



《计算概论A》课程 程序设计部分

流与文件

李 戈

北京大学 信息科学技术学院 软件研究所

lige@sei.pku.edu.cn



北京大学



流

■ 流

- ◆ 由于程序开发的需要，需要在程序间传递数据；
- ◆ 这些数据按顺序组成由若干字节组成的字节序列；
- ◆ 传递时，在内存中开辟一个内存缓冲区，用来存放流动中的数据。
- ◆ 缓冲区中的数据就是流；

■ 流的分类：

- ◆ 输出流：从内存中送一串字节到显示器、打印机；
- ◆ 输入流：从键盘送一串字节到内存；
- ◆ 文件流：从内存送一串字节到外存；





cout 流



北京大學



常用输入输出流

■ `iostream`中的输入输出流

◆ 标准输入流`cin`:

- 从标准输入设备输入到程序的数据流;

◆ 标准输出流`cout`:

- 从程序输出到标准输出设备的数据流;

◆ 出错信息流`cerr`和`clog`流:

- 向显示器输出出错信息;





cout流对象

■ cout (console output, 控制台输出)

- ◆ cout流在内存中对应开辟了一个缓冲区，用来存放流中的数据，
- ◆ 当向cout流插入一个endl时，不论缓冲区是否已满，都立即输出流中所有数据，并加入一个换行符，然后刷新流(清空缓冲区)。
- ◆ 用“cout<<”输出基本类型的数据时，不必考虑数据是什么类型，系统会判断数据的类型；





使用控制符控制输出格式

操 作 符	描 述	备 注
dec	按十进制输出	常量控制符 在 <code>iostream.h</code> 中
hex	按十六进制输出	
oct	按八进制输出	
endl	插入换行符，并刷新流	
ends	插入空字符	
setbase(n)	设置整数基数为 n	函数控制符 在 <code>iomanip.h</code> 中
setfill(c)	设置填充字符 c	
setprecision(n)	设置实数精度为 n	
setw(n)	设置字段宽度为 n	
setiosflags(ios::fixed)	设置浮点数以固定的小数位数显示	
setiosflags(ios::scientific)	设置浮点数以科学记数法显示	
setiosflags(ios::left)	输出数据左对齐	
setiosflags(ios::right)	输出数据右对齐	
setiosflags(ios::skipws)	忽略前导的空格	
setiosflags(ios::uppercase)	十六进制数大写输出	
setiosflags(ios::lowercase)	十六进制数小写输出	
setiosflags(ios::showpos)	输出正数时给出 “+” 号	
resetiosflags()	终止已设置的输出格式状态	


```
#include<iostream>
#include<iomanip>
using namespace std;
int main() {
    int a, b;
    cout<<"以八进制格式输入整数: \na=0";
    cin >> oct >> a;
    cout << "b=0";
    cin >> b; //仍然保持八进制设置
    cout<<"以十六进制显示整数 a=0x" << hex << a << ", b=0x" << b << endl;
    cout<<"以十进制显示整数  a=" << dec << a << ", b=" << b << endl;
    cout<<"以八进制显示整数  a=0" << oct << a << ", b=0" << b << endl;
    cout<<resetiosflags(ios::oct|ios::showbase);
    cout<<setiosflags(ios::hex|ios::showbase);
    cout<<"以十六进制显示整数 a="<<a<<",b="<<b<<endl;
    cout<<resetiosflags(ios::hex|ios::showbase);
    cout<<setiosflags(ios::dec|ios::showbase);
    cout<<"以十进制显示整数  a="<<a<<",b="<<b<<endl;
    cout<<resetiosflags(ios::dec|ios::showbase);
    cout<<setiosflags(ios::oct|ios::showbase);
    cout<<"以八进制显示整数  a="<<a<<",b="<<b<<endl;
    return 0;
}
```

```

#include<iostream>
#include<iomanip>
using namespace std;
int main() {
    int a, b;
    cout<<"以八进制格式输入整数: \na=0";
    cin >> oct >> a;
    cout << "b=0";
    cin >> b; //仍然保持八进制设置
    cout<<"以十六进制显示整数 a=0x" << hex << a << ", b=0x" << b << endl;
    cout<<"以十进制显示整数  a=" << dec << a << ", b=" << b << endl;
    cout<<"以八进制显示整数  a=0" << oct << a << ", b=0" << b << endl;
    cout<<resetiosflags(ios::oct|ios::showbase);
    cout<<setiosflags(ios::hex|ios::showbase);
    cout<<"以十六进制显示整数 a="<<a<<",b="<<b<<endl;
    cout<<resetiosflags(ios::hex|ios::showbase);
    cout<<setiosflags(ios::dec|ios::showbase);
    cout<<"以十进制显示整数  a="<<a<<",b="<<b<<endl;
    cout<<resetiosflags(ios::dec|ios::showbase);
    cout<<setiosflags(ios::oct|ios::showbase);
    cout<<"以八进制显示整数  a="<<a<<",b="<<b<<endl;
    return 0;
}

```

以八进制格式输入整数:

a=0 20

b=0 10

以十六进制显示整数 a=0x10, b=0x8

以十进制显示整数 a=16, b=8

以八进制显示整数 a=020, b=010

以十六进制显示整数 a=0x10, b=0x8

以十进制显示整数 a=16, b=8

以八进制显示整数 a=020, b=010

Press any key to continue.


```

#include<iostream>
#include<iomanip>
using namespace std;
int main() {
    float x = 35.54767f;
    int i;
    cout << x << endl;
    cout << setprecision(7)<<x<< endl;
    cout << setiosflags(ios::fixed);
    for (i = 0; i < 6; i++)
        cout<<setprecision(i)<<x<<endl;
    cout<<resetiosflags(ios::fixed)<<endl;
    cout<<setiosflags(ios::scientific);
    for (i = 0; i < 6; i++)
        cout<<setprecision(i)<<x<<endl;
    return 0;
}

```

```

35. 5477
35. 54767
36
35. 5
35. 55
35. 548
35. 5477
35. 54767
3. 554767e+001
3. 6e+001
3. 55e+001
3. 555e+001
3. 5548e+001
3. 55477e+001
Press any key to continue

```

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
```

```
{
    int i;
    double values[ ] = {1.23, 35.36, 653.7, 4358.24};
    char *names[ ] = {"Zoot", "Jimmy", "Al", "Stan"};
    // cout << setiosflags(ios::scientific);
    for(i=0; i<4; i++)
    {
        cout << setiosflags(ios :: left)
            << setw(6) << names[i]
            << resetiosflags(ios :: left)
            << setw(10) << setprecision(1) << values[i] << endl;
    }
    return 0;
}
```

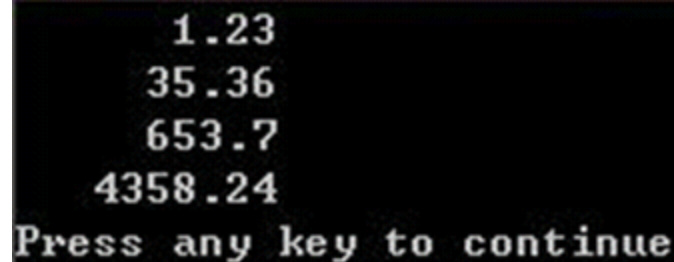
```
Zoot      1
Jimmy     4e+001
Al        7e+002
Stan      4e+003
Press any key to continue_
```

