第一章 开始

初识输入输出

- `istream` 输入流； `ostream` 输出流

  一个流就是一个**字符序列**

  > 流：随时间的推移，字符是顺序生成或消耗的

- `cin` 、 `cout` ：标准输入、输出；

  `cerr` 标准错误； `clog` 输出程序运行时的一般信息

  > 个人理解：将一个对象绑定到对应的流中

- `<<` 输出**运算符**：左侧的运算对象必须是一个 `ostream` 对象，右侧的运算对象是要打印的值。讲给定的值写到给定的 `ostream` 对象中

  > **输出运算符返回左侧的运算对象**（左值），所以输出才可以连续写，即(cout<<"enter")<<endl；左侧括号返回还是一个 `ostream` 对象
  >
  > **由此可以推断，许多可连续写的表达式都要有这种性质才行**

  标准库中定义了不同版本的输入输出运算符（重载），来处理不同类型的运算对象

- `endl` 操作符：结束当前行，并将与设备相关联的缓冲区中的内容刷新到设备中

  > 刷新的目的：保证目前为止所产生的所以输出都真正写入输出流中，而不是仅停留在内存中等待写入流

- **命名空间**：只要命名空间不同，就可以使用相同名字的变量

注释简介

- 一个注释不能嵌套在另一注释之内

```
1 /*  注释 /* */ */        优先匹配到*/后，注释部分就结束
```

控制流

- `for()` 中，循环体每次执行前都会检查循环条件，表达式在for循环体之后执行
- 使用标准输入对象作为条件是，其效果是**检测流的状态**。如果流有效，返回为真；如果遇到EOF或无效输入，则流无效，返回为假

```
// 示例
while(cin>>a)
```

> 从键盘输出文件夹结束符：
>
> Windows—— `Ctrl + z`
>
> Unix—— `ctrl + d`

类简介

类机制是C++最重要的特性之一

- 一个类定义了一个**类型**，以及与其相关联的一组操作。

  一般而言，类的作者定义了类类型对象上可以使用的所有操作（需要我们自己编写相应的行为，即类的成员函数）
- 当用点运算符访问一个成员函数时，要使用**调用运算符()来调用一个函数**

```
book.name();     // 返回书的名字,name是类中一个函数
book.name;       //返回书的名字，name
```

> 单纯一个函数，只是一个可调用对象；
>
> 函数() 才代表调用这个函数