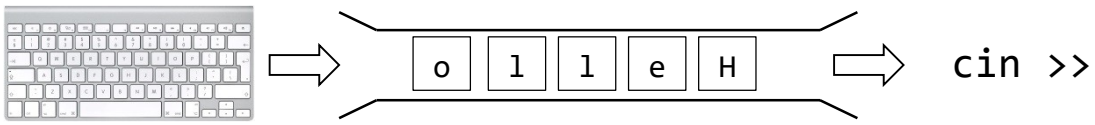
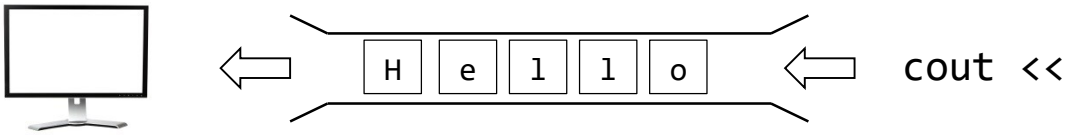
一、stream

我們可以把 C++裡的 stream 看成是資料流，裡面有字元資料在流動。

例如：iostream 裡的 cin 屬於 istream，通常和鍵盤連在一起。輸入 "Hello" 時就像這樣



cout 屬於 ostream，通常和螢幕連在一起。輸出 "Hello" 時就像這樣

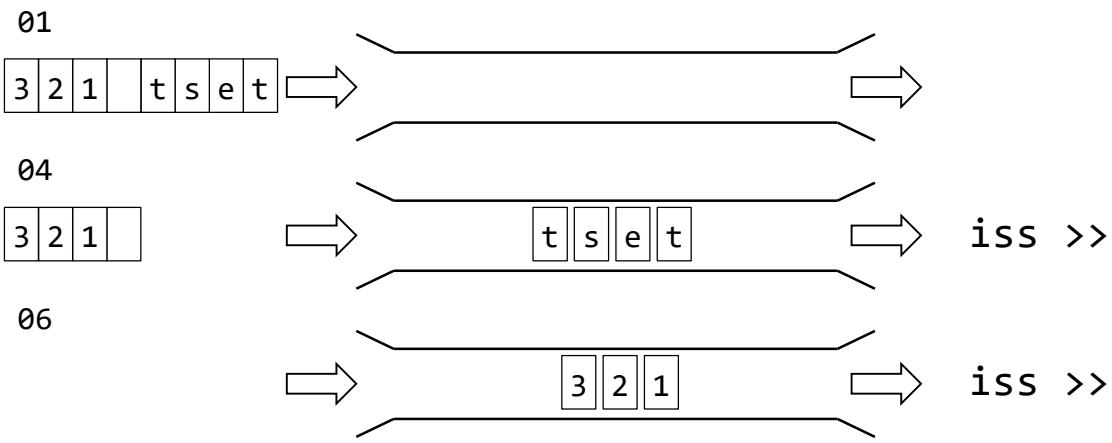


二、stringstream (需 #include <sstream>)