```
Zoot      1.2e+000
Jimmy     3.5e+001
Al        6.5e+002
Stan      4.4e+003
Press any key to continue
```



控制输出宽度 (1)

```
#include <iostream>
using namespace std;
int main()
{
    int i;
    double values[ ] = {1.23, 35.36, 653.7, 4358.24};
    for(i=0; i<4; i++)
    {
        cout.width(10);
        cout << values[i]<<endl;
    }
    return 0;
}
```



```
1.23
35.36
653.7
4358.24
Press any key to continue
```



控制输出宽度 (2)

```
#include <iostream>
using namespace std;
int main()
{
    int i;
    double values[ ] = {1.23, 35.36, 653.7, 4358.24};
    for(i=0; i<4; i++)
    {
        cout.width(10);
        cout.fill('*');
        cout << values[i] << endl;
    }
    return 0;
}
```

```
*****1.23
*****35.36
*****653.7
***4358.24
Press any key to continue
```



控制输出宽度 (3)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
```

```
    int i;
    double values[ ] = {1.23, 35.36, 653.7, 4358.243456789123245};
    char *names[ ] = {"Zoot", "Jimmy", "Al", "Stan"};
    for(i=0; i<4; i++)
    {
        cout << setw(6) << names[i] << setw(10) <<
            setprecision(15) << values[i] << endl;
    }
    return 0;
}
```

```
Zoot      1.23
Jimmy     35.36
Al        653.7
Stan4358.24345678912
Press any key to continue
```



控制输出对齐 (1)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
```

```
Zoot      1.23
Jimmy     35.36
Al        653.7
Stan      4358.24
Press any key to continue
```

```
    int i;
    double values[ ] = {1.23, 35.36, 653.7, 4358.24};
    char *names[ ] = {"Zoot", "Jimmy", "Al", "Stan"};
    for(i=0; i<4; i++)
    {
        cout << setiosflags(ios::left)
              << setw(6) << names[i]
              << resetiosflags(ios::left)
              << setw(10) << values[i] << endl;
    }
    return 0;
}
```





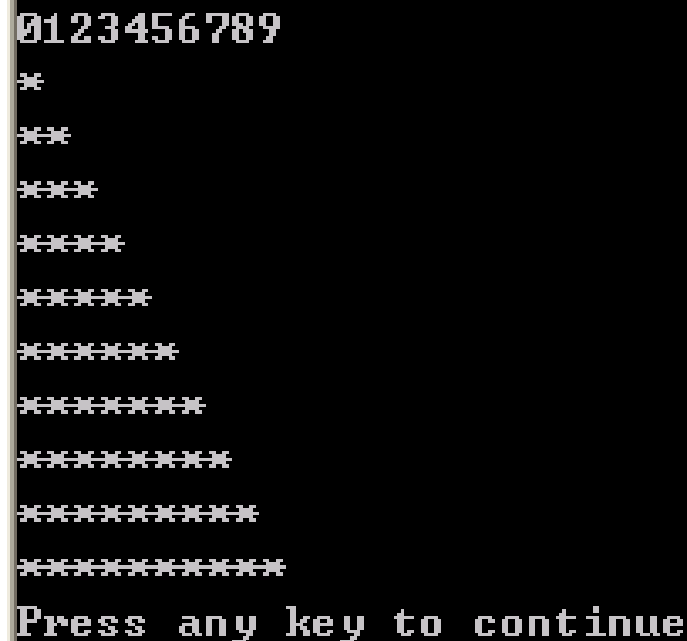
控制输出对齐 (2)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int i=0;i<10;i++)cout<<i;
    cout<<endl;
    for(int x=1;x<=10;x++)
        cout<<setfill('*')<<setw(x)<<"*"<<endl ;
    return 0;
}
```



控制输出对齐 (2)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int i=0;i<10;i++)cout<<i;
    cout<<endl;
    for(int x=1;x<=10;x++)
        cout<<setfill('*')<<setw(x)<<"*"<<endl ;
    return 0;
}
```



```
0123456789
*
**
***
****
*****
*****
*****
*****
*****
*****
*****
*****
Press any key to continue
```





控制输出对齐 (3)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int i=0;i<10;i++)cout<<i;
    cout<<endl;
    for(int x=1;x<=9;x++)
        cout<<setfill(' ')<<setw(10-x)
            <<"<<setfill('*')<<setw(x)<<"*"<<endl;
    cout<<"*****" <<endl;
    return 0;
}
```



控制输出对齐 (3)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int i=0;i<10;i++)cout<<i;
    cout<<endl;
    for(int x=1;x<=9;x++)
        cout<<setfill(' ')<<setw(10-x)
            <<"<<setfill('*')<<setw(x)<<"*"<<endl;
    cout<<"*****" <<endl;
    return 0;
}
```



```
0123456789
          *
         **
        ***
       ****
      *****
     ******
    *******
   ********
  *********
 Press any key to continue
```





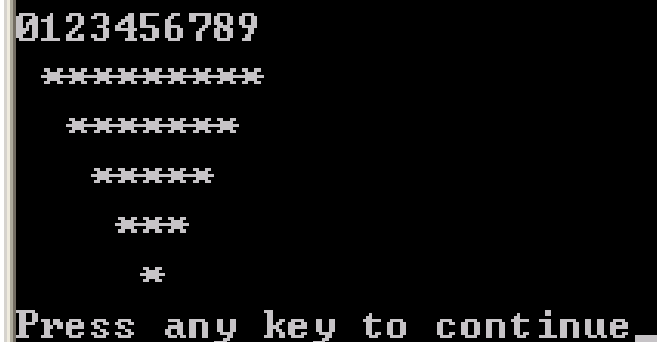
控制输出对齐 (4)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int i=0;i<=9;i++)cout<<i;
    cout<<endl;
    for(int x=1;x<=5;x++)
        cout<<setfill(' ')<<setw(x)<<" "<<setfill('*')
        <<setw(11-2*x)<<"*"<<endl;
    return 0;
}
```



控制输出对齐 (4)

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    for(int i=0;i<=9;i++)cout<<i;
    cout<<endl;
    for(int x=1;x<=5;x++)
        cout<<setfill(' ')<<setw(x)<<" "<<setfill('*')
        <<setw(11-2*x)<<"*"<<endl;
    return 0;
}
```

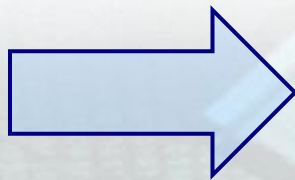


```
0123456789
*****
*****
*****
***
*
Press any key to continue_
```





cin 流



北京大学



cin流对象

■ cin流对象

- ◆ 从标准输入设备(键盘)获取数据;
- ◆ 程序中的变量通过流提取运算符 “>>” 从流中提取数据;
- ◆ 流提取符 “>>” 从流中提取数据时通常跳过输入流中的空格、**tab**键、换行符等空白字符;
- ◆ 遇到无效字符或文件结束标志, 停止提取;
- ◆ 只有在输入完数据再按回车键后, 该行数据才被送入键盘缓冲区, 形成输入流, 提取运算符 “>>” 才能从中提取数据;



cin流对象

