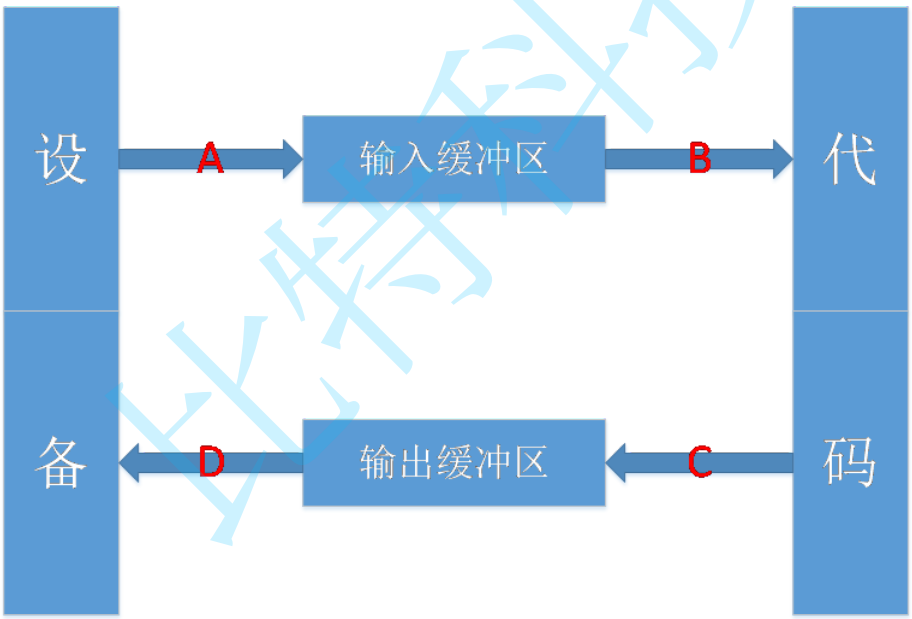
Lesson14---C++的IO流

【本节目标】

- **1. C语言的输入与输出**
- **2. 流是什么**
- **3. C++IO**
- **4. 文件流对象---读写序列化和反序列化**

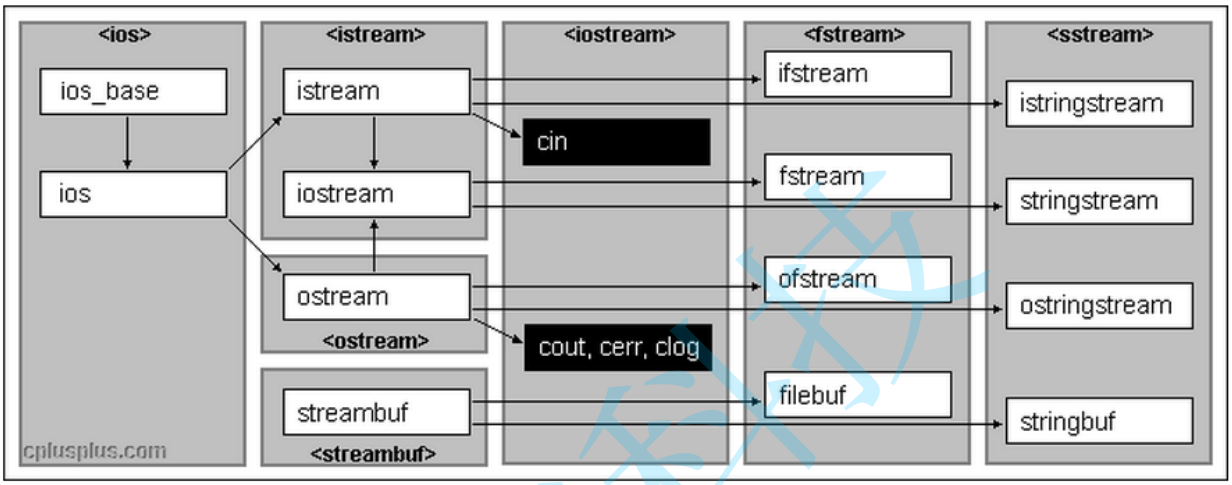
1. C语言的输入与输出

> C语言中我们用到的最频繁的输入输出方式就是**scanf ()与printf()。 scanf(): 从标准输入设备(键盘)读取数据，并将值存放在变量中。printf(): 将指定的文字/字符串输出到标准输出设备(屏幕)。注意宽度输出和精度输出控制。**C语言借助了相应的缓冲区来进行输入与输出。如下图所示：
>
> 
>
>
> 对**输入输出缓冲区**的理解：
>
> 1.可以**屏蔽掉低级I/O的实现**，低级I/O的实现依赖操作系统本身内核的实现，所以如果能够屏蔽这部分的差异，可以**很容易写出可移植的程序。**
>
> 2.可以**使用这部分的内容实现"行"读取的行为**，对于计算机而言是没有"行"这个概念，有了这部分，就可以定义"行"的概念，然后解析缓冲区的内容，返回一个"行"。

2. 流是什么

> “流”即是流动的意思，是物质从一处向另一处流动的过程，**是对一种**有序连续**且**具有方向性**的**数据（`其单位可以是bit,byte,packet`）**的**抽象描述。 C++流是指信息从外部输入设备（如键盘）向计算机内部（如内存）输入和从内存向外部输出设备（显示器）输出的过程。这种输入输出的过程被形象的比喻为“流”。
>
> 它的**特性**是：**有序连续、具有方向性**
>
> **为了实现这种流动，C++定义了I/O标准类库，这些每个类都称为流/流类，用以完成某方面的功能**