stringstream 裡的 istringstream、ostringstream 則是和一個字串連在一起。

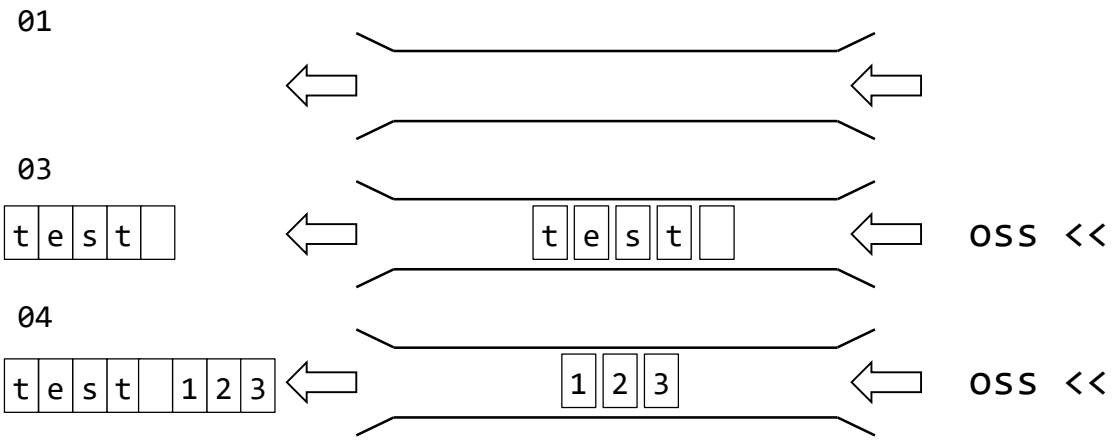
以 istringstream 為例，我們若在初始化時將它的內容初始為字串 "test 123"，便能像使用 cin 那樣來使用它，但當 "test 123" 被讀完後，裡面就空了，無法再被讀取。

```
01 istringstream iss("test 123");
02 int n;
03 string str;
04 iss >> str;
05 cout << str << endl;
06 iss >> n;
07 cout << str << endl;
```



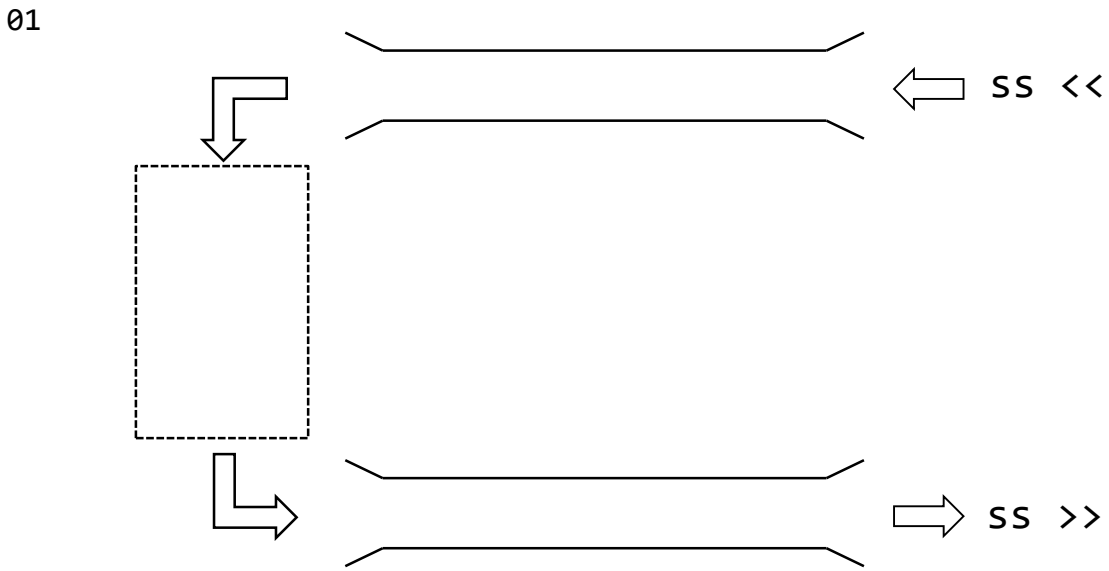
ostringstream 也是一樣，我們能像使用 cout 那樣來使用它。也可以把它裡前目前的字串用 str() 成員函數轉換為 string 物件。

```
01 ostringstream oss("");
02
03 oss << "test ";
04 oss << 123;
05 cout << oss.str() << endl;
```