```
#include<iostream>
#include<iomanip>
using namespace std;
int main() {
    float grade;
    cout<<"enter grade:";
    while (cin>>grade)           //能从cin流读取数据
    {
        if(grade>=85)
            cout<<grade<<"GOOD!"<<endl;
        if(grade<60)
            cout<<grade<<"fail!"<<endl;
        cout<<"enter grade:";
    }
    cout<<"The end."<<endl;
    return 0; }
```



北京大學



用于字符输入的流成员函数

■ get函数

- ◆ 用于读入一个字符；
- ◆ 共有3种形式: 无参数，一个参数，3个参数。

■ 不带参数的get函数cin.get()

- ◆ 用来从指定的输入流中提取一个字符，函数的返回值就是读入的字符。若遇到输入流中的文件结束符，则函数值返回文件结束标志EOF(End Of File) 。





例：用get函数读入字符

```
#include <iostream>
using namespace std;
int main( )
{   int c;
    cout<<"enter a sentence:"<<endl;
    while((c=cin.get())!=EOF)
        cout.put(c);
    return 0;
}
```

运行情况如下：

enter a sentence:

I study C++ very hard.↵ (输入一行字符)

I study C++ very hard. (输出该行字符)

^Z↵ (程序结束)



北京大学

用于字符输入的流成员函数

- 有一个参数的get函数cin.get(ch)
 - ◆ 从输入流中读取一个字符，赋给字符变量ch。
 - ◆ 如果读取成功则函数返回非0值(真)，如失败(遇文件结束符) 则函数返回0值(假)。

```
int main( ){  
    char c;  
    cout<<"enter a sentence:"<<endl;  
    while(cin.get(c))  
        //读取一个字符赋给字符变量c,  
        //如果读取成功, cin.get(c)为真  
    {    cout.put(c);}  
    cout<<"end"<<endl;  
    return 0;  
}
```





用于字符输入的流成员函数

■ 有3个参数的get函数

- ◆ `cin.get(字符数组, 字符个数n, 终止字符)`
- ◆ 从输入流中读取n-1个字符, 赋给指定的字符数组(或字符指针指向的数组);
- ◆ 如果在读取n-1个字符之前遇到指定的终止字符, 则提前结束读取;
- ◆ 如果读取成功则函数返回非0值(真), 如失败(遇文件结束符) 则函数返回0值(假)。
- ◆ `get`函数中第3个参数可以省写, 此时默认为'`\n`'





例：用get函数读入字符

```
#include <iostream>
using namespace std;
int main( )
{ char ch[20];
  cout<<"enter a sentence:"<<endl;
  cin.get(ch, 10, '\n'); //指定换行符为终止字符
  cout<<ch<<endl;
  return 0;
}
```





用于字符输入的流成员函数

■ 成员函数getline()

- ◆ 从输入流中读取一行字符
- ◆ 用法与带3个参数的get函数类似:

`cin.getline(字符数组(或字符指针), 字符个数n, 终止标志字符)`

■ `getline`与`get`

- ◆ `getline`遇到终止标志字符时结束，缓冲区指针移到终止标志字符之后；
- ◆ `get`遇到终止字符是停止读取，指针不移动



例：用getline函数读入字符

```
#include <iostream>
using namespace std;
int main() {
    char ch[20];
    cout<<"enter a sentence:"<<endl;
    cin>>ch;
    cout<<"The string read with cin is:"<<ch<<endl;
    cin.getline(ch, 20, '/'); //读19个字符或遇'/'结束
    cout<<"The second part is:"<<ch<<endl;
    cin.getline(ch, 20);      //读19个字符或遇'/n'结束
    cout<<"The third part is:"<<ch<<endl;
    return 0;
}
```

程序运行情况如下：

enter a sentence: I like C++./I study C++./I am happy.✓

The string read with cin is: I

The second part is: like C++.

The third part is: I study C++./I am h



北京大學



例：用cin 和 getline读入字符串

```
#include<iostream>
#include<stdio.h>
using namespace std;
void main()
{
    char a[10][10];
    int n=0;
    cin>>n;
    for(int i=0;i<n;i++)
        cin.getline(a[i], 10);
    for(i=0;i<n;i++)
        cout<<a[i]<<endl;
}
```

```
#include<iostream>
#include<stdio.h>
using namespace std;
void main()
{
    char a[10][10];
    int n=0;
    cin>>n;
    cin.get( );
    for(int i=0;i<n;i++)
        cin.getline(a[i],10);
    for(i=0;i<n;i++)
        cout<<a[i]<<endl;
}
```





文 件



北京大學



有关文件的概念和名词

■ 文件

- ◆ 存储在外部介质上的数据的集合。

■ 文件流

- ◆ 要以磁盘文件为对象进行输入/输出，必须定义一个文件流类的对象；
- ◆ 通过文件流对象将数据从内存输出到磁盘文件，或者通过文件流对象从磁盘文件将数据输入到内存；





文件的种类

■ 文本文件

- ◆ 又称**ASCII**文件，这种文件在磁盘中存放时每个字符对应一个字节，用于存放对应的**ASCII**码。例如，数**5678**的存储形式为：

ASC码： 00110101 00110110 00110111 00111000



十进制码： 5 6 7 8

- ◆ **ASCII**码文件可在屏幕上按字符显示用**DOS**命令**TYPE**可显示文件的内容。



北京大學



文件的种类

■ 二进制文件

◆ 按二进制的编码方式来存放的文件

- 5678的存储形式为： 00010110 00101110 占二个字节
- 在文本模式中“回车”被当成一个字符‘\n’，而二进制模式认为它是两个字符0x0D,0x0A；
- 如果在文件中读到0x1B，文本模式会认为这是文件结束符，二进制方式则不会；

- ◆ 系统在处理上述文件时，都将其看成字符流，按字节进行处理。
- ◆ 系统处理文本文件时，会按一定的对应关系对文件中的数据作相应的转换；处理二进制文件时，则不作转换处理；





文件的操作

■ 文件的相关操作

- ① 建立文件对象;
- ② 打开文件;
- ③ 对文件进行操作;
 - 读
 - 写
 - 搜
- ④ 关闭文件;

■ 打开文件

在内存中建立缓冲区

- ◆ 如打开成功，打开函数返回一个内存地址值，由一个文件指针接收;
- ◆ 如内存不可建立缓冲区，则打开失败，打开函数返回NULL;

■ 关闭文件

将文件送回磁盘，并从内存中清除，及时释放内存空间，以保证文件安全;





文件操作的工具

- 在C中——文件指针

- ◆ 指向含有文件信息结构的指针;

- 在 C++中——I/O 类库中定义了几种文件类:

- ◆ ifstream 类: 用来支持从磁盘文件的输入;

- ◆ ofstream 类: 用来支持从磁盘文件的输出;

- ◆ fstream 类: 用来支持从磁盘文件的输入输出;



北京大學



创建并打开一个文件

■ 先建立流类对象后打开文件

- ◆ `ofstream outfile;`

 - `//定义ofstream 类(输出文件流类)对象outfile`

- ◆ `outfile.open("myfile.dat", ios::out);`

 - `//调用文件流的成员函数open，使文件流与
//myfile.dat 文件建立关联`

- ◆ 可以写出文件路径，如：“C:\\Temp\\myfile.dat”；默认情况下，在当前目录下创建文件；

- ◆ 若文件尚未存在则在Open的时候创建该文件；

- ◆ 若文件已经存在，则按照**指定的方式**（覆盖/新增）打开该文件；默认情况下，以输出方式打开；



如何指定打开文件的方式?

文件打开方式	功 能
<code>ios::in</code>	以输入方式打开文件
<code>ios::out</code>	以输出方式打开文件(默认方式)
<code>ios::app</code>	以追加方式打开文件, 将所有输出写入文件末尾
<code>ios::ate</code>	打开一个已有文件, 文件指针指向文件末尾
<code>ios::trunc</code>	如果文件存在, 删除文件原有内容, 否则创建新文件
<code>ios::binary</code>	以二进制方式打开文件, 默认时为文本文件
<code>ios::nocreate</code>	打开已有文件, 不存在, 则打开失败
<code>ios::noreplace</code>	如文件不存在则建立新文件, 文件已存在, 则操作失败
<code>ios::in ios::out</code>	以输入/输出方式打开文件, 文件可读可写
<code>ios::out ios::binary</code>	以二进制方式打开一个输出文件
<code>ios::in ios::binar</code>	以二进制方式打开一个输入文件





创建并打开一个文件

还可以写成这样:

- 建立流类对象的同时完成打开文件

- ◆ `ofstream outfile("myfile.dat");`

- `//建立一个输出流类对象`

- 也可以用`fstream`类来完成

- ◆ `fstream outfile("myfile.dat", ios::out);`

- `//通过outfile 对磁盘文件myfile.dat 进行写操作`

- 可以采用“位或”运算符 | 组合上述模式

- ◆ `ios::in|ios::out`

- ◆ `ios::in|ios::nocreate`

- ◆ `ios::in|ios::binary`



北京大學



使用模式

- 如何判定一个文件是否被成功创建或打开
 - ◆ 如果打开操作失败，则流对象的值为0。

