Tutorial 3: File Input/Output

Overview

- Standard I/O and Files
- Using I/O Manipulators
- Why do we study character frequencies?

References

- Gary J. Bronson: C++ for Engineers and Scientists. 3rd Edition. Thomson (2010)
- Stanley B. Lippman, Josée Lajoie, and Barbara E. Moo: C++ Primer. 4th Edition. Addison-Wesley (2006)
- Bruno R. Preiss: Data Structures and Algorithms with Object-Oriented Design Patterns in C++. John Wiley & Sons, Inc. (1999)
- Gary J. Bronson: Object-Oriented Program Development Using C++ - A Class-Centered Approach. Thomson (2006)

File I/O

Recap: I/O Media

- Streams can be associated with
  - Physical devices (e.g., console - cin, cout)
  - Files (e.g., coefficients.txt, sales.dbf)
  - Structured storage (e.g., int values[10])

Chaining Input and Output

output - Notepad

File Edit Format View Help

```
Enter two numbers:
The sum of 7 and 12
```

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      cout << "Enter two numbers:" << endl;
7      int v1, v2;
8      cin >> v1 >> v2;
9      cout << "The sum of " << v1 << " and " << v2
10         << " is " << v1 + v2 << endl;
11     return 0;
12 }
```

14 Column: 1 Tab Size: 4 main

input - Notepad

File Edit Format View Help

```
7
12
```

Working With Files - Program Arguments

euclid.cpp

```cpp
#include <iostream>     // include standard IO library
#include <fstream>      // include file IO library

using namespace std;

int main( int argc, char* argv[] )
{
    if ( argc < 3 )
    {
        cerr << "Arguments missing" << endl;
        cerr << "Usage: euclid infile outfile" << endl;
        return 1;       // program failed
    }

    ...

    return 0;
}
```

Line: 62 Column: 1 C++ Tab Size: 4 distance

- We can pass the name of the files our program needs to work with through the command line arguments.

Opening an Input File

euclid.cpp

```
88  #include <iostream>     // include standard IO library
89  #include <fstream>      // include file IO library
90
91  using namespace std;
92
93  int main( int argc, char* argv[] )
94  {
95      ...
96
97      // set up input file
98      ifstream lInput;    // declare an input file variable (object)
99
100     lInput.open( argv[1], ifstream::in ); // open an input text file
101
102     if ( !lInput.good() )
103     {
104         // operation failed
105         cerr << "Cannot open input file " << argv[1] << endl;
106         return 2;       // program failed (input)
107     }
108
109     ...
110
111     return 0;
112 }
```

Always test whether operation succeeded

Line: 85 Column: 1 C++ Tab Size: 4 main

ifstream – C++ Reference

www.cplusplus.com/reference/fstream/ifstream/

The T-Syste...are Archive | German <->...U Chemnitz | Apple | Amazon | eBay | Yahoo! | News | Blackboard ...demic Suite

std::ifstream

<fstream>

Input file stream class

ifstream provides an interface to read data from files as input streams.

The objects of this class maintain internally a pointer to a filebuf object that can be obtained by calling member rdbuf.

The file to be associated with the stream can be specified either as a parameter in the constructor or by calling member open.

After all necessary operations on a file have been performed, it can be closed (or disassociated) by calling member close. Once closed, the same file stream object may be used to open another file.

The member function is_open can be used to determine whether the stream object is currently associated with a file.

Public members

(constructor)	Construct object and optionally open file (public member function)
rdbuf	Get the associated filebuf object (public member function)
is_open	Check if a file is open (public member function)
open	Open file (public member function)
close	Close file (public member function)

Members inherited from istream

operator>>	Extract formatted input (public member function)
gcount	Get character count (public member function)
get	Get characters (public member function)
getline	Get line (public member function)
ignore	Extract and discard characters (public member function)
peek	Peek next character (public member function)

ifstream::open – C++ Reference

www.cplusplus.com/reference/fstream/ifstream/open/

The T-Syste...are Archive | German <->...U Chemnitz | Apple | Amazon | eBay | Yahoo! | News | Blackboard ...demic Suite

Reference

- *C library:*
- *Containers:*
- *Input/Output:*
  - <fstream>
  - <iomanip>
  - <ios>
  - <iosfwd>
  - <iostream>
  - <istream>
  - <ostream>
  - <sstream>
  - <streambuf>
- *Other:*

<fstream>

- *classes:*
  - filebuf
  - fstream
  - ifstream
  - ofstream

ifstream

ifstream::ifstream

- *member functions:*
  - ifstream::close
  - ifstream::is_open
  - ifstream::open
  - ifstream::rdbuf

public member function

std::ifstream::open

<fstream>

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

Open file

Opens a file whose name is *filename*, associating its content with the stream object to perform input/output operations on it. The operations allowed and some operating details depend on parameter *mode*.

The function effectively calls rdbuf()->open(filename,mode).

If the object already has a file associated (open), the function fails.

On failure, the `failbit` flag is set (which can be checked with member fail), and depending on the value set with exceptions an exception may be thrown.

Parameters

filename
: C-string containing the name of the file to be opened.

mode
: Flags describing the requested i/o mode for the file. This is an object of type ios_base::openmode, which consists on a combination of one or more of the following flags defined as member constants:

flag value	opening mode
app	(**app**end) Set the stream's position indicator to the end of the stream before each output operation.
ate	(**at** end) Set the stream's position indicator to the end of the stream on opening.
binary	(**binary**) Consider stream as binary rather than text.
in	(**in**put) Allow input operations on the stream.
out	(**out**put) Allow output operations on the stream.
trunc	(**trunc**ate) Any current content is discarded, assuming a length of zero on opening.

Return Value

none

Example

ifstream::open - C++ Reference

www.cplusplus.com/reference/fstream/ifstream/open/

Example

