```
Script started on Thu 19 May 2016 11:21:25 AM CDT
\033]0;g_butler4@mars:~/csc122/box\007[g_butler4@mars box]$ pwd
/home/students/g_butler4/csc122/box
\033]0;g_butler4@mars:~/csc122/box\007[g_butler4@mars box]$ cat frame.cpp
#include <string>
#include <iostream>
#include <istream>
#include <vector>
#include <fstream>
#include "ctype.h"
#include "frame.h"

using namespace std;

int main()
{
        int menu = 0;
        int menu2 = 0;
        string work;
        ifstream file;
        ofstream ofile;
        char c;


        frame A;
        cout << "Input from file(1) or from keyboard(2):";
        cin >> menu;
        cout << "\n";
        if (menu == 1)
        {
                cout << "Enter Filename:\n";
                cin >> work;
                if (work.find_first_of('.') == -1)
                {
                        work.append(".txt");
                }
                if (isalpha(work[0]) || isdigit(work[0]))
                {
                        file.open(work.c_str());
                }
                else
                {
                        while ((!isalpha(work[0])) && (!isdigit(work[0])))
                        {
                                work.clear();
                                cout << "\nTry again";
                                cout << "\nEnter file to be written to: ";
                                cin >> work;
                                if (work.find_first_of('.') == -1)
                                {
                                        work.append(".txt");
                                }
                        }
                }
                work.clear();
                getline(file, work);
                A.in_str(work);
        }
        cout << "Output to file(1) or screen(2):\n";
                cin >> menu2;
                if (menu2 == 1)
                {
                        cout << "Enter filename to write to: ";
                        cin >> work;
                        if (work.find_first_of('.') == -1)
                        {
                                work.append(".txt");
                        }
                        ofile.open(work.c_str());
                }
        if (menu == 2)
        {
                cout << "Enter string:\n";
                cin>>ws;
                getline(cin, work);
                A.in_str(work);
        }
        cin.clear();

        cout << "\nBorder type: (S)ingle Line, (D)ouble Line, or a char:\n";
        cin >> c;
        A.set_bt(c);

        cout << "\nShaded?(Y/N):";
        cin >> c;
        if ((c == 'Y') || (c == 'y'))
        {
                cout << "\nEnter shaded char: ";
                cin >> c;
                A.set_board2(c);
        }

        cout << "\nEnter Justification: (C)enter (L)eft (R)ight\n";
        cin >> c;
        A.set_just(c);

        A.in_str("\0");
        if (menu2 == 2)
        {
                cout << "Your output: \n" << A<<endl;
                if (menu == 1)
                {
                        A.reset();
                        while (!(file.eof()))
                        {
                                getline(file, work);
                                A.in_str(work);
                                A.in_str("\0");
                                cout << A<<endl;
                                A.reset();
                        }
                }
        }

        if (menu2 == 1)
        {
                ofile << A;
                A.reset();
                while (!(file.eof()))
                {
                        getline(file, work);
                        A.in_str(work);
                        A.in_str("\0");
                        ofile << A;
                        A.reset();
                }
        }
        file.close();
        ofile.close();

        return 0;
}

\033]0;g_butler4@mars:˜/csc122/box\007[g_butler4@mars box]$ \033[K\007\007c\033[K\033[K
\007\007\007\007cat frame.h
#ifndef FRAME_H
#define FRAME_H

#include <vector>
#include <iostream>
#include <fstream>
#include <string>
#include "ctype.h"
class frame
{
public:
        std::vector<std::string> wl;
        int cpos;
        int l_pos;
        int max_l;
        char j;
        char bt;
        char b2;

        frame()
                :wl(),cpos(0),l_pos(0),max_l(0), j('\0'), bt('\0'), b2('\0')
                {}


        void reset()
        {
                cpos = 0;
                wl.clear();
                max_l = 0;
                l_pos = 0;
        }

        inline int get_max_length()
        {
                return max_l;
        }

        inline void set_max_length(int in)
        {
                max_l = in;
        }

        inline std::string get_wl(int pos)
        {
                return wl[pos];
        }


        inline void set_bt(char in)
        {
                if (isprint(in))
                {
                        bt = in;
                }
        }
        inline char get_bt(void)
        {
```

```
                return bt;
        }

        inline void set_board2(char in)
        {
                if (isprint(in))
                {
                        b2 = in;
                }
        }

        inline char get_board2(void)
        {
                return b2;
        }
        inline void set_just(char & in)
        {
                if (((in == 'l') || (in == 'c') || (in == 'r')))
                {
                        if (in == 'l')
                        {
                                j = 'l';
                        }
                        if (in == 'c')
                        {
                                j = 'c';
                        }
                        if (in == 'r')
                        {
                                j = 'r';
                        }
                }
                else(j == 'l');
        }
        inline char get_just(void)
        {
                return j;
        }

        inline std::ostream & output(std::ostream & out)
        {
                int l = get_max_length();
                std::string word = get_wl(0);
                char bt = get_bt();
                char b2 = get_board2();
                int count = 1;
                if (((bt == 'S') || (bt == 's')) || ((bt == 'd') || (bt == 'D')))
                {
                        if ((bt == 'S') || (bt == 's'))
                        {
                                out << '+' << std::string((l + 2), '-') << '+' << std:
:endl;
                        }