```
ofstream outfile;  
outfile.open("f1.dat");  
if (!outfile)  
{  
    cerr<<"open error!"<<endl;  
    return 1;  
}  
//操作文件  
outfile.close();
```



文件流错误处理状态函数

函 数	功 能
is_open	流对象是否与一个打开的文件相联系，若是则返回 true，否则返回 false
bad	如果进行了非法操作返回 true，否则返回 false
good	操作成功返回 true，否则返回 false
eof	若到文件尾则返回 true，否则返回 false
fail	操作失败返回非 0 值
clear	设置内部错误状态，如果用默认参量调用，则清除所有错误位
rdstate	返回当前错误状态

```
if(outfile.is_open()==0)
{
    cerr<<"open error!"<<endl;
    return 1;
}
```



北京大學



对文件的操作

■ 读/写

- ◆ ASCII文件的读写操作
- ◆ 二进制文件的读写操作



北京大学



ASCII文件的读/写方式

C++支持两种读写文件的方法

■ 使用 “>>”运算符和 “<<”运算符读/写数据

◆ `ofstream outfile;` `outfile`与`cout`用法类似

◆ `ifstream infile;` `infile`与`cin`用法类似

```
outfile << 1 << " " << 2 << endl;
```

```
infile >> a >> b; //把1, 2输入到a, b中
```



北京大學

例1：将整数存入文件

- 从键盘输入10个整数，并保存到磁盘文件中，再从文件中读出数据，放入数组。

```
#include<iostream>
```

```
#include <fstream>
```

```
using namespace std;
```

```
int main( )
```

```
{ int a[10];
```

```
    ofstream outfile("f1.dat");
```

```
    if (!outfile)
```

```
{
```

```
        cerr<<"open error!"<<endl;
```

```
        return 1;
```

```
}
```



```
cout<<"enter 10 integer numbers: "<< endl;
for(int i = 0;i < 10;i++)
{
    cin>>a[i];
    outfile <<a[i]<<" ";           //用空格分开整数
}
outfile.close();
ifstream infile("f1.dat");
if (!infile)
{   cerr<<"open error!"<<endl; return 1; }

for(i = 0; i < 10; i++)
{
    infile>>a[i];                  //存入的空格并不读出
    cout << a[i]<<" ";
}
cout<<endl;  infile.close();
return 0;
}
```

例2：将字符存入文件

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    ofstream outfile;
    outfile.open("abc.txt");           //以写的方式打开文件abc.txt
    if(!outfile)                       //检查文件是否正常打开
    {
        cout<<"abc.txt can't open"<<endl;
        return 1;                     //abort(); //打开失败，结束程序
    }
    char ch='a';
    while(ch<='z')
    {
        outfile<<ch;                 //通过插入操作将内存变量的内容写到文件中
        ch++;
    }
    outfile.close();
    fstream infile("abc.txt",ios::in); //以读的方式打开文件abc.txt
    while(infile>>ch)                 //通过提取操作从文件中读数据到内存变量
        cout<<ch;
    cout<<endl;
    return 0;
}
```

例3：将二维表存入文件

	Mon	Tue	Wen	Thu	Fri
John	10	3	11	9	7
Mike	11	15	4	3	7
Tom	9	10	12	2	9
Ben	12	5	7	14	12

- `char Name[10][4]; int Sum[10][5];`
- `int num;`



```
void savedata(int num, char Name[ ][4], int Sum[ ][5])
{
    ofstream outfile("data.txt");
    outfile << num << " ";
    for (int i = 0; i < num; i++)
    {
        outfile << Name[i] << " ";
    }
    for (int n = 0; n < num; n++)
        for(int j = 0; j < 5; j++)
            outfile << Sum[n][j] << " ";
}
```

4 John Mike Tom Ben 10 3 11 9 7 11 15 4 3 7

```

int loaddata(char Name[][3], int Sum[][5])
{
    int num;
    ifstream infile("data.txt");
    infile >> num;
    for (int i = 0; i < num; i++)
    {
        infile >> Name[i];
    }
    for (int n = 0; n < num; n++)
        for(int j = 0; j < 5; j++)
            infile >> Sum[n][j];
    return num;
}

```

	Mon	Tue	Wen	Thu	Fri
John	10	3	11	9	7
Mike	11	15	4	3	7
Tom	9	10	12	2	9
Ben	12	5	7	14	12



ASCII文件的读/写方式

C++支持两种读写文件的方法

- 使用文件对象调用put, get, getline 等成员函数, 进行数据读写;

函 数	功 能
输入函数	
read()	从文件读取特定数量的字节
get()	读取一个字符或一串字符
getline()	读取特定数量的字符, 或直到\n 为止
peek()	读取下一个字符, 但不删除当前字符
输出函数	
open()	把流与一个特定的磁盘文件关联起来。需要指定打开模式
write()	向文件写入一定数量的字节
put()	向文件写入一个字符
close()	关闭与一个输出文件流关联的磁盘文件

例1：将字符串存入文件

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    char s[100];
    ofstream outf;
    outf.open("dialogue.dat");
    if(!outf)
    {
        cout<<"can't open dialogue.dat"<<endl;
        abort();
    }
    outf<<"How are you?\n";           //将若干字符串写到文件中
    outf<<"I'm fine,thank you, and you?\n";
    outf<<"I'm very well.\n";
    outf.close();
    ifstream inf("dialogue.dat");
    while(!inf.eof())                //判断是否到文件结尾
    {
        inf.getline(s,sizeof(s));    //从文件中读一行字符串到数组
        cout<<s<<endl;
    }
    system("pause");
    return 0;
}
```



例2：文本文件的拷贝