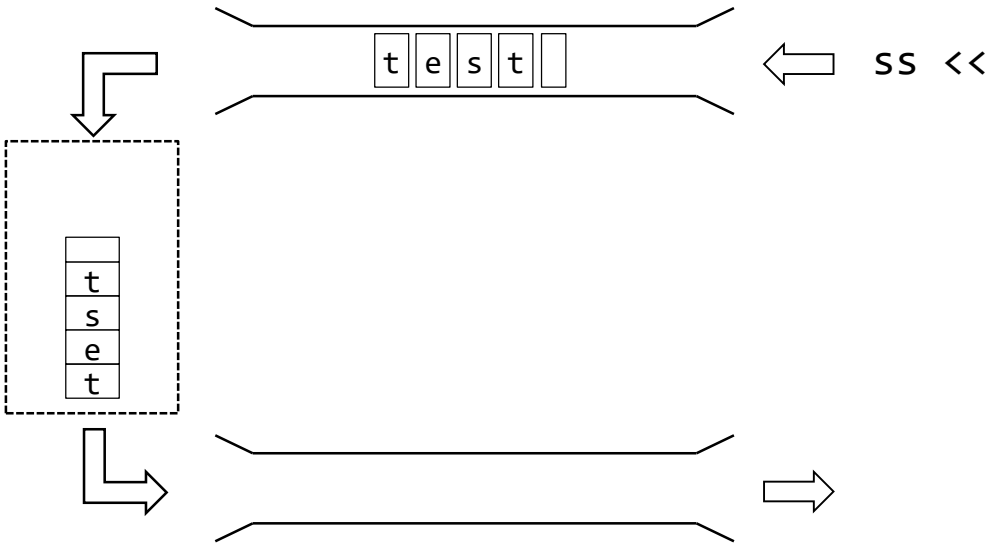


使用 stringstream (前面沒有 i 或 o)則能同時用來輸入與輸出。

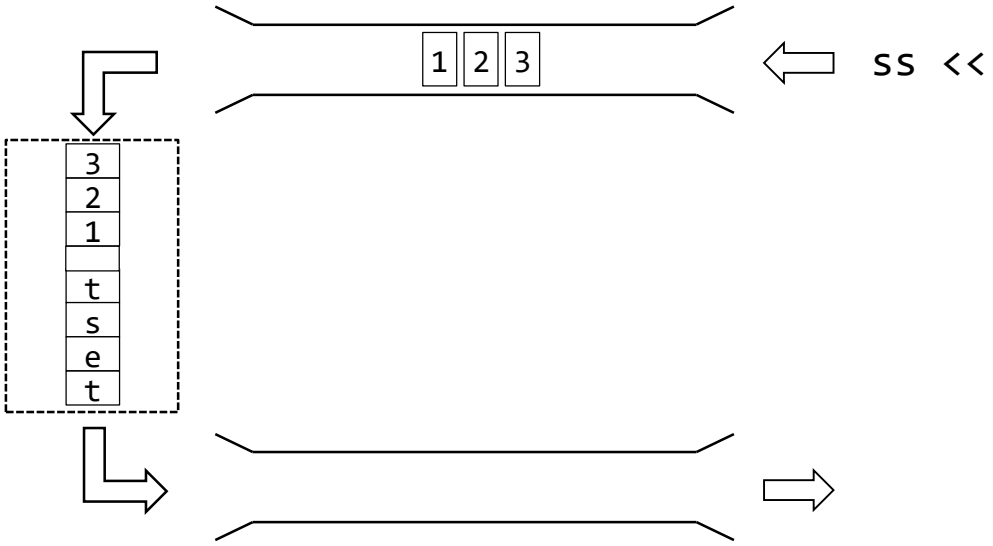
```
01 stringstream ss("");
02 int n;
03 string str;
04 ss << "test ";
05 ss << 123;
06 ss >> str >> n;
```



