```
// buffile.cc

#include "buffile.h"

BufferFile::BufferFile (IOBuffer & from)
 // create with a buffer
    : Buffer (from)
{
}

int BufferFile::Open (char * filename, int mode)
// open an existing file and check the header
// a correct header must be on the file
// use ios::nocreate to ensure that a file exists
{
    // these modes are not allowed when opening an existing file
    if (mode&ios::noreplace||mode&ios::trunc) return FALSE;

    File . open (filename, mode|ios::in|ios::nocreate);
    if (! File.good()) return FALSE;
    File . seekg(0, ios::beg); File . seekp(0, ios::beg);
    HeaderSize = ReadHeader();
    if (!HeaderSize) // no header and file opened for output
        return FALSE;
    if (!(ios::in & mode))
    {   // requested mode does not include input
        // close and reopen
        File . close();
        File . open (filename, mode|ios::nocreate);
    }
    File . seekp (HeaderSize, ios::beg);
    File . seekg (HeaderSize, ios::beg);
    return File . good();
}

int BufferFile::Create (char * filename, int mode)
// create a new file and write a header on it.
// use ios::nocreate to ensure that no file exists
{
    if (!(mode & ios::out)) return FALSE; // must include ios::out
    //File . open (filename, ios::out);
    File . open (filename, ios::out|ios::noreplace);
    cout << "open "<<filename<<" result "<<File.good()<<endl;
    if (!File . good())
    {
        File . close();
        return FALSE;
    }
    if (mode & ios::in)
    {// close and reopen the file
        File . close ();
        File . open (filename, mode|ios::nocreate);
    }
    if (!File . good()) return FALSE;
    HeaderSize = WriteHeader ();
    return HeaderSize != 0;
}

int BufferFile::Close ()
{
    File . close();
    return TRUE;
}

int BufferFile::Rewind ()
{
    File . seekg (HeaderSize, ios::beg);
    File . seekp (HeaderSize, ios::beg);
    return 1;
}

// Input and Output operations
int BufferFile::Read (int recaddr)
// read a record into the buffer
// return the record address
```

```
// return <0 if read failed
// if recaddr == -1, read the next record in the File
// if recaddr != -1, read the record at that address
{
    if (recaddr == -1)
        return Buffer . Read (File);
    else
        return Buffer . DRead (File, recaddr);
}

int BufferFile::Write (int recaddr)
 // write the current buffer contents
{
    if (recaddr == -1)
        return Buffer . Write (File);
    else
        return Buffer . DWrite (File, recaddr);
}

int BufferFile::Append ()
// write the current buffer at the end of File
{
    File . seekp (0, ios::end);
    return Buffer . Write (File);
}

// Access to IOBuffer
IOBuffer & BufferFile::GetBuffer ()
{ return Buffer;}

// protected methods
int BufferFile::ReadHeader ()
{
    return Buffer . ReadHeader (File);
}

int BufferFile::WriteHeader ()
{
    return Buffer . WriteHeader (File);
}
```