```
#include <fstream>
```

```
int main()
```

```
{  char ch, f1[10], f2[10];
```

```
    cout<<“Enter the source file name: ”;
```

```
    cin.getline(f1, 20);
```

```
    ifstream infile(f1);
```

```
    if (!infile)
```

```
    {    cerr<<“open error!”<<endl;
```

```
        return 1;
```

```
    }
```

第一步：打开源文件；



北京大學



例2：文本文件的拷贝

```
cout<<“Enter the destination file name: ”;
```

```
cin.getline(f2, 20);
```

```
ofstream outfile(f2);
```

```
if (!outfile)
```

```
{
```

```
    cerr<<“open error!”<<endl;
```

```
    return 1;
```

```
}
```

第一步：打开源文件；
第二步：打开目标文件；



北京大學



例2：文本文件的拷贝

```
while(infile.get(ch))  
{  
    outfile.put(ch);  
}  
  
infile.close();  
outfile.close();  
return 0;
```

运行情况：

Enter the infile name:

File1.cpp

Enter the outfile name:

File2.cpp

验证

c>type file1.cpp

c>type file2.cpp



北京大学



例3：统计字符串出现次数

■ 问题：

- ◆ 统计一个文件中某字符串出现的次数；

■ 思路：

- ◆ 把文件中的内容全部读到一个数组中，看作一个长串
- ◆ 在长串中匹配，用系统提供的函数：
 - `char *strstr(*char, *char);`
- ◆ 给出两个串的首地址，返回匹配处的地址。

`char *p = strstr(longstr, str);`



北京大學



例3：统计字符串出现次数

```
#include <iostream>
#include <fstream>
#include <cstring>
#include <iomanip>
using namespace std;
int main( )
{ char f[] = "test.cpp";
  ifstream fin(f);
  if (!fin)
  {
    cout<<"open file failed" <<endl;
    return 1;
  }
```



```
char str[ ]="str";
int lenStr= strlen(str);
int n = 0;
char word[50];
while(fin >> setw(50) >> word)           //没到文件结束
{
    char *newWord = word;
    char *p = strstr(newWord, str);
    while( p != NULL )                     //没有读到数组尾
    {
        n++; newWord = p+lenStr;
        p = strstr(newWord, str);
    }
}
fin.close( );
cout<<f<<" has " <<n<<"<<"\ "<<str<<"\ "<<endl;
return 0;
}
```



二进制文件的创建与打开

```
ofstream outfile("file.dat", ios::binary);
```

```
if(!outfile)
```

```
{
```

```
    cerr<<"open error!"<<endl;
```

```
    abort( );//退出程序
```

```
}
```

```
//文件操作
```

```
outfile.close( );
```

默认情况下，打开/创建文本文件



北京大学



二进制文件的读/写

■ 对二进制文件的读写主要用 `istream` 类的成员函数 `read` 和 `write` 来实现:

◆ `istream& read(char *buffer,int len);`

◆ `ostream& write(const char * buffer,int len);`



北京大學

二进制文件的读

读二进制文件相当于将文件中的数据原样照搬到内存中

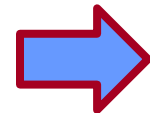
```
istream & read(char *buffer, int len);
```

//从文件中读入len个字节的内容，存放在字符指针buffer指向的地址

```
ifstream inf("data.bin", ios::binary);  
if(!inf) return 0;  
int array2[10];  
inf.read((char *)array2, sizeof(array));  
inf.close();  
for(i=0; i<10; i++) cout<<array2[i]<<" ";
```

0 1 2 3 4 5 6 7 8 9

data.bin x																
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00000000h:	00	00	00	00	01	00	00	00	02	00	00	00	03	00	00	00
00000010h:	04	00	00	00	05	00	00	00	06	00	00	00	07	00	00	00
00000020h:	08	00	00	00	09	00	00	00								



0

1

2

3

4

5

6

7

8

9

二进制文件的写

写二进制文件相当于将内存中的数据及其存储方式原样照搬到文件中

`ostream & write(const char *buffer, int len);`

//将字符指针buffer指向地址开始的len个字节的内容不加转换地写入文件

```
ofstream of("data.bin",ios::binary);  
if(!of) return 0;  
int array[10],i;  
for(i=0;i<10;i++) array[i]=i;  
of.write((char *)array,sizeof(array));  
of.close();
```

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00000000h:	00	00	00	00	01	00	00	00	02	00	00	00	03	00	00	00
00000010h:	04	00	00	00	05	00	00	00	06	00	00	00	07	00	00	00
00000020h:	08	00	00	00	09	00	00	00								

0

1

2

3

4

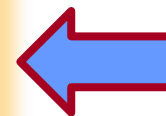
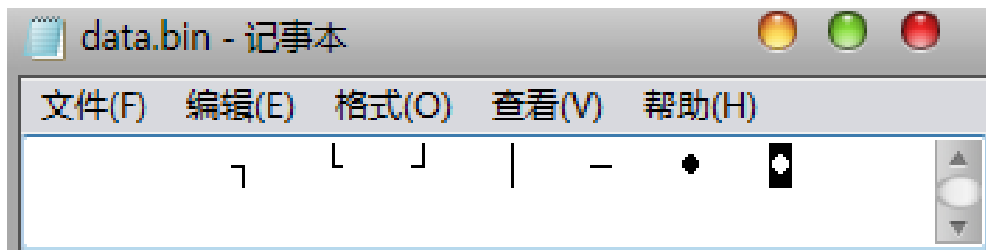
5

6

7

8

9





二进制文件的操作

■ 用流文件的read和write操作

◆ **istream& read(char *buffer, int len);**

● **buffer**指向内存中的一段存储空间，**len**是字节数

● 如果文件中没有足够的数量，则遇到文件结束为止。

● **outfile.read((char)(stud+i), sizeof(student))**

◆ **ostream& write(const char * buffer, int len);**

● **outfile.write((char)(stud+i), sizeof(student))**



北京大學



二进制文件的操作

- 将结构体数据写入文件。

```
void display(struct student *stud);
```

```
void writeToFile(struct student *stud);
```

```
void readFromFile(struct student *stud);
```

```
struct student
```

```
{ char name[20];
```

```
    int num;
```

```
    int age;
```

```
    char sex;
```

```
};
```



北京大學



二进制文件的读写

```
int main( )  
{ student stud1[3]={ "Li",1001,18,'f',  
                      "Fang",1002,19,'m',  
                      "Wang",1004,7,'f'    };  
  
  student stud2[3];  
  writeToFile(stud1);  
  readFromFile(stud2);  
  display(stud2);  
  return 0;  
}
```



二进制文件的读写

```
void writeToFile(struct student *stud)
{
    ofstream outfile("student.dat", ios::binary);
    if (!outfile)
    {
        cerr<<"open outfile error!"<<endl;
        exit(1);
    }
    for (int i = 0; i < 3; i++)
        outfile.write((char *)(stud+i), sizeof(stud[i]));
    outfile.close();
}
```



二进制文件的读写

