```
Script started on 2023-07-27 12:23:54-05:00 [TERM="xterm" TTY="/dev/pts/2" COLUMNS=
bj94684@ares:~$ pwd
/home/students/bj94684
bj94684@ares:~$ cat box.info
/***************************************************************************
 *                                                                         *
 *  NAME:  Jose Barron                           CLASS:  CSC122-002        *
 *                                                                         *
 *          Lab: A box upon ye                    Level:   5               *
 *       Option:                                  Level: + 0               *
 *                                          Total Level:   5.0             *
 *                                                                         *
 * This program is designed to create a frame around a phrase given by a   *
 * user, either from keyboard input or a file. The user has the ability    *
 * to choose how the words are centered (center, left, or right) and what  *
 * symbol the frame is made of. The words are processed with a vector of   *
 * strings which are the words then with the use of the FramedWord class   *
 * the words are framed.
 *                                                                         *
***************************************************************************/
bj94684@ares:~$ show-code strextra.h


strextra.h:


#ifndef STREXTRA_H_INC
#define STREXTRA_H_INC

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

using namespace std;


inline std::string tolower_str(std::string s)
{
    std::string t = s;
    for(std::string::size_type pos = 0; pos != s.length(); ++pos)
    {
        t[pos] = static_cast<char>( tolower(t[pos]) );
    }
    return t;

}

inline std::vector<std::string::size_type> find(std::string s, char t,
                                      bool case_sensitive = true)
{
    if( ! case_sensitive)
    {
        s = tolower_str(s);
        t = static_cast<char>( tolower(t) );
    }

    std::vector<string::size_type> num;
    for (std::string::size_type pos = 0; pos != s.length(); ++pos)
    {
        if (static_cast<char>( s[pos] ) == t )
        {
            num.push_back(pos);
        }
    }
    return num;
}

inline bool str_is_incl(std::string s, std::string t, std::string::size_ty
                                            std::string::size_type end)
{
    std::string nr = s.substr(beg, end-beg+1);
    bool tf;
    if ( nr == t)
    {
        return tf = true;
    }
    return tf = false;
}

inline std::string::size_type strcom(std::vector<string::size_type> b,
                                std::vector<string::size_type> e,
                                std::string s, std::string t)
{
    string::size_type num;
    for(auto p : b)
    {
        for(auto p1 : e)
        {
            if ( p < p1)    // The starter position can never be greater
            {
                bool ys = str_is_incl(s, t, p, p1);
                if ( ys)
                {
                    return num = p;
                }
            }
        }
    }
    return num = s.length();
}

std::string::size_type find(std::string s, std::string t, bool c_s = true)



#endif
bj94684@ares:~$ show-code strextra.h
```

```
strextra.h:


#ifndef STREXTRA_H_INC
#define STREXTRA_H_INC

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

using namespace std;


inline std::string tolower_str(std::string s)
{
    std::string t = s;
    for(std::string::size_type pos = 0; pos != s.length(); ++pos)
    {
        t[pos] = static_cast<char>( tolower(t[pos]) );
    }
    return t;

}

inline std::vector<std::string::size_type> find(std::string s, char t,
                                      bool case_sensitive = true)
{
    if( ! case_sensitive)
    {
        s = tolower_str(s);
        t = static_cast<char>( tolower(t) );
    }

    std::vector<string::size_type> num;
    for (std::string::size_type pos = 0; pos != s.length(); ++pos)
    {
        if (static_cast<char>( s[pos] ) == t )
        {
            num.push_back(pos);
        }
    }
    return num;
}

inline bool str_is_incl(std::string s, std::string t, std::string::size_typ
                                            std::string::size_type end)
{
    std::string nr = s.substr(beg, end-beg+1);
    bool tf;
    if ( nr == t)
    {
        return tf = true;
    }
    return tf = false;
}

inline std::string::size_type strcom(std::vector<string::size_type> b,
                                std::vector<string::size_type> e,
                                std::string s, std::string t)
{
    string::size_type num;
    for(auto p : b)
    {
        for(auto p1 : e)
        {
            if ( p < p1)    // The starter position can never be greater
            {
                bool ys = str_is_incl(s, t, p, p1);
                if ( ys)
                {
                    return num = p;
                }
            }
        }
    }
    return num = s.length();
}

std::string::size_type find(std::string s, std::string t, bool c_s = true)



#endif
bj94684@ares:~$ show-code box.cpp


box.cpp:


#include <iostream>
#include <fstream>
#include <vector>
#include <cctype>
#include <limits>
#include "strextra.h"

using namespace std;

inline string::size_type find_space( string s, char t)
{
    string::size_type num = s.length();
    for (string::size_type pos = 0; pos < s.length(); ++pos)
    {
        if (static_cast<char>( s[pos] ) == t )