3. C++IO流

> **C++系统实现了一个庞大的类库，其中ios为基类，其他类都是直接或间接派生自ios类**
>
> 
>
>
> cerr和clog标准错误输出流，输出设备是显示器在流类库中，最重要的两部分功能为标准输入/输出（standard input/output）和文件处理。在C++的流类库中定义了四个全局流对象：cin，cout，cerr和clog：cin标准输入流对象，键盘为其对应的标准设备；cout标准输出流对象，显示器为标准设备。**在新库中要使用这四个功能，必须包含文件并引入std标准命名空间**
>
> **注意：**
>
> - cin为缓冲流。键盘输入的数据保存在缓冲区中，当要提取时，是从缓冲区中拿。如果一次输入过多，会留在那儿慢慢用，如果输入错了，必须在回车之前修改，如果回车键按下就无法挽回了。只有把输入缓冲区中的数据取完后，才要求输入新的数据。不可能用刷新来清除缓冲区，所以不能输错，也不能多输
> - 输入的数据类型必须与要提取的数据类型一致，否则出错。出错只是在流的状态字state中对应位置位（置1），程序继续。
> - 空格和回车都可以作为数据之间的分格符，所以多个数据可以在一行输入，也可以分行输入。但如果是字符型和字符串，则空格（ASCII码为32）无法用cin输入，字符串中也不能有空格。回车符也无法读入。

4. 文件流对象

> C++根据文件内容的数据格式分为二进制文件和文本文件
>
> 文件的操作步骤：
>
> 1. 定义一个文件流对象
>
> - ifstream ifile(只输入用)
> - ofstream ofile(只输出用)
> - fstream iofile(既输入又输出用)

2. 使用文件流对象的成员函数打开一个磁盘文件，使得文件流对象和磁盘文件之间建立联系

3. 使用提取和插入运算符对文件进行读写操作，或使用成员函数进行读写

4. 关闭文件

```
// 使用文件IO流用文本及二进制方式演示读写配置文件
struct ServerInfo
{
    char _ip[32];   // ip
    int _port;      // 端口
};

struct ConfigManager
{
public:
    ConfigManager(const char* configfile = "bitserver.config")
        :_configfile(configfile)
    {}

    void WriteBin(const ServerInfo& info)
    {
        // 这里注意使用二进制方式打开写
        ofstream ofs(_configfile, ifstream::in|ifstream::binary);
        ofs.write((const char*)&info, sizeof(ServerInfo));
        ofs.close();
    }

    void ReadBin(ServerInfo& info)
    {
         // 这里注意使用二进制方式打开读
        ifstream ifs(_configfile, ifstream::out | ifstream::binary);
        ifs.read((char*)&info, sizeof(ServerInfo));
        ifs.close();
    }

    void WriteText(const ServerInfo& info)
    {
        // 这里会发现IO流写整形比C语言那套就简单多了，
        // C 语言得先把整形itoa再写
        ofstream ofs(_configfile);
        ofs << info._ip << endl;
        ofs << info._port << endl;
        ofs.close();
    }

    void ReadText(ServerInfo& info)
    {
        // 这里会发现IO流读整形比C语言那套就简单多了，
        // C 语言得先读字符串，再atoi
        ifstream ifs(_configfile);
        ifs >> info._ip;
        ifs >> info._port;

        ifs.close();
```

```
    }

private:
    string _configfile; // 配置文件
};

int main()
{
    ConfigManager cfgMgr;

    ServerInfo wtinfo;
    ServerInfo rdinfo;
    strcpy(wtinfo._ip, "127.0.0.1");
    wtinfo._port = 80;

    // 二进制读写
    cfgMgr.WriteBin(wtinfo);
    cfgMgr.ReadBin(rdinfo);
    cout << rdinfo._ip << endl;
    cout << rdinfo._port << endl;

    // 文本读写
    cfgMgr.WriteText(wtinfo);
    cfgMgr.ReadText(rdinfo);
    cout << rdinfo._ip << endl;
    cout << rdinfo._port << endl;

    return 0;
}
```

5. 文档参考

> 具体的其他使用参考文档

【总结】

知识块	知识点	分类	掌握程度
C语言的输入与输出	输入：scanf（）；输出：printf（）	概念型	掌握
	输入输出都使用了缓冲区技术	原理型	了解
流的概念与特性	流的概念：物质从一处向另一处流动的过程，是对一种有序连续且具有方向性的数据（其单位可以是bit,byte,packet）的抽象描述	概念型	了解
	流的特性：有序、连续、具有方向性	概念型	掌握
C++IO流	庞大的类库，其中ios为基类，其他类都是直接或间接派生自ios类。分为标准输入输出 及 文件处理。	概念型	了解
	标准输入输出：std命名空间中cin，cout，cerr，clog是最常用	考点型	掌握
	文件处理：fstream ,ofstream,ifstream	考点型	了解
文件流对象 ---读写序列化和反序列化	fstream	应用型	掌握
	ifstream	应用型	掌握
	ofstream	应用型	掌握

【作业】

> 写一篇博客，归纳总结本节课的内容。