```
void readFromFile(struct student *stud)
{
    ifstream infile("student.dat", ios::binary);
    if (!infile)
    {
        cerr<<"open outfile error!"<<endl;
        exit(1);
    }
    for (int i = 0; i < 3; i++)
        infile.read((char *)(stud+i), sizeof(stud[i]));
    infile.close();
}
```





二进制文件的读写

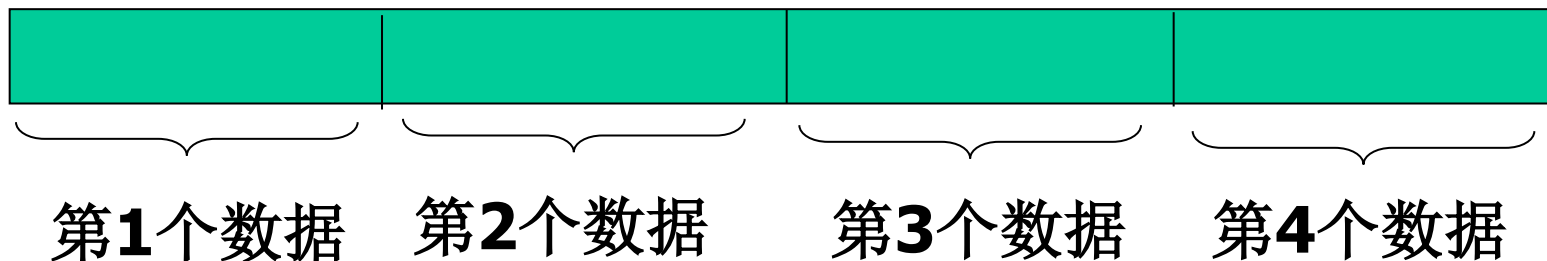
```
void display(struct student *stud)
{
    for (int i = 0; i < 3; i++)
    {
        cout << "No. " << i+1 << endl;
        cout << "Name: " << stud[i].name << endl;
        cout << "num: " << stud[i].num << endl;
        cout << "age: " << stud[i].age << endl;
        cout << "sex: " << stud[i].sex << endl;
    }
}
```



随机访问文件

■ `outfile.seekp((n - 1) * sizeof(student))`

将指针指向从文件头开始的第n个字节的地方



■ 适合随机访问的情况

- ◆ 数据具有相同的结构，例如，结构体。
- ◆ 文本文件/二进制文件都可以；



北京大学



修改文件的内容

■ 例如：

- ◆ 一个文件保存若干数据，将其中第 n 个数据用变量`data`的值进行替换。

■ 两种方式

- ◆ 顺序文件操作，将文件内容读到数组，修改数组的数据，再将数组全部写回文件。
- ◆ 随机文件操作，将指针指到 n ，直接写入`data`。




```
int main()  
{   int n,data;  
    cout << “which data need to update” << endl;  
    cin >> n;  
    cout << “give the number to update” << endl;  
    cin >> data;  
    fstream file;  
    file.open("data.txt", ios::out|ios::in);  
    file.seekp( (n-1) * sizeof(int));  
    file.write((char *)(&data), sizeof(int));  
    file.close();  
    return 0;  
}
```



应用实例

- 成绩管理

- 要求功能:

- 1、输入数据(顺序)
- 2、统计功能(按班号)
- 3、列表功能(按学号)
- 4、查询功能(按学号)
- 5、删除功能(按学号)



北京大學



应用实例

■ 系统的框架：

- ◆ 有一个文件，存放学生记录数据，所有操作都是基于这个文件。
- ◆ 提供用户界面，让用户可以任意选择不同功能。
- ◆ 将各个功能模块写成函数，并由统一界面管理。

学号	姓名	班级	期中	作业	期末	总成绩	
1001	***	1	78	80	82		
1002	***	2	85	86	66		
1003	***	3	80	91	85		
...		
1180	***	4	90	??	77		



应用实例

■ 用户界面

欢迎使用成绩管理程序

1. 输入记录
2. 统计成绩
3. 全部显示
4. 查询成绩
5. 删除记录
6. 退 出



北京大学

应用实例

```
struct student
{   int    ID;
    char    name[10];
    int     age;
    int     class1;
    float    middle ;
    float    hw;
    float    end;
    float    score;
    char     tag;
}
```





statement.h

- **void append();**
- **void statistic();**
- **void listall();**
- **void search();**
- **void dele();**

```
struct student
{ int  ID;
  char  name[10];
  int   age;
  int   class1;
  float middle ;
  float hw;
  float end;
  float final;
  char  tag;
}
```





myfunc.h

```
void append()
```

```
{ ; }
```

```
void listall()
```

```
{ ; }
```

```
void statistic()
```

```
{ ; }
```

```
void dele()
```

```
{ ; }
```

```
void search()
```

```
{ ; }
```



北京大學


```
#include <iostream> #include <fstream> ##include <iomanip>
```

```
#include "statement.h"
```

```
#include "myfunc.h"
```

```
using namespace std;
```

```
int main()
```

```
{ int n=0;
```

```
while (n!=6)
```

```
{ system("cls");
```

```
cout <<"          \n";
```

```
cout <<" |  欢迎使用  | \n";
```

```
cout <<" |  1. 追加记录  | \n";
```

```
cout <<" |  2. 统计分数  | \n";
```

```
cout <<" |  3. 全部显示  | \n";
```

```
cout <<" |  4. 查询成绩  | \n";
```

```
cout <<" |  5. 删除记录  | \n";
```

```
cout <<" |  6. 退出      | \n";
```

```
cout <<"          \n";
```

```
cout <<"请输入数据选择 1/2/3/4/5/6: " << endl;
```

```
cin >> n; cin.get();
```

```
switch(n)
```

```
{ case 1:append();break;
```

```
case 2:statistic();break;
```

```
case 3:listall();break;
```

```
case 4:search();break;
```

```
case 5:dele();break;
```

```
default:break;
```

```
}
```

```
}
```

```
return 0;
```

```
}
```

```
void append()
{
    student st;
    system("cls");
    ofstream FILE;
    FILE.open("data.txt",ios::app);
    if(!FILE)
    {
        cout << "Can't open the file" << endl;
        exit(1);
    }
}
```

FILE.open("data.txt",ios::app);

打开一个输出，在文件末尾添加新数据

for (;;)

```
{    cout << "Please input ID "; cin>>st.ID; cin.get();  
    if (st.ID == -1) break;  
    cout << "Please input name ";  
    cin.getline(st.name,10);  
    cout << "Please input age "; cin >> st.age;  
    cout << "please intput class ";  
    cin >> st.class1;  
    cout << "please input midterm, homework, final ";  
    cin >> st.middle >> st.hw >> st.final;  
    st.score =  
        st.middle * 0.2 + st.hw * 0.1 + st.final * 0.7;  
    st.tag='1';  
    FILE.write((char *)&st,sizeof(student));  
}  
FILE.close();}
```

```
void listall()
{ system("cls"); student st;
  ifstream FILE("data.txt");
  if(!FILE)
  {cout << "Can't open the file" << endl;exit(1);}
  cout<< "学号 姓名 年龄 班级 期中 作业 期末 综合" << endl;
  cout<< "===== " << endl;
  while(!FILE.eof())
  { if (FILE.read((char *)&st,sizeof(student)))
    { if(st.tag == '1')
      { cout<<setw(5)<<st.ID<<setw(12)<<st.name<<setw(4)
        <<st.age<<setw(6)<<st.class1<<setw(8)<<fixed
        <<setprecision(2)<<st.middle <<setw(6)<<st.hw
        <<setw(6)<<st.final<<setw(6)<<st.score<<endl;
          cout << "-----"<<endl;
        } } }
  cin.get();  FILE.close();
}
```

```
#include <iostream> #include <cstring>
#include <fstream> #include <iomanip>
#include "statement.h"
using namespace std;
void update()
{  student st; int ID,count = 0;
   char ch;    char temp[11];

   fstream FILE("data.txt",ios::in|ios::out);

   if(!FILE)
   {    cout << "Can't open the file" << endl;
        exit(1);
   }
   cout << "输入要修改的学号( -1 exit) ";
   cin >> ID; cin.get();
```