        {
            return num = pos;
        }
    }
    return num;
}

vector<string> word_get(string sen);

class FrameWord
{
private:
    vector<string> words;
    short max_length_word;
    vector<string> framed_words;
    vector<string> framed_words_left;
    vector<string> framed_words_right;
    char frame_c;

    void center_word(string & word) const;
    void center_word_left(string & word) const;
    void center_word_right(string & word) const;


public:
    FrameWord(const vector<string> & inputted_words, char f_c);
    void frame_words();
    void frame_words_left();
    void frame_words_right();
    void display_frame() const;
    void display_frame_left() const;
    void display_frame_right() const;

};

FrameWord::FrameWord(const vector<string> & inputted_words, char f_c)
                    : words(inputted_words), max_length_word(0), frame_c(f_

{
    for (const auto & word : words)
    {
        max_length_word = max(max_length_word, static_cast<short>(word.len
    }
}

void FrameWord::center_word(string & word) const
{
    short space = max_length_word - static_cast<short>(word.length());
    short left_side = space / 2;
    short right_side = space - left_side;
    word = frame_c + string(left_side + 1, ' ') + word + string(right_side
}

void FrameWord::center_word_left(string & word) const
{
    short space = max_length_word - static_cast<short>(word.length());
    word = frame_c + word + string(space + 2, ' ') + frame_c;
}

void FrameWord::center_word_right(string & word) const
{
    short space = max_length_word - static_cast<short>(word.length());
    word = frame_c + string(space + 1, ' ') + word + " " + frame_c;
}

void FrameWord::frame_words()
{
    for (const auto & word : words)
    {
        string framed_word = word;
        center_word(framed_word);
        framed_words.push_back(framed_word);
    }
}

void FrameWord::frame_words_left()
{
    for (const auto & word : words)
    {
        string framed_word_left = word;
        center_word_left(framed_word_left);
        framed_words_left.push_back(framed_word_left);
    }
}

void FrameWord::frame_words_right()
{
    for (const auto & word : words)
    {
        string framed_word_right = word;
        center_word_right(framed_word_right);
        framed_words_right.push_back(framed_word_right);
    }
}

void FrameWord::display_frame() const
{
    short frame_size = max_length_word + 4;
    cout << string(frame_size, frame_c) << '\n';
    for (const auto & word : framed_words)
    {
        cout << word << endl;
    }

    cout << string(frame_size, frame_c) << '\n';
}

void FrameWord::display_frame_left() const
{
    short frame_size = max_length_word + 4;
    cout << string(frame_size, frame_c) << '\n';
    for (const auto & word : framed_words_left)
    {
        cout << word << endl;
    }
    cout << string(frame_size, frame_c) << '\n';
}

void FrameWord::display_frame_right() const
{
    short frame_size = max_length_word + 4;
    cout << string(frame_size, frame_c) << '\n';
    for (const auto & word : framed_words_right)
    {
        cout << word << endl;
    }
    cout << string(frame_size, frame_c) << '\n';
}

int main()
{
    cout << "\n\t\tA Box Upon Ye! Program\n\n";
    bool done = false;
    vector<string> words;
    do
    {
        char choice;
        cout << "Are you inputting from keyboard or file?: ";
        cin >> choice;
        choice = static_cast<char>( toupper( choice ) );
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        if ( choice == 'F')
        {
            ifstream file;
            string fn;
            cout << "\nPlease enter the name of your names files:  ";
            getline(cin, fn);
            file.open(fn);
            while ( ! file )
            {
                file.close();
                file.clear();
                cout << "\nIm sorry I could open " << '"' << fn << '"'
                    << ". Please enter another name:  ";
                getline(cin, fn);
                file.open(fn);
            }
            cout << "\nFile " << '"' << fn << '"' << " was opened succesful
            string s;
            while (!file.eof() )
            {
                getline(file, s);
                words.push_back(s);
            }
            file.close();
            file.clear();
            done = true;
        }
        else if ( choice == 'K')
        {
            cout << "\nInsert Sentence: ";
            string sen;
            getline(cin,sen);
            words = word_get(sen);
            done = true;
        }
        else
        {
            cout << "You did not input any correct answer."
                 << "Please Try Again\n";
        }


    } while ( ! done );

    bool done1 = false;
    bool center = false;
    bool left = false;
    bool right = false;

    do
    {
        char choice;
        cout << "\nHow do you want to center the word? (Center, Left, or R
        cin >> choice;
        choice = static_cast<char>( toupper( choice ) );
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        if ( choice == 'C')
        {
            center = true;
            done1 = true;
        }
        else if ( choice == 'L')
        {
            left = true;
            done1 = true;
        }
        else if ( choice == 'R')
        {
            right = true;
            done1 = true;
        }
        else
        {
            cout << "You did not input Center, Left, or Right."
