```
Script started on Sat 08 Dec 2012 08:56:15 PM CST
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ p
wd
/home/georgia/cplusplus
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ C
PP --version
This is CPP version 1.219 executing under perl v5.12.4 and compiling with:

g++ (Ubuntu/Linaro 4.6.1-9ubuntu3) 4.6.1
Copyright (C) 2011 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.


\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ c
at phraselib.h
#ifndef PHRASELIB_H_INC
#define PHRASELIB_H_INC

#include <iostream>
#include <iomanip>
#include <string>
#include <vector>
#include <cctype>

// Draws a box around a phrase entered at the keyboard or from a file
class phraseBox
{
    char border;
    char alignment;
    std::vector<std::string> stringvec;

public:
    phraseBox(void) : border('*'), alignment('L'),
                      stringvec(std::vector<std::string>()) { }
    phraseBox(char c, char a) : border(c), alignment(a),
                                    stringvec(std::vector<std::string>())
    {
        set_alignment(a);
        set_border(c);
    }

    std::vector<std::string>::size_type longest(void) const
    {
        std::vector<std::string>::size_type i, larindex=0;

        for (i=0; i<stringvec.size(); i++)
        {
            if (stringvec[i].size() > stringvec[larindex].size())
            {
                larindex = i;
            }
        }
        return larindex;
    }
    void set_border(char b) { isprint(b) ? border = b : border = '*' ; return; }
    char get_border(void) const { return border; }
    void set_alignment(char a)
    {
        alignment = (toupper(a)!='L' && toupper(a)!='C' && toupper(a)!='R')
        ? 'L' : toupper(a);
        return;
    }
    char get_align(void) const { return alignment; }
    std::string operator[](std::vector<std::string>::size_type i) const
    {
        return stringvec[i];
    }
    std::vector<std::string>::size_type size(void) const
    {
        return stringvec.size();
    }
    void insert(std::string s)
    {
        stringvec.push_back(s);
        return;
    }
    void reset(void)
    {
        stringvec.erase(stringvec.begin(), stringvec.end());
        alignment = 'L';
        border = '*';
    }


};

std::ostream & operator<<(std::ostream & out, const phraseBox & pb);
std::istream & operator>>(std::istream & in, phraseBox & pb);
#endif

\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ c
at phraselib.cpp
#include "phraselib.h"

std::ostream & operator<<(std::ostream & out, const phraseBox & pb)
{
    size_t width = pb[pb.longest()].size()+2;
    size_t pre, post;

    out << std::setw(width+3) << std::setfill(pb.get_border()) << '\n';

    for (std::vector<std::string>::size_type i=0; i < pb.size(); i++)
    {
        pre = pb.get_align() == 'C' ? (width-pb[i].size())/2 : 0;
        if (pre*2 < width-pb[i].size() && pre != 0)
        {
            post = (width-pb[i].size())/2 + 1;
        }
        else if (pre > 0)
        {
            post = pre;
        }
        else
        {
            post = 0;
        }
        out << pb.get_border();
        out << std::setw(pre) << std::setfill(' ') << "";
        if (pb.get_align() == 'L')
        {
            out << ' ' << std::setw(width-1)
                << std::setiosflags(std::ios_base::left) << pb[i];
            out << std::resetiosflags(std::ios_base::left);
        }
        else if (pb.get_align() == 'R')
        {
            out << std::setw(width-1)
                << std::setiosflags(std::ios_base::right) << pb[i] << ' ';
            out << std::resetiosflags(std::ios_base::right);
        }
        else
        {
            out << pb[i];
        }
        out << std::setw(post) << std::setfill(' ') << "";
        out << pb.get_border() << '\n';
    }
```

```
    out << std::setw(width+3) << std::setfill(pb.get_border()) << '\n';

    return out;
}

std::istream & operator>>(std::istream & in, phraseBox & pb)
{
    std::string s;
    while(in.peek() != '\n')
    {
        in >> s;
        pb.insert(s);
    }

    return in;
}
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ c
at phrasetest.cpp
#include <iostream>
#include <string>
#include <fstream>
#include <climits>
#include "phraselib.h"

using namespace std;

int main(void)
{
    char choice, borderchar;
    phraseBox pbox;
    string filename;
    ifstream infile;
    ofstream outfile;

    do
    {
        cout << "\nPlease choose from the menu: "
         << "\n1. Read from keyboard and write to screen"
         << "\n2. Read from keyboard and write to file"
         << "\n3. Read from file and write to screen"
         << "\n4. Read from file and write to file"
         << "\n5. Quit\n";
         cin >> choice;
         cin.ignore(INT_MAX, '\n');
            if (choice == '1')
            {
                cout << "\nEnter the border character: ";
                cin >> borderchar;