```
while(ID != -1)  
{ FILE.seekp(0);  
  while (!FILE.eof())  
    {if (FILE.read((char *)&st,sizeof(student)))  
      { count ++;  
        if ((st.ID == ID) && (st.tag == '1'))  
          { //显示记录  
            cout << "confirm to update (Y/N) " ;  
            cin >> ch;cin.get();
```

```
if (ch == 'y' || ch == 'Y')
{ cout << "Please input ID ";
  cin.getline(temp,11);
  if (temp[0] != '\0') st.ID = atoi(temp);
  cout << "Please input name " ;
    cin.getline(temp,11);
    if (temp[0] != '\0') strcpy(st.name,temp);
  cout << "Please input age ";
    cin.getline(temp,11);
    if (temp[0] != '\0') st.age = atoi(temp);
  cout << "please input class ";
    cin.getline(temp,11);
    if (temp[0] != '\0') st.class1 = atoi(temp);
  cout << "please input midterm : ";
    cin.getline(temp,11);
    if (temp[0] != '\0') st.middle = atoi(temp);
  cout << "please input homework : ";
    cin.getline(temp,11);
```



```
    if (temp[0] != '\0') st.hw = atoi(temp);  
    cout << "please input final : "  
    cin.getline(temp,11);  
    if (temp[0] != '\0') st.final = atoi(temp);  
    st.score=st.middle*0.2+st.hw*0.1+st.final*0.7;    st.tag =  
'1';  
    FILE.seekp((count - 1) * sizeof(student));  
    FILE.write((char *)&st,sizeof(student));  
    break;  
}
```

} (是修改)

}(是正确到一个记录)

} (没到文件尾)

```
if (count == 0 )
```

```
{ cout << "not found" << endl; }
```

```
cout << "输入要修改的学号: (-1 exit)";
```

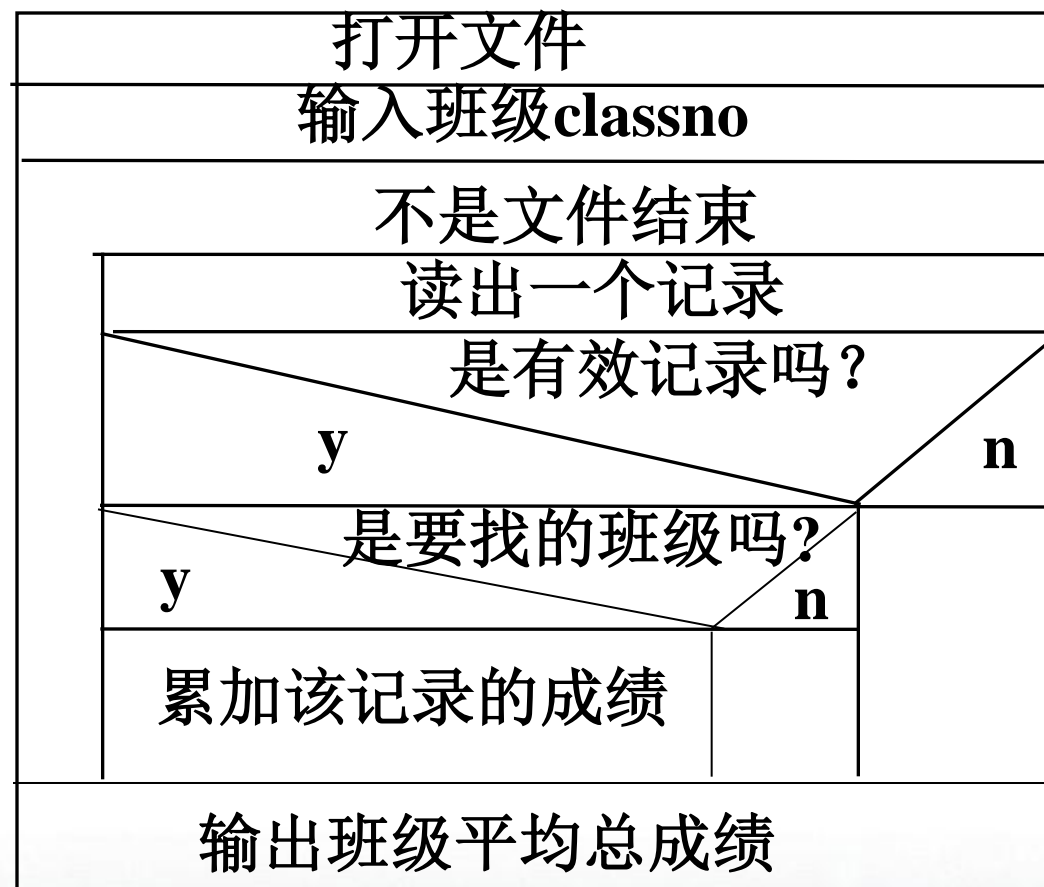
```
cin >> ID; cin.get();
```

```
} (继续修改, -1 exit)
```

```
FILE.close();}
```



统计成绩

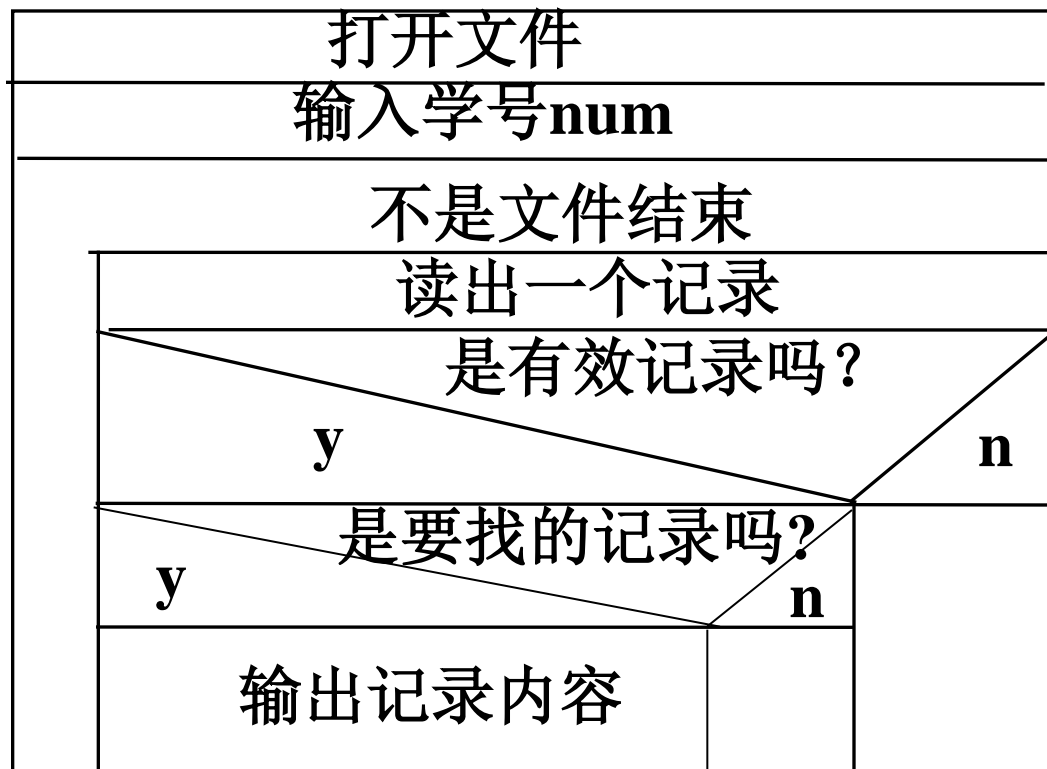


```
void statistic()  
{  
    student st;  
    int classno,count=0;  
    float sum = 0.0, aver = 0.0;  
    ifstream FILE("data.txt");  
    if(!FILE)  
    {  
        cout << "Can't open the file" << endl;  
        exit(1);  
    }  
}
```