                 << " Please Try Again";
        }
    } while ( ! done1 );
    char frame_c;
    bool done2 = false;
    do
    {
        char t;
        cout << "\nWhat would you like your frame be made of? ";
        cin >> t;
        if ( isprint(t))
        {
            frame_c = t;
            done2 = true;
        }
        else
        {
            cout << "You did not input a printable character."
                 << "Please Try Again";
        }
    } while ( ! done2 );

    FrameWord FrameWord(words,frame_c);

    if ( center)
    {
        FrameWord.frame_words();
        cout << "\nFramed Words:\n";
                FrameWord.display_frame();
        cout << '\n';
    }
    else if (left)
    {
        FrameWord.frame_words_left();
        cout << "\nFramed Words:\n";
                FrameWord.display_frame_left();
        cout << '\n';
    }
    else if (right)
    {
        FrameWord.frame_words_right();
        cout << "\nFramed Words:\n";
                FrameWord.display_frame_right();
        cout << '\n';
    }
    return 0;
}

vector<string> word_get(string sen)
{
    vector<string> words;
    vector<string::size_type> num = find(sen, ' ');
    string sen1 = sen;
    for( vector<string::size_type>::size_type p = 0; p != num.size(); ++p)
        {
            string::size_type spaces = find_space(sen1, ' ');
            if ( spaces != sen1.length())
            {
                string word = sen1.substr(0,spaces);
                words.push_back(word);
                sen1 = sen1.substr(spaces + 1, sen1.length());
            }
        }
    string word1 = sen.substr(num[num.size() - 1] + 1, sen.length());
    words.push_back(word1);
    return words;
}
```

```
bj94684@ares:~$ CPP strextra box
box.cpp***
strextra.cpp...
box.cpp: In constructor
‘FrameWord::FrameWord(const
std::vector<std::__cxx11::basic_string<char> >&, char)’:
box.cpp:51:1: warning:
‘FrameWord::framed_words’ should be initialized
in the member initialization list [-Weffc++]
   51 | FrameWord::FrameWord(const vector<string>
   & inputted_words, char f_c)
      | ^~~~~~~~~
box.cpp:51:1: warning:
‘FrameWord::framed_words_left’
should be initialized in the member initialization list
[-Weffc++]
box.cpp:51:1: warning:
‘FrameWord::framed_words_right’
should be initialized in the member initialization list
[-Weffc++]


bj94684@ares:~$ ./box.out

                A Box Upon Ye! Program

Are you inputting from keyboard or file?: sa
You did not input any correct answer.Please Try Again
Are you inputting from keyboard or file?: key

Insert Sentence: O | -+- | /-\

How do you want to center the word? (Center, Left, or Right)? s
You did not input Center, Left, or Right. Please Try Again
How do you want to center the word? (Center, Left, or Right)? center

What would you like your frame be made of? X

Framed Words:
XXXXXXX
X  O  X
X  |  X
X -+- X
```

```
X  |  X
X /-\ X
XXXXXXX

bj94684@ares:~$ ./box.out

                A Box Upon Ye! Program

Are you inputting from keyboard or file?: keyboard

Insert Sentence: I went to eat food

How do you want to center the word? (Center, Left, or Right)? rightt

What would you like your frame be made of? +

Framed Words:
++++++++
+    I +
+ went +
+   to +
+  eat +
+ food +
++++++++

bj94684@ares:~$ ./box.out

                A Box Upon Ye! Program

Are you inputting from keyboard or file?: fi

Please enter the name of your names files:  phrase

Im sorry I could open "phrase". Please enter another name:  phrase.k

File "phrase.k" was opened succesfully

How do you want to center the word? (Center, Left, or Right)? left

What would you like your frame be made of? ^

Framed Words:
^^^^^^^^^
^hello  ^
^world  ^
^       ^
^^^^^^^^^

bj94684@ares:~$ ./box.out

                A Box Upon Ye! Program

Are you inputting from keyboard or file?: fileee
```

```
Please enter the name of your names files:  phrase1.k

File "phrase1.k" was opened succesfully

How do you want to center the word? (Center, Left, or Right)? h
You did not input Center, Left, or Right. Please Try Again
How do you want to center the word? (Center, Left, or Right)? right

What would you like your frame be made of? *

Framed Words:
***********
*     the *
*     sky *
*      is *
* falling *
*         *
***********

bj94684@ares:~$ ./box.out

                A Box Upon Ye! Program

Are you inputting from keyboard or file?: Key

Insert Sentence: the sky is falling

How do you want to center the word? (Center, Left, or Right)? Center

What would you like your frame be made of? @

Framed Words:
@@@@@@@@@@@
@   the   @
@   sky   @
@   is    @
@ falling @
@@@@@@@@@@@

bj94684@ares:~$ exit
exit

Script done on 2023-07-27 12:27:34-05:00 [COMMAND_EXIT_CODE="0"]
```