一、ASCII文件 输出

　　为了使用下面的方法, 你必须包含头文件 <fstream.h>(在标准C++中，已经使用取代< fstream.h>，所有的C++标准头文件都是无后缀的。)这是 <iostream.h>的一个扩展集, 提供有缓冲的文件输入输出操作. 事实上, <iostream.h> 已经被<fstream.h>包含了。我们从文件操作类的设计开始, 我会讲解如何进行ASCII I/O操作。 这篇文章介绍的方法,我们分别使用”ifstream”和 “ofstream” 来作输入输出。如果你用过标准控制台流”cin” 和 “cout,” 那现在的事情对你来说很简单。 我们现在开始讲输出部分，首先声明一个类对象。  
ofstream fout; 这就可以了，不过你要打开一个文件的话, 必须像这样调用 ofstream::open()。

fout.open(“output.txt”); 你也可以把文件名作为构造参数来打开一个文件.

ofstream fout(“output.txt”); 这是我们使用的方法, 因为这样创建和打开一个文件看起来更简单. 顺便说一句, 如果你要打开的文件不存在，它会为你创建一个, 所以不用担心文件创建的问题. 现在就输出到文件，看起来和”cout”的操作很像。对不了解控制台输出”cout”的人, 这里有个例子。

int num = 150;

char name[] = "John Doe";

cout << "Here is a number: " << num << "/n";

cout << "Now here is a string: " << name << "/n";

保存文件，你必须关闭文件，或者回写文件缓冲.  
1.文件关闭之后就不能再操作了, 所以只有在你不再操作这个文件的时候才调用它，它会自动保存文件。2.回写缓冲区会在保持文件打开的情况下保存文件, 所以只要有必要就使用它。 回写看起来像另一次输出, 然后调用方法关闭。像这样：

fout << flush; fout.close(); 现在你用文本编辑器打开文件，内容看起来是这样：

Here is a number: 150 Now here is a string: John Doe 很简单吧! 现在继续文件输入, 需要一点技巧, 所以先确认你已经明白了流操作，对 “<<” 和”>>” 比较熟悉了, 因为你接下来还要用到他们。

二、ASCII 文件输入

输入和”cin” 流很像. 和刚刚讨论的输出流很像, 但你要考虑几件事情。在我们开始复杂的内容之前, 先看一个文本：  
12 GameDev 15.45 L This is really awesome! 为了打开这个文件，你必须创建一个in-stream对象

ifstream fin(“input.txt”); 现在读入前四行. 你还记得怎么用”<<” 操作符往流里插入变量和符号吧？好,在 “<<” (插入)操作符之后，是”>>” (提取) 操作符. 使用方法是一样的. 看这个代码片段.

int number;

float real;

char letter, word[8];

fin >> number; fin >> word; fin >> real; fin >> letter; 也可以把这四行读取文件的代码写为更简单的一行。

fin >> number >> word >> real >> letter; 它是如何运作的呢? 文件的每个空白之后, “>>” 操作符会停止读取内容, 直到遇到另一个>>操作符. 因为我们读取的每一行都被换行符分割开(是空白字符), “>>” 操作符只把这一行的内容读入变量。这就是这个代码也能正常工作的原因。但是，可别忘了文件的最后一行。

This is really awesome! 如果你想把整行 读入一个char数组, 我们没办法用”>>”操作符，因为每个单词之间的空格（空白字符）会中止文件的读取。但是有读取整行的方法, 它就是 getline()这就是我们要做的。

fin.getline(sentence, 100); 这是函数参数. 第一个参数显然是用来接受的char数组. 第二个参数是在遇到换行符之前，数组允许接受的最大元素数量. 现在我们得到了想要的结果：“This is really awesome!”。

读文件相关函数  
方法get() 每次返回一个字符。  
方法ignore(int,char) 跳过一定数量的某个字符, 但你必须传给它两个参数。第一个是需要跳过的字符数。 第二个是一个字符, 当遇到的时候就会停止。 例子,

fin.ignore(100, ‘/n’); 会跳过100个字符，或者不足100的时候，跳过所有之前的字符，包括 ‘/n’。  
方法peek() 返回文件中的下一个字符, 但并不实际读取它。所以如果你用peek() 查看下一个字符, 用get() 在peek()之后读取，会得到同一个字符, 然后移动文件计数器。  
方法putback(char) 输入字符, 一次一个, 到流中。我没有见到过它的使用，但这个函数确实存在。