```
// print the content of a text file.
#include <iostream>
#include <fstream>
using namespace std;

int main () {

  ifstream infile;

  infile.open ("test.txt", ifstream::in);

  int ch = infile.get();
  while (infile.good()) {
    cout << (char) ch;
    ch = infile.get();
  }

  infile.close();

  return 0;
}
```

This example opens a file and prints out its content.

Basic template member declaration

(basic_ifstream<charT,traits>)

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

See also

ifstream::close	Close file (public member function)
ifstream::ifstream	Construct object and optionally open file (public member function)

Opening an Output File

euclid.cpp

```
113 #include <iostream>      // include standard IO library
114 #include <fstream>       // include file IO library
115
116 using namespace std;
117
118 int main( int argc, char* argv[] )
119 {
120     ...
121
122     // set up output file
123     ofstream lOutput;    // declare an output file variable (object)
124
125     lOutput.open( argv[2], ofstream::out ); // open an output text file
126
127     if ( !lOutput.good() )
128     {
129         // operation failed
130         cerr << "Cannot open output file " << argv[2] << endl;
131         lInput.close(); // never forget, we must close input file
132         return 3;       // program failed (output)
133     }
134
135     ...
136
137     return 0;
138 }
```

Always test whether operation succeeded

Line: 110 Column: 1 C++ Tab Size: 4 main

ofstream - C++ Reference

www.cplusplus.com/reference/fstream/ofstream/

The T-Syste...are Archive | German <->...U Chemnitz | Apple | Amazon | eBay | Yahoo! | News | Blackboard ...demic Suite

- C library:
- Containers:
- Input/Output:
  - <fstream>
  - <iomanip>
  - <ios>
  - <iosfwd>
  - <iostream>
  - <istream>
  - <ostream>
  - <sstream>
  - <streambuf>
- Other:

<fstream>

- classes:
  - filebuf
  - fstream
  - ifstream
  - ofstream

ofstream

ofstream::ofstream

- member functions:
  - ofstream::close
  - ofstream::is_open
  - ofstream::open
  - ofstream::rdbuf

std::ofstream

<fstream>

Output file stream

ios_base → ios → ostream → ofstream

ofstream provides an interface to write data to files as output streams.

The objects of this class maintain internally a pointer to a filebuf object that can be obtained by calling member rdbuf.

The file to be associated with the stream can be specified either as a parameter in the constructor or by calling member open.

After all necessary operations on a file have been performed, it can be closed (or disassociated) by calling member close. Once closed, the same file stream object may be used to open another file.

The member function is_open can be used to determine whether the stream object is currently associated with a file.

Public members

(constructor)	Construct object and optionally open file (public member function)
rdbuf	Get the associated filebuf object (public member function)
is_open	Check if a file is open (public member function)
open	Open file (public member function)
close	Close file (public member function)

Members inherited from ostream

operator<<	Insert formatted output (public member function)
put	Put character (public member function)
write	Write block of data (public member function)
tellp	Get position in output sequence (public member function)
seekp	Set position in output sequence (public member function)

Close Files

euclid.cpp

