Stream States

- Every stream has a 'state' associated with it.
- Errors and non-standard conditions are handled by setting and testing this state appropriately.
- The stream state can be examined by operations on class ios.

Example :

```
fstream   file("Temp") ;
file.eof() ;      //end of file seen
file.fail() ;     //next operation will fail
file.bad() ;      //stream corrupted
file.good() ;     //next operation might succeed
```

Binary *read, write* to files : (The binary format is more accurate for storing numbers and always faster for saving data as there are no conversions involved in the process.)

```
#include <string.h>
#include <iostream>
#include <fstream>
class MyClass {
      int  intMem ; char     chMem ; char     chArr[5] ;
public :
       int  pubInt ; MyClass() {}
      MyClass(int i){  intMem = 1 ;chMem = 'w' ;strcpy(chArr, "good") ;
            pubInt = 20 ;}
      void f(){}   friend ostream &operator <<(ostream &s, MyClass c) ;} ;
ostream & operator << (ostream &s, MyClass c) {
      s<<"\nIntMem:"<<c.intMem ;s<<"\nCharMem:"<<c.chMem ;
      s<<"\nchArr : "<<c.chArr ;s<<"\npubInt : "<<c.pubInt<<endl ;
      return s ;}
int main() {   MyClass      o1(10), o2 ; cout<<o1 ;    cout<<o2 ;
      fstream       file("TEST", ios::in|ios::app) ;
      file.write((char *)&o1, sizeof(o1)) ;  file.flush() ;
      file.seekg(0,ios::beg) ;  file.read((char *)&o2, sizeof(o2)) ;
      cout<<o1 ;    cout<<o2 ;
      file.seekp(0,ios::end) ;      file<<o2 ;      file.close() ;}
```

A simple example :

```
#include <string.h>
#include <iostream>
#include <fstream>
using namespace std ;

int main() {
    char name[30] ; float cost ;

    ofstream out("TEST") ;
    cout<<"Enter name:" ;     cin>>name ;  out<<name<<endl ;
    cout<<"\nEnter cost:" ;   cin>>cost ;   out<<cost<<"\n" ;

    out.close() ;
    strcpy(name, "") ;      cost = 0 ;

    ifstream in("TEST") ;
    in>>name ;   in>>cost ;

    cout<<"\nName : "<<name ;
    cout<<"\nCost : "<<cost ;
}
```

❖ Functions for manipulating get pointer :

➢ seekg(offset , refposition)

- Moves get pointer to the specified location.
- Parameter offset represents no of bytes the file pointer is to be moved from the location specified by the parameter refposition
- refposition can be *ios::beg,ios::cur or ios::end*

➢ tellg()

- Gives the current position of get pointer

❖ Functions for manipulating put pointer :

➢ seekp(offset , refposition)

- Moves put pointer to the specified location.
- Parameter offset represents no of bytes the file pointer is to be moved from the location specified by the parameter refposition
- refposition can be *ios::beg,ios::cur or ios::end*

– tellp()

- Gives the current position of put pointer

File Pointers & Manipulations

- Each file has two file pointers associated with it
  - Input pointer (get pointer)
  - Output pointer (put pointer)
- Input pointer is used for reading the content from file
- Output pointer is used for writing to the file
- Pointer advances automatically on each input/output operation
- File stream classes provide functions for file pointer manipulation

❖File mode constants are defined in the class ios

❖Filemode parameter can take one or more filemode constants

- –ios::app Appends to end of file
- –ios::ate Go to end of file on opening
- –ios::in Open file for reading only
- –ios::nocreate Open fails if file does not exists
- –ios::noreplace Open fails if file already exists
- –ios::out Open file for writing only
- –ios::trunc Delete content of file if it exists

- open() member function can take one OR two arguments

- open(filename , filemode)
  - ➢ filename , specifies the file to be opened
  - ➢ filemode , specifies the purpose for which file is opened

- Prototype for open() contains default values for filemode argument
  - ➢ ios::in for ifstream functions meaning open for reading only
  - ➢ ios::out for ofstream functions meaning open for writing only

Opening A File

- For opening a file
  - Create the file stream object
  - Link the file stream with filename

- Two ways to open file
  - Using constructor function of the class
    - Create a file stream object to manage the stream using appropriate class
    - Initialize the file object with the desired filename

Example :

ofstream outfile("myfile.hcl") ;

This statement opens the file myfile.hcl and attaches it to output stream outfile

This method is more appropriate when we use only one file in the stream

- Using the member function open() of the class
  - Create a file stream object to manage the stream using appropriate class
  - Call open() method of stream object with the desired filename

    Example :

ofstream outfile ; // creates stream
outfile.open("myfile.hcl") ; // connects

This statement opens the file myfile.hcl and attaches it to output stream outfile

This method is used when we want to manage multiple files using one stream object.