如果是个很大的多行存储的文本型文件可以这么读：

char buf[1024]; //临时保存读取出来的文件内容

string message;

ifstream infile;

infile.open("myfile.js");

if(infile.is\_open()) //文件打开成功,说明曾经写入过东西

{

while(infile.good() && !infile.eof())

{

memset(buf,0,1024);

infile.getline(buf,1204);

message = buf;

...... //这里可能对message做一些操作

cout<<message<<endl;

}

infile.close();

}

你应该已经知道如何读取和写入ASCII文件了。但我们还不能罢休，因为二进制文件还在等着我们。

三、二进制 输入输出

二进制文件会复杂一点, 但还是很简单的。 首先你要注意我们不再使用插入和提取操作符(译者注：<< 和 >> 操作符). 你可以这么做，但它不会用二进制方式读写。你必须使用read() 和write() 方法读取和写入二进制文件. 创建一个二进制文件, 看下一行。

ofstream fout(“file.dat”, ios::binary); 这会以二进制方式打开文件, 而不是默认的ASCII模式。首先从写入文件开始。函数write() 有两个参数。 第一个是指向对象的char类型的指针, 第二个是对象的大小（译者注：字节数）。 为了说明，看例子。

int number = 30; fout.write((char \*)(&number), sizeof(number)); 第一个参数写做”(char *)(&number)”. 这是把一个整型变量转为char*指针。如果你不理解，可以立刻翻阅C++的书籍，如果有必要的话。第二个参数写作”sizeof(number)”. sizeof() 返回对象大小的字节数. 就是这样!  
二进制文件最好的地方是可以在一行把一个结构写入文件。 如果说，你的结构有12个不同的成员。 用ASCII?文件，你不得不每次一条的写入所有成员。 但二进制文件替你做好了。 看这个。

struct OBJECT { int number; char letter; } obj;

obj.number = 15;

obj.letter = ‘M’;

fout.write((char \*)(&obj), sizeof(obj));

这样就写入了整个结构! 接下来是输入. 输入也很简单，因为read()?函数的参数和 write()是完全一样的, 使用方法也相同。

ifstream fin("file.dat", ios::binary); fin.read((char \*)(&obj), sizeof(obj)); 我不多解释用法, 因为它和write()是完全相同的。二进制文件比ASCII文件简单, 但有个缺点是无法用文本编辑器编辑。 二进制文件使用二进制保存，打开后显示的是乱码，需要使用BinaryViewer打开。而ASCII文件显示的就是你想要的，和cout<< 一样。

写文件相关函数  
只有一个你可能会关注的方法.那就是 put(char), 它每次向输出流中写入一个字符。

四、更多方法

检查文件  
你已经学会了open() 和close() 方法, 不过这里还有其它你可能用到的方法。  
方法good() 返回一个布尔值，表示文件打开是否正确。  
类似的，bad() 返回一个布尔值表示文件打开是否错误。 如果出错，就不要继续进一步的操作了。  
最后一个检查的方法是fail(), 和bad()有点相似, 但没那么严重。

五、C++文件的读取和写入程序举例：  
C++文件的读取和写入程序举例1：

#include <iostream>

#include <iomanip>

#include <fstream>

using namespace std;

int main(){

char buffer[256];

ifstream myfile ("c:\\a.txt");

ofstream outfile("c:\\b.txt");

if(!myfile){

cout << "Unable to open myfile";

exit(1); // terminate with error

}

if(!outfile){

cout << "Unable to open otfile";

exit(1); // terminate with error

}

int a,b;

int i=0,j=0;

int data[6][2];

while (! myfile.eof() )

{

myfile.getline (buffer,10); //读入每行

sscanf(buffer,"%d %d",&a,&b);

cout<<a<<" "<<b<<endl;

data[i][0]=a;

data[i][1]=b;

i++;

}

myfile.close();

for(int k=0;k<i;k++)

{

outfile<<data[k][0] <<" "<<data[k][1]<<endl;

cout<<data[k][0] <<" "<<data[k][1]<<endl;

}

outfile.close();

return 0;

}

C++文件的读取和写入程序举例2：

/\*/从键盘读入一行字符，把其中的字母依次放在磁盘文件fa2.dat中，再把它从磁盘文件读入程序，  
将其中的小写字母改成大写字母，再存入磁盘fa3.dat中\*/

