C/C++的文件操作

C++的文件操作：P.433 - P.447

13.4 对数据文件的操作与文件流

13.4中又用到了13.1-13.3的内容

C语言的文件操作：补充

讲课顺序：13.1-13.4

补充C语言对文件的操作

§13. 输入输出流

13.1. C++的输入与输出

13.1.1. 第3章中有关输入输出的概念

3.4.1. 流的基本概念

流的含义：流是来自设备或传给设备的一个数据流，由一系列字节组成，按顺序排列

★ C/C++没有支持输入/输出的语句

★ C语言用printf/scanf等函数来实现输入和输出，通过#include <stdio.h>来调用

★ C++通过cin和cout的流对象来实现，通过#include <iostream>来调用

cout：输出流对象　　<<：流插入运算符

cin：输入流对象　　>>：流提取运算符

3.4.2. 输出流的基本操作

格式：cout << 表达式1 << 表达式2 << ... << 表达式n;

★ 插入的数据存储在缓冲区中，不是立即输出，要等到缓冲区满(不同系统大小不同)或者碰到换行符("\n"/endl)或者强制立即输出(flush)才一齐输出

★ 默认的输出设备是显示器(可更改，称输出重定向)

★ 一个cout语句可写为若干行，或者若干语句

★ 一个cout的输出可以是一行，也可以是多行，多个cout的输出也可以是一行

★ 一个插入运算符只能输出一个值

★ 系统会自动判断输出数据的格式

§13. 输入输出流

13.1. C++的输入与输出

13.1.1. 第3章中有关输入输出的概念

3.4.3. 输入流的基本操作

格式：cin >> 变量1 >>变量2 >> ... >>变量n;

★ 键盘输入的数据存储在缓冲区中，不是立即被提取，要等到缓冲区满(不同系统大小不同)或碰到回车符才进行提取

★ 默认的输入设备是键盘(可更改，称输入重定向)

★ 一行输入内容可分为若干行，或者若干语句

★ 一个提取运算符只能输入一个值

★ 提取运算符后必须跟变量名,不能是常量/表达式等

★ 输入终止条件为回车、空格、非法输入

★ 系统会自动判断输入数据，若超过变量范围则错误

★ 字符型变量只能输入图形字符(33-126)，不能以转义符方式输入(单双引号、转义符全部当作单字符)

★ 浮点数输入时，可以是十进制数或指数形式，只取有效位数(4舍5入)

★ cin不能跟endl，否则编译错

§13. 输入输出流

13.1. C++的输入与输出

13.1.1. 第3章中有关输入输出的概念

3.4.5. 字符的输入和输出

3.4.5.1. 字符输出函数putchar

形式：putchar(字符变量/常量)

功能：输出一个字符

```
char a='A';
putchar(a);
putchar('A') ;
putchar('\x41');
putchar('\101');
```

★ 加#include <cstdio>或#include <stdio.h>

3.4.5.2. 字符输入函数getchar

形式：getchar()

功能：输入一个字符(给指定的变量)

★ 加#include <cstdio>或#include <stdio.h>

★ 返回值是int型，是输入字符的ASCII吗，可赋值给字符型/整型变量

★ 输入后，按回车结束输入

§13. 输入输出流

13.1. C++的输入与输出

13.1.2. 输入输出的基本概念

输入输出的种类：

系统设备：标准输入设备：键盘

(标准I/O) 标准输出设备：显示器

其它设备：鼠标、打印机、扫描仪等

外存文件：从文件中得到输入

(文件I/O) 输出到文件中

内存空间：输入/输出到一个字符数组中

(串I/O)

★ 操作系统将所有系统设备都统一当作文件进行处理

C++输入/输出的特点：

★ 与C兼容，支持printf/scanf

★ 对数据类型进行严格的检查，是类型安全的I/O操作

★ 具有良好的可扩展性，可通过重载操作符的方式输入/输出自定义数据类型

C++的输入/输出流：

★ 采用字符流方式，缓冲区满或遇到endl才输入/输出

★ cin,cout不是C++的语句，也不是函数，是类的对象 >> 和 << 的本质是左移和右移运算符，被重载为输入和输出运算符

§13. 输入输出流

13.1. C++的输入与输出

13.1.2. 输入输出的基本概念

C++中与输入/输出相关的类及对象：(P. 417 - 419)

iostream类库中有关的类

与iostream类库有关的头文件

在iostream头文件中定义的流对象

在iostream头文件中重载运算符

§13. 输入输出流

13.2. 标准输出流

13.2.1. cout，cerr和clog流

cout：向控制台进行输出，缺省是显示器

cerr：向标准出错设备进行输出，缺省是显示器
（直接输出，不必等待缓冲区满或回车）

clog：向标准出错设备进行输出，缺省是显示器
（放在缓冲区中，等待缓冲区满或回车才输出）

★ 三者的使用方法一样

★ 缺省都是显示器，可根据需要进行输出重定向

13.2.2. 格式输出(P.423-426表格及应用举例)

需要掌握的基本格式：

不同数制：dec、hex、oct

设置宽度：setw

左右对齐：setiosflags(ios::left/right)

其余当作手册来查：

P.54　表3.1

P.424 表13.3、13.4　(错误：所有iso=>ios)

★ 输出格式可用控制符控制，也可以流成员函数形式

P.424 例13.2

§13. 输入输出流

13.2. 标准输出流

13.2.3. 流成员函数put

形式：cout.put(字符常量/字符变量)

★ 功能与putchar相同，输出一个字符

```
char a='A';
cout.put(a);          //变量
cout.put('A') ;       //常量
cout.put('\x41');     //十六进制转义符
cout.put('\101');     //八进制转义符
cout.put(65) ;        //整数当作ASCII码
cout.put(0x41) ;      //整数当作ASCII码(十六)
cout.put(0101) ;      //整数当作ASCII码(八)
```

★ 允许连续调用

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

int main()
{
    cout.put(72).put(0x65).put('l').put(0154).put('a'+14);
    return 0;
}
```

Hello

```
//P.427 例13.3
#include <iostream>
using namespace std;
int main()
{   char *p="BASIC";
    for(int i=4; i>=0; i--)
       cout.put(*(p+i));
    cout.put('\n');
    return 0;;
}
```

CISAB

§13. 输入输出流

13.3. 标准输入流

13.3.1. cin流

★ cin提取数据后，会根据数据类型是否符合要求而返回逻辑值

```
#include <iostream>
using namespace std;
int main()
{
    int a=-9;
    cin >> a;
    cout << a << " " << (cin ? 1 : 0) << endl;
    return 0;
} //不同编译器，cin为0时，a值可能不同
```

输入	cout的结果	
10	10	1
ab	-9	0
12ab	12	1
很大的数字	-9	0

● 上例为VS2015下的运行结果

输入	cout的结果	
10	10	1
ab	0	0
12ab	12	1
很大的数字	2147483647	0

● 上例为CodeBlocks运行结果

循环一直执行，
直到输入为非数字格式

```
#include <iostream>
using namespace std;
int main()
{
    float grade;
    cout << "enter grade:";
    while(cin>>grade) {
        if (grade>=85 && grade<=100)
            cout << "Good!" << endl;
        if (grade<60)
            cout << "fail!" << endl;
        }
     return 0;
}
```

★ 允许进行输入重定向