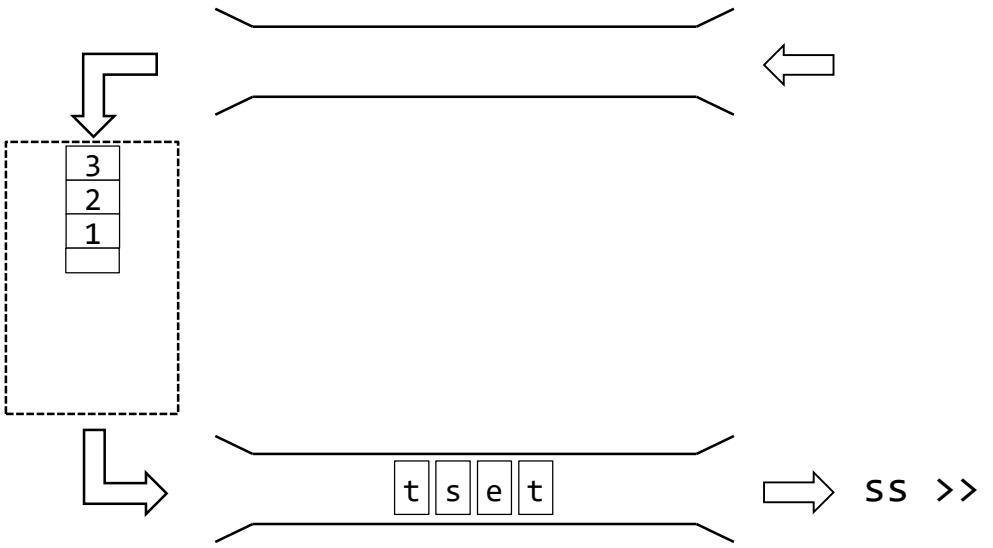
04

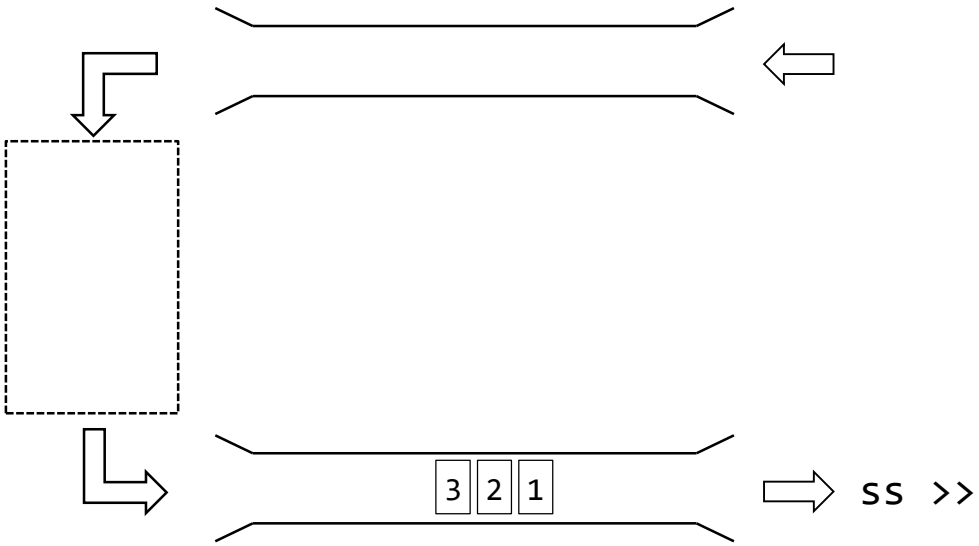


05



06





三、filestream (需 #include <fstream>)

類似 stringstream，但連結的目標是一個檔案。輸出時(<<)會將資料寫入檔案中，輸入時(>>)則會讀入檔案裡的資料。

（一）檔案處理可用的方式很多，我們先介紹使用 file stream 的作法。

namespace：
std
標頭檔：
<fstream>
物件類別：
fstream：處理檔案輸入與輸出
ofstream：處理檔案輸出
ifstream：處理檔案輸入

（二）建立一個 ifstream/ofstream 物件用來繫結至一個檔案以讀取/寫入其內容

```
ifstream 名稱(檔名);     // 檔名應使用 C 字串(c-style string)方式來表達
ofstream 名稱(檔名);
```