＃i nclude<fstream>

＃i nclude<iostream>

＃i nclude<cmath>

using namespace std;

//从键盘上读取字符的函数

void read\_save(){

char c[80];

ofstream outfile("f1.dat");//以输出方式打开文件

if(!outfile){

cerr<<"open error!"<<endl;//注意是用的是cerr

exit(1);

}

cin.getline(c,80);//从键盘读入一行字符

for(int i=0;c[i]!=0;i++) //对字符一个一个的处理，直到遇到'/0'为止

if(c[i]>=65&&c[i]<=90||c[i]>=97&&c[i]<=122){//保证输入的字符是字符

outfile.put(c[i]);//将字母字符存入磁盘文件

cout<<c[i]<<"";

}

cout<<endl;

outfile.close();

}

void creat\_data(){

char ch;

ifstream infile("f1.dat",ios::in);//以输入的方式打开文件

if(!infile){

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

exit(1);

}

ofstream outfile("f3.dat");//定义输出流f3.dat文件

if(!outfile){

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

exit(1);

}

while(infile.get(ch)){//当读取字符成功时

if(ch<=122&&ch>=97)

ch=ch-32;

outfile.put(ch);

cout<<ch;

}

cout<<endl;

infile.close();

outfile.close();

}

int main(){

read\_save();

creat\_data();

system("pause");

return 0;

}

C++文件的读取和写入程序举例3：  
Example : 从in.txt　文件中读入数据，并输出到out.txt中,输出的文件中每行之间有一空行相隔

#include

#include

#include

using namespace std;

int main (int){

string s;

ifstream inf;

inf.open("in.txt");

//打开输出文件

ofstream outf;

outf.open("out.txt");

//从in.txt　文件中读入数据，并输出到out.txt中

/\*其中 getline(参1,参2);　作用是从 inf 指向的文件中

每次读入一行，把数据存到字符串s中，从第一行开始

　每读完一行后，系统自动地把指针指向下一行，不用人为

　干预\*/

while( getline(inf,s ) ){

outf << s << '\n\n'; //我这里并没有用到字符串

cout << s << endl << endl; //数组，而是只用了一个串

} //S，是因为我每次读入一行

//后，立即就把它输出到

//out.txt中，跟着读下一行

inf.close();

outf.close();

return 0;

}

=============== 方法二 ================

#include

#include

#include

#include

using namespace std;

int main (int){

ifstream inf;

inf.open("in.txt");

ofstream outf;

outf.open("out.txt");

/\*这道题有许多解法的，重要的要了它文件输入输出的原理

　你可以一行行地读入，也可以一个字一个字地读入，或一个词

　一个词地读入，整型或浮点型读入，看你定义的是哪种数据类型\*/

char c;

inf >> noskipws; //不忽略空白，把每行最后那个'\n'

//也读进来。

while(inf >>c)

{

if (c == '\n'){ //遇到 '\n'回车、换行。

outf << "\n\n"; //输出到文件

cout << "\n\n"; //输出到屏幕

}

else{

outf << c; //输出到文件

cout << c; //输出到屏幕

}

}

/\* 同样的原理，从文件中读入单个字符，每次读入一个后，

　　系统自动地把指针指向下一个字，而不用你指定这次读哪个，

下次读哪个，除非你不想从第一个开始读，比如说：我想从

第100个字开始读，或者我想读最后50个字。这就需要调用

相应的函数，并指定相应的位置。\*/

inf.close();

outf.close();

return 0;

}

六、C++文件的读取和写入讨论：

主　　题1： 请问VC++中,怎么实现读取.txt文件时的行定位,就是读取指定的行的内容.