```
cout << "which class to be calculated:" ;
cin >> classno;
while(!FILE.eof())
{
    if (FILE.read((char *)&st,sizeof(student)))
    {
        if((st.tag=='1') && (st.class1==classno))
            { count++; sum += st.score; }
    }
}
aver = sum / count;
cout << " The class " << classno
    << "'s average is " << fixed << setprecision(2)
    << aver << endl;
cin.get();
FILE.close();
}
```



查询记录



```
void search()  
{  student st;  
    int choice,ID = -1 ,count=0, classno = -1,flag = 2;  
    char name[10]=" ";  
  
    ifstream FILE("data.txt");  
if(!FILE)  
    {    cout << "Can't open the file" << endl;  
        exit(1);  
    }  
    cout << "by which key do you want to search " << endl;  
    cout << "1--by ID" << endl;  
    cout << "2--by name" << endl;  
    cout << "3--by class" << endl;  
    cin >> choice; cin.get();
```

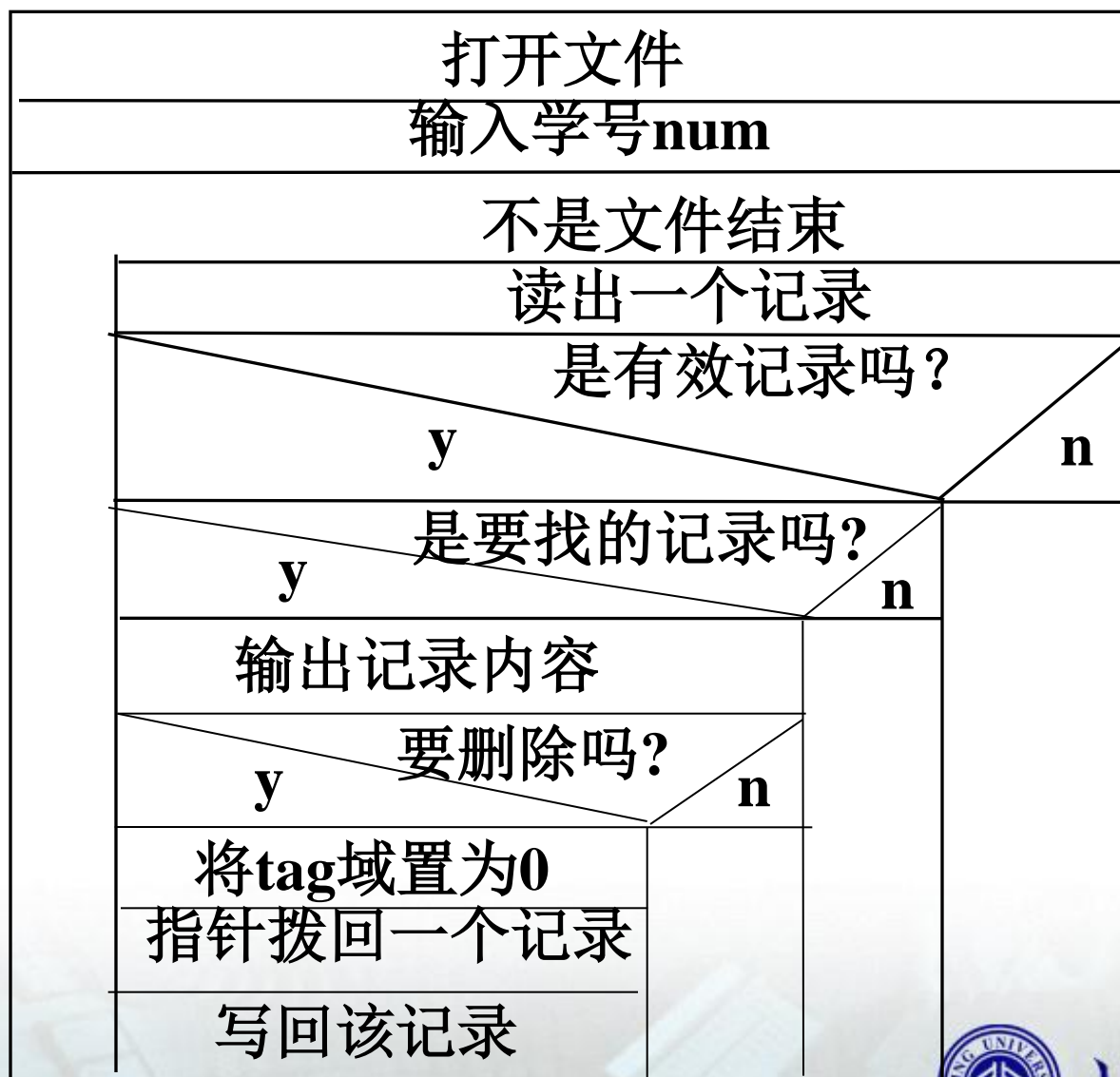
```
switch(choice)
{
    case 1 :
        cout << "please input a ID " ;
        cin >> ID; break;
    case 2 :
        cout << " please input a name ";
        cin.getline(name,10); break;
    case 3 :
        cout << " please input class ";
        cin >> classno; break;
    default:
        cout << "error" << endl;
}
cout<<“学号 姓名 年龄 班级 期中 作业 期末 综合 ” <<endl;
cout<< "===== " <<endl;
```



```
while(!FILE.eof())
{  if (FILE.read((char *)&st,sizeof(student)))
    {
        if(st.tag=='1')
        {  count ++;
            flag = strcmp(st.name,name);
            if (!flag||st.ID==ID||st.class1==classno)
            {
                显示记录;
                cout<< "----- "
                << endl;
            }
        }
    }
}
if (count ==0) cout << " not found" << endl;
cin.get(); cin.get();FILE.close();}
```



删除记录



北京大学

```
void dele()  
{  
    student st;      int ID,count = 0;   char ch;  
  
    fstream FILE("data.txt",ios::in|ios::out);  
  
    if(!FILE)  
    {  
        cout << "Can't open the file" << endl;  
        exit(1);  
    }  
  
    cout << "输入要删除的学号(-1 exit): ";  
    cin >> ID; cin.get();
```

```
while(ID != -1)
{ FILE.seekp(0);
  while (!FILE.eof())
  { if (FILE.read((char *)&st,sizeof(student)))
    { count ++;
      if ((st.ID == ID) && (st.tag == '1'))
      { 显示记录;    }
      cout << "confirm (Y/N) ";cin >> ch;cin.get();
      if (ch == 'y' || ch == 'Y')
      { st.tag='0';
        FILE.seekp((count - 1) * sizeof(student));
        FILE.write((char *)&st,sizeof(student)); break;
      } }
    }
  if (count == 0 )
  { cout << "not found" << endl;}
  cout<<"输入要删除的学号: "; cin >> ID; cin.get();
}
FILE.close();}
```



好好想想，有没有问题？

谢谢！



清华大学