// CSCI 301

// Computer Science 2

// File: format.cxx

// This program reads an input file of text and writes an output file of the

// same text, formatted into lines no longer than a maximum length. The

// names of the input and output files and the maximum line length are

// read from the terminal. Functions open the files, and continue prompting

// for file names until names are entered than can be successfully opened.

// Another function reads the maximum line length, which must fall within

// bounds set by two program constants.

// The program reads and writes words from the input file one at a time.

// It keeps track of the length of the current line so far; if the next

// word would cause that line to exceed the maximum length, the program

// terminates that line, writes the word on the next line, and resets the

// line length. The program writes a blank after each word, except perhaps

// the last word on a line. A word is a string of contiguous non-blank

// characters, and we assume that no input word is longer than the input line

// length set for the run.

#include

#include

#include

#include

using namespace std;

const int MIN = 30; // Minimum line length

const int MAX = 80; // Maximum line length

typedef char string[MAX+1];

void open\_input\_file ( ifstream& in\_f );

// Opens for input a file named from the terminal.

// Postcondition: A file stream has been opened for input.

void open\_output\_file ( ofstream& out\_f );

// Opens for output a file named from the terminal.

// Postcondition: A file stream has been opened for output.

int read\_int ( int small, int large );

// Reads an input value within specified bounds.

// Precondition: small and large are positive integers, with small <= large.

// Postcondition: The function returns a value in [small,large] entered from

// the terminal.

int main()

{

ifstream in\_file; // The input file stream

ofstream out\_file; // The output file stream

int max\_length; // Maximum line length

string s; // Each string read in and printed out

int s\_len; // The length of the string s

int line\_len; // The length of the current output line so far

open\_input\_file(in\_file); // Open the input file.

open\_output\_file(out\_file); // Open the output file.

max\_length = read\_int(MIN,MAX); // Read the maximum line length.

line\_len = 0; // Initially, the line length is zero.

in\_file >> s; // Read from the input file.

while ( ! in\_file.eof() ) // Are we done yet?

{

s\_len = strlen(s); // Identify the string's length.

if ( line\_len + s\_len <= max\_length ) // If there is room on the line ...

{

out\_file << s; // Write to the output file.

line\_len = line\_len + s\_len; // Increment the line length.

}

else // Start a new line.

{

out\_file << endl << s; // Write to the output file.

line\_len = s\_len; // Reset the line length.

}

if ( line\_len < max\_length ) // If there is room for a blank ...

{

out\_file << ' '; // Write to the output file.

++line\_len;

}

in\_file >> s; // Read from the input file.

}

out\_file << endl; // Write to the output file.

in\_file.close(); // Close the input file.

out\_file.close(); // Close the output file.

return EXIT\_SUCCESS;

}

void open\_input\_file ( ifstream &in\_f )

{

char input\_file\_name[80];

do

{ in\_f.clear();

cout << "Enter input file name: ";

cin >> input\_file\_name;

in\_f.open(input\_file\_name);

} while ( in\_f.fail() );

}

void open\_output\_file ( ofstream &out\_f )

{

char output\_file\_name[80];

do

{ out\_f.clear();

cout << "Enter output file name: ";

cin >> output\_file\_name;

out\_f.open(output\_file\_name);

} while ( out\_f.fail() );

}

int read\_int ( int small, int large )

{

int value;

do

{ cout << "Enter an integer value between " << setw(1) << small

<< " and " << setw(1) << large << ": ";

cin >> value;

} while ( value < small || value > large );

return value;

}