一行一行的读到你想要的那行。  
CStdioFile的ReadString()方法是读一行。

文件对话框读写文本文件/***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****  
\* 读文本文件  
***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****/  
//显示文件打开对话框  
CFileDialog dlg(TRUE, “SQL”, “\*.txt”,OFN\_HIDEREADONLY  
|OFN\_OVERWRITEPROMPT,”Text Files(*.txt)|*.txt|SQL Files(*.sql)|*.sql|All Files(*.*)|*.*||”);  
if ( dlg.DoModal()!=IDOK ) return;  
//获取文件的绝对路径  
CString sFileName=dlg.GetPathName();  
//打开文件  
CStdioFile out;  
out.Open(sFileName, CFile::modeRead);  
CString sSql=”“,s;  
//读取文件  
do{  
out.ReadString(s);  
sSql=sSql+s+”\r\n”;  
}while ( out.GetPosition()out.Close();  
//AfxMessageBox(sSql);

/***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****  
\* 写文本文件  
***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****/  
//显示文件保存对话框  
CFileDialog dlg(FALSE, “SQL”, “\*.txt”,OFN\_HIDEREADONLY  
| OFN\_OVERWRITEPROMPT,”Text Files(*.txt)|*.txt|SQL Files(*.sql)|*.sql|All Files(*.*)|*.*||”);  
if ( dlg.DoModal()!=IDOK ) return;  
//获取文件的绝对路径  
CString sFileName=dlg.GetPathName();  
CStdioFile out;  
//打开文件  
out.Open(sFileName, CFile::modeCreate | CFile::modeWrite);  
//保存文件  
CString sSql=”文本文件内容”;  
out.WriteString(sSql);  
out.Close();

主　　题2：我用CFile::Write,怎样才可以写入回车,空格?而且我要写两栏要对齐?  
回车换行：”\r\n”

主　　题3:如何正确的使用CFile::Open中的参数?

我设计了一个从记事本中读数据的程序。将数据显示在视中。  
代码如下：

void CTry1View::OnShow()  
{  
// TODO: Add your command handler code here  
CStdioFile file;

CString filename;

CFileDialog opendlg(true,NULL,NULL,OFN\_HIDEREADONLY|OFN\_OVERWRITEPROMPT,".(\*.txt)|\*.txt|All Files(\*.\*)|\*.\*||",NULL);

if(opendlg.DoModal()==IDOK)

filename=opendlg.GetPathName();

if(file.Open(filename,CFile::modeCreate|CFile::modeReadWrite|CFile::typeText)==0)

{

AfxMessageBox("error");

return;

}

while(file.ReadString(string))

{

strList.AddTail(string);

string.ReleaseBuffer();

}

flag=true;

Invalidate();

}  
结果不但在视中没有任何显示，而且记事本中的数据也全部丢失。变成了一片空白。真是搞不懂了。  
记事本中的数据是我随便写的。如下：  
11  
222  
3333  
44444  
…….

[解决方法]  
在file.Open(filename,CFile::modeCreate|CFile::modeReadWrite|CFile::typeText)中,CFile::modeCreate去掉,modeCreate的意思是没有此文件就建立,有了此文件,清空文件.

最好是这样file.Open(filename,CFile::modeCreate|CFile::modeNoTruncate|CFile::modeReadWrite|CFile::typeText)，因为CFile::modeNoTruncate意思是文件即使存在也不清空。

主　　题4:如何使用CStdioFile::ReadString来读文件?  
一数据文件一行一条记录,我用file.readstring()一次读一行,并对读取的数据做一些处理  
请问:while(文件还没到结尾)  
{  
file.readstring();  
…  
}  
文件还没到结尾如何判断?  
如果,到了指定位置不读了,过一会儿再读又如何做?  
[解决方法]  
while()中不用另加判断条件了,因为CStdioFile::ReadString()本身就是判断标志,若没有了(文件到头)返回NULL,因此:while(file.ReadString(s)){}就可.返回NULL的条件是遇到文件尾或者一个空行  
[程序实现]  
假设你已有了名为ts.txt的文件在你的工程目录下:  
{  
CStdioFile file;  
CString sss;  
char ccc[100];  
DWORD o=0;  
int ol=0;  
file.Open(“ts.txt”,CFile::modeRead);  
while(file.ReadString(sss))  
{  
ol++;  
if(ol>1)//读两次就不读了.  
break;

}  
o=file.GetPosition();//记录上次的结果(读到哪了)  
……………..  
file.Seek(o,0);//接着上回读  
while(file.ReadString(sss))  
{  
strcpy(ccc,sss);  
AfxMessageBox(ccc);  
}  
}

　　VC＋＋只提供了删除一个空目录的函数，而在实际应用中往往希望删除其下有很多子目录与文件的目录。为了实现这一功能，我编写了DeleteDirectory函数，它可以实现这一功能。