```
138 #include <iostream>     // include standard IO library
139 #include <fstream>      // include file IO library
140
141 using namespace std;
142
143 int main( int argc, char* argv[] )
144 {
145
146     ...
147
148     // close all file streams
149     lInput.close();
150     lOutput.close();
151
152     ...
153
154     return 0;
155 }
```

Line: 124 Column: 5 C++ Tab Size:

Release files once these resources are not needed anymore.

File-based Character Counter

Main.cpp

```cpp
    CharacterCounter lCounter;

    unsigned char lChar;

    while ( lInput >> lChar )
    {
        lCounter.count( lChar );
    }

    lOutput << lCounter;
```

Line: 1 Column: 1 C++ Tab Size: 4

- We read input through input file stream lInput and write results to output file stream lOutput.

Using IO manipulators

I/O Manipulator Tests

IOManip.cpp

```cpp

#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
    cout << "The number 12345" << endl;
    cout << "in HEX: " << hex << 12345 << " and DEC: " << dec << 12345 << endl;

    cout << "The number 28 as 8 digit hexadecimal number:\t"
         << setw(8) << setfill('0') << hex << 28 << endl;

    cout << "The number 28 as 8 digit decimal number:\t"
         << setw(8) << setfill('0') << dec << 28 << endl;

    cout << "The number 28 as 8 digit hexadecimal number:\t"
         << setw(8) << setfill(' ') << hex << 28 << endl;

    cout << "The number 28 as 8 character decimal number:\t"
         << setw(8) << setfill(' ') << dec << 28 << endl;

    return 0;
}
```

Line: 19 Column: 34 C++ Tab Size: 4 main

cctype

<cctype> (ctype.h)

Character handling functions

This header declares a set of functions to classify and transform individual characters.

fx Functions

These functions take the `int` equivalent of one character as parameter and return an `int` that can either be another character or a value representing a boolean value: an `int` value of `0` means false, and an `int` value different from `0` represents true.

There are two sets of functions:

Character classification functions

They check whether the character passed as parameter belongs to a certain category:

isalnum	Check if character is alphanumeric (function)
isalpha	Check if character is alphabetic (function)
isblank	Check if character is blank (function)
iscntrl	Check if character is a control character (function)
isdigit	Check if character is decimal digit (function)
isgraph	Check if character has graphical representation (function)
islower	Check if character is lowercase letter (function)
isprint	Check if character is printable (function)
ispunct	Check if character is a punctuation character (function)
isspace	Check if character is a white-space (function)
isupper	Check if character is uppercase letter (function)
isxdigit	Check if character is hexadecimal digit (function)

Character conversion functions

Two functions that convert between letter cases:

tolower	Convert uppercase letter to lowercase (function)
toupper	Convert lowercase letter to uppercase (function)

Output

Command Prompt

```
X:\Desktop\Courses@Swin\2013-1\HIT3303\Labs\Lab3\Program\UseIOManip\D
 Volume in drive X is Shared Folders
 Volume Serial Number is 0000-0000

 Directory of X:\Desktop\Courses@Swin\2013-1\HIT3303\Labs\Lab3\Progra

03/18/2013  01:30 PM    <DIR>          .
03/18/2013  01:30 PM    <DIR>          ..
03/18/2013  01:30 PM            39,936 UseIOManip.exe
03/18/2013  01:30 PM           388,832 UseIOManip.ilk
03/18/2013  01:30 PM           584,704 UseIOManip.pdb
               3 File(s)      1,013,472 bytes
               2 Dir(s)  302,827,110,400 bytes free

X:\Desktop\Courses@Swin\2013-1\HIT3303\Labs\Lab3\Program\UseIOManip\D
The number 12345
in HEX: 3039 and DEC: 12345
The number 28 as 8 digit hexadecimal number:    0000001c
The number 28 as 8 digit decimal number:        00000028
The number 28 as 8 digit hexadecimal number:          1c
The number 28 as 8 character decimal number:          28