                cin.ignore(INT_MAX, '\n');
                pbox.set_border(borderchar);
                cout << "\nEnter a phrase: ";
                cin >> pbox;
                cout << "\nYour phrase looks like this:\n";
                pbox.set_alignment('L');
                cout << pbox;
                pbox.set_alignment('R');
                cout << pbox;
                pbox.set_alignment('C');
                cout << pbox;
                cout << "\nWould you like to continue? ";
                cin >> choice;
                cin.ignore(INT_MAX, '\n');
                pbox.reset();
            }
        else if (choice == '2')
        {
            cout << "\nEnter the border character: ";
            cin >> borderchar;
            cin.ignore(INT_MAX, '\n');
            pbox.set_border(borderchar);
            cout << "\nEnter the output file name:  ";
            getline(cin, filename);
            outfile.open(filename.c_str());
            while (!outfile)
            {
                outfile.close();
                outfile.clear();
                cout << "Invalid output file name!" << endl;

                cout << "Enter the output file name:  ";
                getline(cin, filename);

                outfile.open(filename.c_str());
            }

            cout << "\nEnter your phrase: ";
            cin >> pbox;
            pbox.set_alignment('L');
            outfile << pbox;
            pbox.set_alignment('R');
            outfile << pbox;
            pbox.set_alignment('C');
            outfile << pbox;
            outfile.close();
            outfile.clear();
            cout << "\nOutput file written successfully!";
            cout << "\nWould you like to continue? ";
            cin >> choice;
            cin.ignore(INT_MAX, '\n');
            pbox.reset();

        }
        else if (choice == '3')
        {
            cout << "\nEnter the border character: ";
            cin >> borderchar;
            cin.ignore(INT_MAX, '\n');
            pbox.set_border(borderchar);
            cout << "\nEnter the input file name:  ";
            getline(cin, filename);
            infile.open(filename.c_str());
            while (!infile)
            {
                infile.close();
                infile.clear();

                cout << "File Not Found!" << endl;

                cout << "Input file name:  ";
                getline(cin, filename);

                infile.open(filename.c_str());
            }
            infile >> pbox;
            infile.close();
            infile.clear();
            cout << "\nYour phrase looks like this:\n";
            pbox.set_alignment('L');
            cout << pbox;
            pbox.set_alignment('R');
            cout << pbox;
            pbox.set_alignment('C');
            cout << pbox;
            cout << "\nWould you like to continue? ";
            cin >> choice;
            cin.ignore(INT_MAX, '\n');
            pbox.reset();
```

```
        }
        else if (choice == '4')
        {
            cout << "\nEnter the border character: ";
            cin >> borderchar;
            cin.ignore(INT_MAX, '\n');
            pbox.set_border(borderchar);
            cout << "\nEnter the input file name:  ";
            getline(cin, filename);
            infile.open(filename.c_str());
            while (!infile)
            {
                infile.close();
                infile.clear();

                cout << "File Not Found!" << endl;

                cout << "Input file name:  ";
                getline(cin, filename);
                infile.open(filename.c_str());
            }
            cin.clear();
            infile >> pbox;
            infile.close();
            infile.clear();
            cout << "\nFile read successfully! Enter the output file name:  ";
            getline(cin, filename);
            outfile.open(filename.c_str());
            while (!outfile)
            {
                outfile.close();
                outfile.clear();
                cout << "Invalid output file name!" << endl;

                cout << "Enter the output file name:  ";
                getline(cin, filename);

                outfile.open(filename.c_str());
            }
            pbox.set_alignment('L');
            outfile << pbox;
            pbox.set_alignment('R');
            outfile << pbox;
            pbox.set_alignment('C');
            outfile << pbox;
            outfile.close();
            outfile.clear();
            cout << "\nFile written successfully!";
            cout << "\nWould you like to continue? ";
            cin >> choice;
            cin.ignore(INT_MAX, '\n');
            pbox.reset();
         }
    } while (toupper(choice) == 'Y');

    cout << "\nThanks for using the phrase-boxer!\n";

    return 0;
}


\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ C
PP phraselib phrasetest
phraselib.cpp...