函数原型：BOOL DeleteDirectory(char \*DirName);  
返回值：成功删除时返回TRUE，否则返回FALSE  
参数DirName为要删除的目录名，必须为绝对路径名，如“c:\temp”。

函数定义如下：

BOOL DeleteDirectory(char \*DirName)

{

CFileFind tempFind;

char tempFileFind[200];

sprintf(tempFileFind,"%s\\\*.\*",DirName);

BOOL IsFinded=(BOOL)tempFind.FindFile(tempFileFind);

while(IsFinded)

{

IsFinded=(BOOL)tempFind.FindNextFile();

if(!tempFind.IsDots())

{

char foundFileName[200];

strcpy(foundFileName,tempFind.GetFileName().GetBuffer(200));

if(tempFind.IsDirectory())

{

char tempDir[200];

sprintf(tempDir,"%s\\%s",DirName,foundFileName);

DeleteDirectory(tempDir);

}

else

{

char tempFileName[200];

sprintf(tempFileName,"%s\\%s",DirName,foundFileName);

DeleteFile(tempFileName);

}

}

}

tempFind.Close();

if(!RemovwDirctory(DirName))

{

MessageBox(0,"删除目录失败！","警告信息",MK\_OK);

return FALSE;

}

return TRUE;

}

关键词：文本文件按行读取查找特定字符串替换字符串

C/C++ VC/MFC

按行：fgets();getline(); StdioFile::ReadString(buf)CArchive::ReadString(CString &str)  
查找：strstr();\_tcschr();fseek() buf.Find(str)（buf是CString类型的）  
替换：替换字符串  
代码：  
CStringFile CLASS

我用fopen打开一个txt文件，想查找某一字符串，找到这个字符串后，再读取它对应行的  
其他数据，请教大侠指点小弟如何去做，用什么函数。谢谢。

ａ.　fopen过后，逐行读入str,用 strstr 判断是否含有特定字符串.例子：

FILE \*fd=fopen("你要打开的文件","r+");

char tmpLine[200];

while(fgets(tmpLine,200,fd)!=NULL)

{

if( strstr(tmpLine,"你要找的字符串"))

{

printf("find the string!\n");

break;

}

memset(tmpLine,0,200);

}

free(tmpLine);

//从指定文件中提取指定字符串

CStrOP& CStrOP::GetSpecifyString(LPCTSTR szSpec, CArchive \*ar, BOOL bMid, int iMid, int \*pRow, int \*pCol)

{

do

{

//读取文件内容

if(ar)

{

//判断是否到了文件最末

if(!ar->ReadString(\*this))

{

//未找到找到字符串

if(pRow)

{

\*pRow = 0;

}

if(pCol)

{

\*pCol = 0;

}

(\*this).Format("%s", "");

break;

}

}

//获得需要查找的字符串所在行

if(pRow)

{

(\*pRow)++;

}

//去掉空白字符

TrimLeft();

TrimRight();

//判断是否已经找到需要的字串

if(Find(szSpec) >= 0)

{

//获得需要查找的字符串所在列

if(pCol)

{

\*pCol = Find(szSpec) + 1;

}

//判断是否需要提取字符串

if(bMid)

{

//判断提取字符串的规则，如果iMid为－1，则不论字串在任何位置，都可以

//进行提取；否则，字串必须在指定位置iMid时才可以提取

if(iMid == -1)

{

(\*this).Format("%s", Mid(Find(szSpec) + lstrlen(szSpec)));

break;

}

else if(iMid >= 0)

{

if(Find(szSpec) == iMid)

{

(\*this).Format("%s", Mid(iMid + lstrlen(szSpec)));

break;

}

}

}

}

else

{

if(!ar)

{

//未找到找到字符串

if(pRow)

{

\*pRow = 0;

}

if(pCol)

{

\*pCol = 0;

}

(\*this).Format("%s", "");

break;

}

}

} while(1);

return \*this;

}

其中CStrOP是我新建的类，函数声明为  
CStrOP& GetSpecifyString(LPCTSTR szSpec, CArchive *ar = NULL, BOOL bMid = TRUE, int iMid = 0, int* pRow = NULL, int\* pCol = NULL);

这个函数能实现在指定文本格式的文件中进行特定字符串的查找、提取、定位等工作，其中参数CArchive ar初始化为：

CFile file（cMyTxtFile，CFile::modeRead）

CArchive ar(&file, CArchive::load)