X:\Desktop\Courses@Swin\2013-1\HIT3303\Labs\Lab3\Program\UseIOManip\D
```

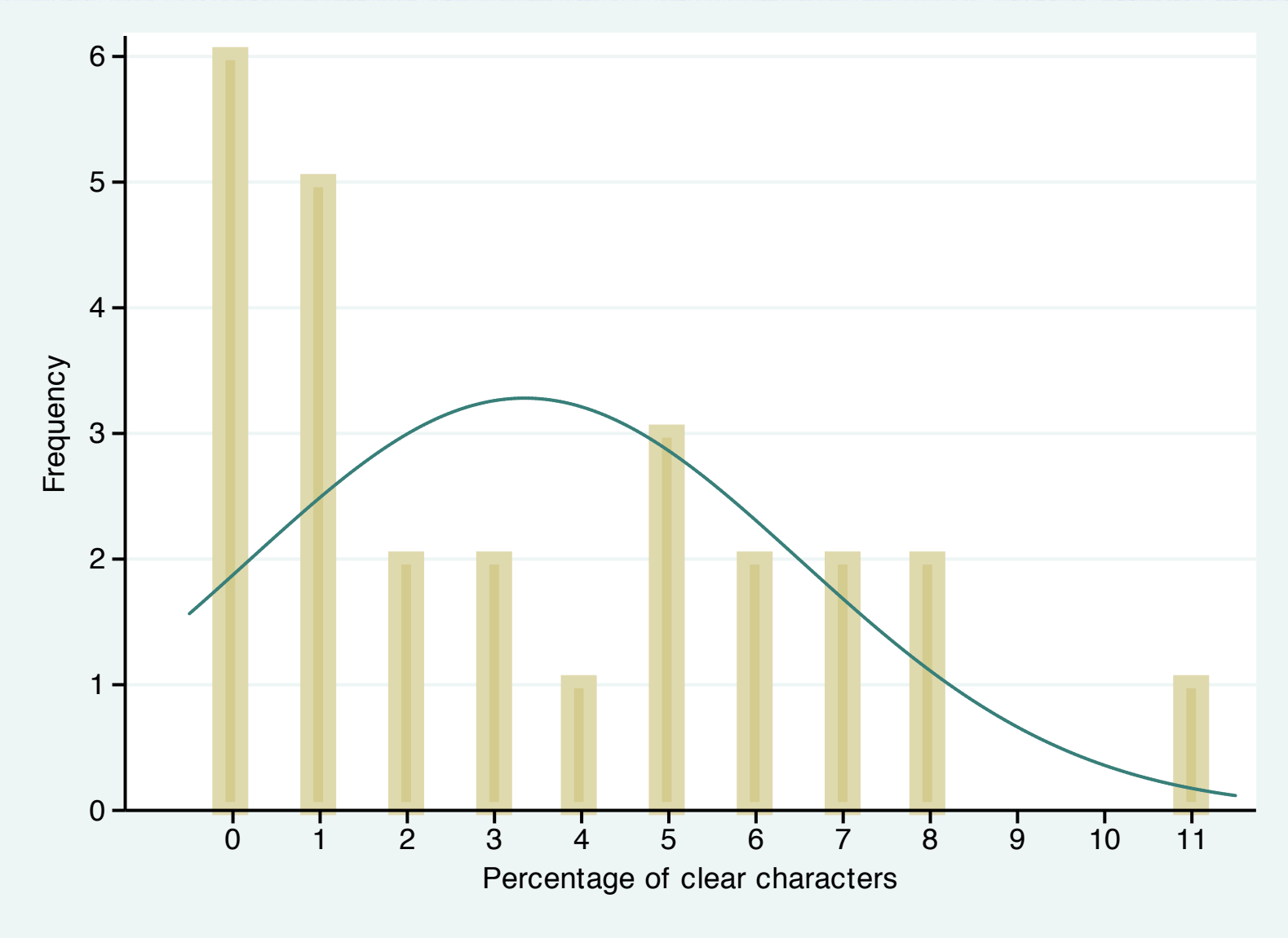
Character Frequencies

sample.txt