phrasetest.cpp***
In file included from phraselib.cpp:1:0:
phraselib.h: In member function â\200\230void
phraseBox::set_alignment(char)â\200\231:
phraselib.h:45:26: warning: conversion to â\200\230charâ\200\231 from â\200\230intâ
\200\231
may alter its value [-Wconversion]
In file included from phrasetest.cpp:5:0:
phraselib.h: In member function â\200\230void
phraseBox::set_alignment(char)â\200\231:
phraselib.h:45:26: warning: conversion to â\200\230charâ\200\231 from â\200\230intâ
\200\231
may alter its value [-Wconversion]


\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ C
\033[Kphraselib phrasetest.cpp phrasetest
georgia@georgia-MT6017:~/cplusplus$ ./phrasetest.out

Please choose from the menu:
1. Read from keyboard and write to screen
2. Read from keyboard and write to file
3. Read from file and write to screen
4. Read from file and write to file
5. Quit
1

Enter the border character: #

Enter a phrase: Here is a phrase written on the keyboard!

Your phrase looks like this:
##############
# Here       #
# is         #
# a          #
# phrase     #
# written    #
# on         #
# the        #
# keyboard!  #
##############
##############
#      Here  #
#        is  #
#         a  #
#    phrase  #
#   written  #
#        on  #
#       the  #
# keyboard!  #
##############
##############
#   Here     #
#    is      #
#     a      #
#  phrase    #
#  written   #
#    on      #
#    the     #
# keyboard!  #
##############

Would you like to continue? y

Please choose from the menu:
1. Read from keyboard and write to screen
2. Read from keyboard and write to file
3. Read from file and write to screen
4. Read from file and write to file
5. Quit
2

Enter the border character: @
```

```
Enter the output file name:  outphrase.txt

Enter your phrase: Here is a phrase written into a file!

Output file written successfully!
Would you like to continue? y

Please choose from the menu:
1. Read from keyboard and write to screen
2. Read from keyboard and write to file
3. Read from file and write to screen
4. Read from file and write to file
5. Quit
3

Enter the border character: &

Enter the input file name:  boxfile.txtx

Your phrase looks like this:
&&&&&&&&&&
& Here   &
& is     &
& a      &
& phrase &
& read   &
& from   &
& a      &
& file!  &
&&&&&&&&&&
&&&&&&&&&&
&   Here &
&     is &
&      a &
& phrase &
&   read &
&   from &
&      a &
&  file! &
&&&&&&&&&&
&&&&&&&&&&
&  Here  &
&   is   &
&   a    &
& phrase &
&  read  &
&  from  &
&   a    &
& file!  &
&&&&&&&&&&

Would you like to continue? y

Please choose from the menu:
1. Read from keyboard and write to screen
2. Read from keyboard and write to file
3. Read from file and write to screen
4. Read from file and write to file
5. Quit
4

Enter the border character: #

Enter the input file name:  boxfile.txt

File read successfully! Enter the output file name:  filetofile.txt

File written successfully!
Would you like to continue? n


Thanks for using the phrase-boxer!
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ c
at outphrase.txt
@@@@@@@@@@@
@ Here    @
@ is      @
@ a       @
@ phrase  @
@ written @
@ into    @
@ a       @
@ file!   @
@@@@@@@@@@@
@@@@@@@@@@@
@    Here @
@      is @
@       a @
@  phrase @
@ written @
@    into @
@       a @
@   file! @
@@@@@@@@@@@
@@@@@@@@@@@
@  Here   @
@   is    @
@    a    @
@ phrase  @
@ written @
@  into   @
@    a    @
@  file!  @
@@@@@@@@@@@
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ c
at boxfile.tzx033[K033[Kxt
Here is a phrase read from a file!
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ c
at filetofile.txt
##########
# Here   #
# is     #
# a      #
# phrase #
# read   #
# from   #
# a      #
# file!  #
##########
##########
#   Here #
#     is #
#      a #
# phrase #
#   read #
#   from #
#      a #
#  file! #
##########
##########
#  Here  #
#   is   #
#   a    #
# phrase #
#  read  #
#  from  #
#   a    #
# file!  #
##########
```

```
\033]0;georgia@georgia-MT6017: ~/cplusplus\007georgia@georgia-MT6017:~/cplusplus$ e
xit
exit

Script done on Sat 08 Dec 2012 09:01:32 PM CST
```