例：
```
ifstream infile("D:\\test.txt");
```

或

```
ifstream 名稱;
名稱.open(檔名);
ofstream 名稱;
名稱.open(檔名);
```

例：
```
ifstream infile;
infile.open("D:\\test.txt");
```

（三）判斷開檔動作是否成功

如果因為檔案不存在或其他因素造成這個動作失敗的話，該物件的值會是 NULL，所以我們可以藉此判斷其結果是否成功。

```
ifstream infile;
infile.open("test.txt");

if(!infile) {
    cout << "檔案開啟錯誤。" << endl;
    return -1;
}
```

（四）關閉檔案

檔案存取完畢後，記得要關閉它。

```
infile.close();
```

（五）開檔時的額外參數 – 開檔模式。

Flag value	說　　明
in	**in**put – 開啟為讀取模式。
out	**out**put – 開啟為寫入模式(若未同時指定 app，則刪除檔案所有資料)。
app	**app**end – 每次寫入時，將位置指針(indicator)移到檔案末端。
ate	**at e**nd – 開啟後馬上將位置指針(indicator)移到檔案末端。
trunc	**trunc**ate – 開啟後將所有資料刪除。
binary	**binary** – 以二進制模式進行存取。

有些函數的參數有預設值，例如 ofstream 的 open，其函數宣告原型如下。

```
void open ( const char * filename, ios_base:openmode mode = ios_base::out );
```

當我們不輸入第二個參數時，預設是以 out 模式開啟檔案。

```
outfile.open("test.txt");
```

等於

```
outfile.open("test.txt", ofstream::out);
```

若是要開啟一個檔案，但不想刪除其資料，只想在後面新增資料，可以加上 app 模式。

```
outfile.open("test.txt", ofstream::out|ofstream::app);
```

（六）檔案存取實例 – 保留最近十次登入紀錄

```
#include <iostream>
#include <fstream>
#include <ctime>

using namespace std;
```

```
void writeHistory(string *historyName, time_t *historyTime)
{
    ofstream outfile;
    // 開啟 history.txt 檔
    outfile.open("history.txt");
    if(!outfile) {
        cout << "無法寫入 history.txt。";
        return;
    }
    // 寫入登入歷史記錄
    for(int i=0; i<5; i++) {
        outfile << historyName[i] << "\t" << historyTime[i] << endl;
    }
    // 關檔
    outfile.close();
}
```

```
void showHistory(string *historyName, time_t *historyTime)
{
    ifstream infile;
    // 開啟 history.txt 檔
    infile.open("history.txt");


    if(!infile) {
        cout << "無法開啟 history.txt。重建一個新檔。" << endl;
        writeHistory(historyName, historyTime);
        // 嘗試重新開啟 history.txt 檔
        infile.open("history.txt");
    }

    int i=0;
    while(infile >> historyName[i] >> historyTime[i]) {
        cout << i+1 << "\t" << historyName[i] << "\t";
        if(historyTime[i] == 0) {
            // 若為 0 則輸出 none
            cout << "none" << endl;
        }else {
            // 否則將其轉換為字串輸出
            cout << ctime(&historyTime[i]);
        }
        i++;
    }
    // 關檔
    infile.close();
}
```

```
int main()
{
    string name;         // 姓名
    time_t loginTime;    // 登入時間

    string historyName[5];  // 登入歷史-姓名
    time_t historyTime[5];  // 登入歷史-時間

    // 初始化陣列內容
    for(int i=0; i<5; i++) {
        historyName[i] = "none";
        historyTime[i] = 0;
    }

    // 讀取並顯示登入歷史紀錄
    showHistory(historyName, historyTime);

    cout << "=================" << endl;
    cout << "Name: ";
    cin >> name;                 // 使用者輸入姓名
    loginTime = time(NULL);      // 取得目前時間

    // 移動並擠掉最舊的一個登入紀錄
    for(int i=4; i>0; i--) {
        historyName[i] = historyName[i-1];
        historyTime[i] = historyTime[i-1];
    }
    // 填入本次登入紀錄，並寫入歷史紀錄檔案
    historyName[0] = name;
    historyTime[0] = loginTime;
    writeHistory(historyName, historyTime);

    return 0;
}
```