21          courseName = name; // store the course name in the object
22
23       if ( name.length() > 25 ) // if name has more than 25 characters
24       {
25          // set courseName to first 25 characters of parameter name
26          courseName = name.substr( 0, 25 ); // start at 0, length of 25
27
28          cout << "Name \"" << name << "\" exceeds maximum length (25).\n"
29             << "Limiting courseName to first 25 characters.\n" << endl;
30       } // end if
31    } // end function setCourseName
```

❖ 第28-29行

- 转义序列

```
Name "CS101 Introduction to Programming in C++" exceeds maximum length (25).
Limiting courseName to first 25 characters.
```

Summary

- 理解class、object、data member、
- member function基本概念
- 掌握类的定义、对象的创建和成员函数的调用
- 理解public/private两种访问权限指示符
- 理解数据成员和局部变量的区别
- 掌握构造函数和缺省构造函数
- 掌握函数原型的概念，以及类接口的定义
- 掌握基本的UML类图

Homework

- 实验必选题目：

  3.12, 3.15

- 实验任选题目：

  3.16

- 作业题目(Homework)：

  3.6, 3.11