```
1
2                ACT I
3
4                   SCENE I
5
6              London. A Street.
7
8              Enter Gloucester.
9
10 Gloucester. Now is the winter of our discontent
11     Made glorious summer by this sun of York;
12     And all the clouds that lour'd upon our house
13     In the deep bosom of the ocean buried.
14     Now are our brows bound with victorious wreaths;
15     Our bruised arms hung up for monuments;
16     Our stern alarums changed to merry meetings;
17     Our dreadful marches to delightful measures.
18     Grim-visag'd war hath smooth'd his wrinkled front;
19     And now, - instead of mounting barbed steeds,
20     To fright the souls of fearful adversaries, -
21     He capers nimbly in a lady's chamber
22     To the lascivious pleasing of a lute.
23     But I, that am not shap'd for sportive tricks,
24     Nor made to court an amorous looking-glass;
```

Line: 29 Column: 14 Plain Text Tab Size: 4

Character Distribution - Clear Text



Purpose of Encoding

- Secure a message
- Change the character frequencies to make it more difficult to use brut-force attacks
- Communicate more effectively

Characters as Integers

- Character values in C/C++ are numbers.
- Individual characters are enclosed in single quotes. We write 'A' to mean the character A.
- The numeric value of a character is given as its ASCII code. For example, the character 'A' has ASCII code 65 or 0x41, whereas ' ' (the space character) is assigned 32 or 0x20.
- We can use integer operation to manipulate, create, or convert character values:
  - 'A' + 1 == 'B'
  - 0x41 + 0x20 == 'a'
  - 'B' - 'A' == 1

Caesar's Cipher

caesar.cpp

```cpp
#include <iostream>

using namespace std;

char performCaesar( char aSource )
{
    return 'A' + ((aSource - 'A' + 4) % 26);
}

int main()
{
    cout << "performCaesar( 'M' ) = " << performCaesar( 'M' ) << endl;

    return 0;
}
```

Line: 18 Column: 1 C++ Tab Size: 4 main

- Julius Caesar used a substitution cipher to encrypt private text messages.
- In this cipher every letter (26 upper-case character alphabet) is replaced by a letter some fixed number down the alphabet. This cipher uses a round-robin method (the modulo term) to keep the resulting letter within the original alphabet.

Enigma Machine

- Enigma machine used by the Germans in World War 2.
- Its biggest problem was that no character could be encoded as itself. This and other structural elements allowed crypto analysts at Bletchley Park to decode messages without having the key.

source: http://www.bletchleypark.org.uk/

sample.txt.secure.txt

```
1
2              URI V
3
4                 TQDGJ L
5
6            Dthwic. P Fufdxy.
7
8            Hfyyk Aadhdsrmju.
9
10 Yqinwthgff. Mhb lk ybx qxcgff ny txj iclwdcgfbs
11     Ffgw lfhlxdht gtfrhj gs mbxh fvb ny Drjp;
12     Ugx pay uvd vqrmim mbpi ypiq'w zsgs inl wdhts
13     Hg ykw iyxj qdfpa ny ykw twxuc qhswdw.
14     Sro flx ijg oscvl grmsx pciw ijqshwlgzm pltpgig;
15     Nnw ejzclys peng gnsj mu zhl bdavadgyv;
16     Gzl lntga bzzkzpk hbthvtq uc lxwuq ryxnxctt;
17     Ctk iuwfxyoa bnsqgxx wg iyecvwggik fjdkzlxm.
18     Vgvn-jhlfj'v buk bpiu tanhyk'v mcl qgxalzdw kugsn;
19     Ths cbx, - wmlyhsi iy gdjauwmz gdjgyw mitreg,
20     Sh kualbm nwt fpikl ti xjukzja nejdkxdjnyl, -
21     Bt rnqsql slegfr cc p ybrx'l hksrvxl
22     Id gis ktxfaachoh eyforbsj gk u eoit.
23     Ovh H, mmdl fg gii hubd'c ytu kuiknxkr ufhvpv,
24     Ftl fust gp qnnww ss ufigdht znhplfl-aeuhh;
```

Line: 36 Column: 16 Plain Text Tab Size: 4

Character Distribution - Encoded Text