Opening A File

- For opening a file
  - Create the file stream object
  - Link the file stream with filename

- Two ways to open file
  - Using constructor function of the class
    - Create a file stream object to manage the stream using appropriate class
    - Initialize the file object with the desired filename

Example :

ofstream outfile("myfile.hcl") ;

This statement opens the file myfile.hcl and attaches it to output stream outfile

This method is more appropriate when we use only one file in the stream

String Stream

- A stream can be attached to an array of characters in main memory.
- A rough sketch of the available classes is given below :
  - *istrstream*
    Contains open() with default input mode
    Provides input operations

  - *ostrstream*
    Contains open() with default output mode
    Provides output operations

  - *strstream*
    Contains open()
    Provides support for simultaneous input and output operations

Example :

```
#include <string.h>
#include <iostream>
#include <strstream>

class MyClass {
  int     intMem ; char      chMem ; char      chArr[5] ;
public :
   int    pubInt ; MyClass() {}
   MyClass(int i){  intMem = 1 ;chMem = 'w' ;strcpy(chArr, "good") ;
     pubInt = 20 ;}
   void f(){}   friend ostream &operator <<(ostream &s, MyClass c) ;} ;
ostream & operator << (ostream &s, MyClass c) {
   s<<"\nIntMem:"<<c.intMem ;s<<"\nCharMem:"<<c.chMem ;
   s<<"\nchArr : "<<c.chArr ;s<<"\npubInt : "<<c.pubInt<<endl ;
   return s ;}
main() {      MyClass      o1(40), o2 ; cout<<o1 ;    cout<<o2 ;
   char *p = new char[50] ;
   strstream       str(p, 50) ;
   str.write((char *)&o1, sizeof(o1)) ;
   cout<<o1 ;      cout<<o2 ;
   str.read((char *)&o2, sizeof(o2)) ;
   cout<<o2 ;}
```

Mishra - Outlook | Online Training | Online Test Window | Online C++ Compiler - online e

tests.mettl.com/v2/test-window/open/0/0?key=4a72723f

ercer | mettl | Mridul Mishra
LP_Practice_2ndWordUppercase / Saved: 30 seconds ago | Test Time: 00:24:31 | Finish

am | 1 | Attempted: 1/1

```
#include<stdio.h>
#include<string.h>
#include<bits/stdc++.h>
// Read only region start

char* secondWordUpperCase(char* input1)
{
    // Read
    // Write
    int size=strlen(input1);
    int i=0,j=0;
    char *a= new char[20];
    while(i<size && input1[i]!=' ')
    {
        ++i;
    }
    if(i==size)
    {
        return "LESS";
    }
    ++i;
    j=0;
    while(i<size && input1[i]!=' '){
        if(input1[i]>='a' && input1[i]<='z'){
            a[j]=input1[i]-'a'+'A';
        }
```

https://tests.mettl.com/v2/authenticateKey/4a72723f

Custom Input | Compile and Test | Submit C

Assessment © 2021-2031 | Need Help? Contact us: +1 (800) 265-6038 +91 80471-90902 | Powered By Merc

Mishra - Outlook | Online Training | Online Test Window | Online C++ Compiler - online e

tests.mettl.com/v2/test-window/open/0/0?key=4a72723f

ercer | mettl | Mridul Mishra
LP_Practice_2ndWordUppercase / Saved: 0 seconds ago
Test Time: 00:22:27 | Finish

am | 1 | Attempted: 1/1

```
{
    // Read only region end
    // Write code and remove the below exception.
    int size=strlen(input1);
    int i=0,j=0;
    char *a= new char[20];
    while(i<size && input1[i]!=' ')
    {
        ++i;
    }
    if(i==size)
    {
        return "LESS";
    }
    ++i;
    j=0;
    while(i<size && input1[i]!=' '){
        if(input1[i]>='a' && input1[i]<='z'){
            a[j]=input1[i]-'a'+'A';
        }
        else {
            a[j]=input1[i];
        }
        ++i;
        ++j;
    }
```

https://tests.mettl.com/v2/authenticateKey/4a72723f

Custom Input | Compile and Test | Submit C

ne Assessment © 2021-2031 | Need Help? Contact us: +1 (800) 265-6038 +91 80471-90902 | Powered By Merc