                        if (((bt == 'd') || (bt == 'D')))
                        {
                                out << '+' << std::string((l + 2), '=') << '+' << std:
:endl;
                        }
                }
                else
                {
                        out << std::string((l + 4), bt) << std::endl;
                }
                while (word != "\0")
                {
                        if (((bt == ’s’) || (bt == ’S’)) || ((bt == ’d’) || (bt == ’D’
)))
                        {
                                if ((bt == ’s’) || (bt == ’S’))
                                {
                                        out << "| ";
                                }

                                if ((bt == ’d’) || (bt == ’D’))
                                {
                                        out << "||";
                                }
                        }
                        else
                        {
                                out << bt << " ";
                        }
                        if (get_just() == ’l’)
                        {

                                if (word.length() != l)
                                {
                                        out.width(l);
                                        out << std::left << word;

                                }
                                else(out << word);
                        }
                        if (get_just() == ’c’)
                        {

                                out << std::string(((l - word.length()) / 2), ’ ’);
                                out << word;
                                out << std::string((((l - word.length()) / 2)), ’ ’);

                        }
                        if (get_just() == ’r’)
                        {
                                if (word.length() != l)
                                {
                                        out.width(l);
                                        out << std::right << word;

                                }
                                else(out << word);
                        }
                        if (((bt == ’s’) || (bt == ’S’)) || ((bt == ’d’) || (bt == ’D’
)))
                        {
                                if ((bt == ’s’) || (bt == ’S’))
                                {
                                        out << " |";
                                }
                                if ((bt == ’d’) || (bt == ’D’))
                                {
                                        out << "||";
                                }
                        }
                        else
                        {
                                out << " " << bt;
                        }
                        if (b2 != (’\0’))
                        {
                                out << b2 << std::endl;
                        }
                        else(out << std::endl);

                        word = get_wl(count);
                        ++count;
                }
                if ((bt == ’s’) || (bt == ’S’) || (bt == ’d’) || (bt == ’D’))
                {
                        if ((bt == ’s’) || (bt == ’S’))
                        {
                                out << "+" << std::string((l + 2), ’-’) << "+";
                                if (b2 != ’\0’)
                                {
                                        out << b2;
                                }
                        }
                        if ((bt == ’d’) || (bt == ’D’))
                        {
                                out << "+" << std::string((l + 2), ’=’) << "+";
                                if (b2 != ’\0’)
                                {
                                        out << b2;
                                }
                        }
                        out << std::endl;
                }
                else
                {
                        out << std::string((l + 4), bt);
                        if (b2 != ’\0’)
                        {
                                out << b2;
                        }
                        out << std::endl;
                }
                if (b2 != (’\0’))
                {
                        out << " " << std::string((l + 4), b2) << std::endl;
                }
                return out;

        }

        inline void in_str(std::string in)
        {
                int a = 0;
                int s = 0;
                std::string temp;
                if (in != ("\0"))
                {
                        while (a < in.length())
                        {
                                if ((!isspace(in[a])))
                                {
                                        temp.push_back(in[a]);
                                }
                                if (isspace(in[a]))
                                {
                                        s = a;
                                        if (cpos == 0)
                                        {
                                                wl.push_back(temp);
                                                set_max_length(temp.length());
                                                l_pos = 0;
                                                cpos++;
                                        }
                                        else
                                        {
                                                wl.push_back(temp);
                                                if (wl[cpos].length() > wl[l_pos].leng
th())
                                                {
                                                        l_pos = cpos;
                                                        set_max_length(wl[cpos].length
());
                                                }
                                                ++cpos;
                                        }
                                        temp.clear();
                                }
                                a++;
                        }
                }
                wl.push_back(temp);
        }

        friend std::ostream & operator<<(std::ostream & out,frame & b);//output

};

inline std::ostream & operator<< (std::ostream & out, frame & b)//output
{
        b.output(out);
        return out;
}


#endif\033]0;g_butler4@mars:˜/csc122/box\007[g_butler4@mars box]$ CPP frame.cpp frame.
h
frame.cpp***
In file included from frame.cpp:7:
frame.h:12:   instantiated from here
frame.h:12:   instantiated from here
frame.h:12:   instantiated from here
frame.h: In member function ‘void frame::set_just(char&)’:
frame.h:90: warning: statement has no effect
frame.h: In member function ‘std::ostream&
frame::output(std::ostream&)’:
frame.h:140: warning: comparison between signed and unsigned integer
expressions
frame.h:158: warning: comparison between signed and unsigned integer
expressions
frame.h: In member function ‘void frame::in_str(std::string)’:
frame.h:234: warning: comparison between signed and unsigned integer
expressions
In file included from frame.cpp:7:
frame.h:279:7: warning: no newline at end of file
frame.cpp: In function ‘int main()’:
frame.cpp:29: warning: comparison between signed and unsigned integer
expressions
frame.cpp:45: warning: comparison between signed and unsigned integer
expressions
frame.cpp:61: warning: comparison between signed and unsigned integer
expressions
frame.h:28:   instantiated from here
frame.h:245:   instantiated from here
```

```

\033]0;g_butler4@mars:˜/csc122/box\007[g_butler4@mars box]$ ./frame.out
Input from file(1) or from keyboard(2):2

Output to file(1) or screen(2):
2
Enter string:
this is my test

Border type: (S)ingle Line, (D)ouble Line, or a char:
*

Shaded?(Y/N):y

Enter shaded char: &

Enter Justification: (C)enter (L)eft (R)ight
l
Your output:
********
* this *&
* is   *&
* my   *&
* test *&
********&
 &&&&&&&&

\033]0;g_butler4@mars:˜/csc122/box\007[g_butler4@mars box]$ ./frame.out
Input from file(1) or from keyboard(2):1

Enter Filename:
stick
Output to file(1) or screen(2):
1
Enter filename to write to: stick1

Border type: (S)ingle Line, (D)ouble Line, or a char:
s

Shaded?(Y/N):y

Enter shaded char: #

Enter Justification: (C)enter (L)eft (R)ight
c
\033]0;g_butler4@mars:˜/csc122/box\007[g_butler4@mars box]$ cat stick1.txt
+-----+
|  O  |#
| -o- |#
|  |  |#
| /-\ |#
+-----+#
 #######
\033]0;g_butler4@mars:˜/csc122/box\007[g_butler4@mars box]$ ./frame.out
Input from file(1) or from keyboard(2):1

Enter Filename:
trythis
Output to file(1) or screen(2):
2

Border type: (S)ingle Line, (D)ouble Line, or a char:
d
```

```
Shaded?(Y/N):n

Enter Justification: (C)enter (L)eft (R)ight
r
Your output:
+=======+
|| This||
|| test||
||   is||
||   on||
||  the||
||first||
|| line||
+=======+

+========+
||  This||
||    is||
||   the||
||middle||
||  line||
+========+

+======+
||this||
||  is||
|| the||
||last||
||line||
+======+

\033]0;g_butler4@mars:~/csc122/box\007[g_butler4@mars box]$ exit
exit

Script done on Thu 19 May 2016 11:28:00 AM CDT
```