```
Script started on 2022-12-08 20:53:30-06:00 [TERM="xterm" TTY="/dev/pts/12" COLUMNS
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ pwd
/home/students/j_pec2/JPMainDir/CSC122/Port3/frameProject
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ cat FrameInfo.txt
Jack Pec

CSC122-001

Frame Project

Base level: 5

Add 1.5 for choices of frame types
Add 1 for option configuration thats saved and loaded

Overall level 7.5


Desc:

It's the "A BOX Upon Ye" Projectj_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$
frame.h  FrameInfo.txt  frameMe.txt  frameOut.txt  FrameProjectDriver.cpp  saveOp.t
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ show-code FrameProjectDriver.cp
```

FrameProjectDriver.cpp:

```cpp
#include <iostream>
#include <fstream>

#include "frame.h"

using namespace std;


ofstream openOutputfile();
void closeOutputFile(ofstream & file);
void mainframe();


int main(void)
{

    bool run = true;
    char in;

    cout << "Frame Project:\n";

    do
    {
        mainframe();

        cout << "\nPress q to quit, any other char to continue\n";
        cin >> in;

        if(in == 'q')
        {
            run = false;
        }


    }while(run);

    return 0;
}

void mainframe()
{
    frame source;
    //frame mainFrame;

    char framechoice = '1';
    char indentchoice = '4';
    char choice = '1';
    char choice2 = '0';
    char borderChoice = '*'; //default

    string savename = "saveOp.txt";

    ifstream save_fileIn;


    ofstream outputfile;




    save_fileIn.open(savename);

    if (!save_fileIn)
    {
        save_fileIn.close();
        save_fileIn.clear();

        cerr << "Could not read in save file, A new save file will be\n"
             "made after a getting settings for a frame";

    }
    else
    {

        //save_fileIn >> ws;

        save_fileIn >> framechoice;
        save_fileIn >> indentchoice;
        save_fileIn >> choice;
```

```
        save_fileIn >> choice2;
        save_fileIn >> borderChoice;

        //save_fileIn >> ws;

        /*
        cout << framechoice << "\n";
        cout << indentchoice<< "\n";
        cout << choice<< "\n";
        cout << choice2<< "\n";
        cout << borderChoice<< "\n";
        */

        save_fileIn.close();
        save_fileIn.clear();

    }


    char y;
    cout << "Edit/create options executing program?\n"
         "Press y, otherwise press any key\n"
         "to run program based on previous settings\n";
    cin >> y;
    if(y == 'y')
    {
        cout << "Press 1 for single line frame\n"
             "Press 2 double line frame\n"
             "Press 3 for shaded line frame\n"
             "Press 4 for custom frame from one char\n";


        cin >> framechoice;

        if(framechoice == '4')
        {
            cout << "Enter a char for it's border: ";
            cin >> borderChoice;
        }

        cout << "Press 1 for Left indentation\n"
             "Press 2 for Right indentation\n"
             "Press 3 for Centered indentation\n";


        cin >> indentchoice;

        cout << "Press 1 to enter phrases from file,"
             << " any other number to input manually: ";

        cin >> choice;

        cout << "Press 1 to enter print into a file,"
             << " any other number to print onto screen: ";

        cin >> choice2;
    }

    switch(framechoice)
    {
    case '1':
        source.thisBorder = frame::ChoiceBorder::singleL;
        break;

    case '2':
        source.thisBorder = frame::ChoiceBorder::doubleL;
        break;

    case '3':
        source.thisBorder = frame::ChoiceBorder::shaded;
        break;
    case '4':

        source.thisBorder = frame::ChoiceBorder::defaultB;
        source.set_border(borderChoice);

        break;


    }

    switch(indentchoice)
    {
    case '1':
        source.thisChoice = frame::ChoiceOri::left;
        break;

    case '2':
        source.thisChoice = frame::ChoiceOri::right;
        break;

    case '3':
        source.thisChoice = frame::ChoiceOri::center;
        break;

    default:
        source.thisChoice = frame::ChoiceOri::left;
        break;
    }

    if(choice2 == '1')
    {
        outputfile = openOutputfile();
    }


    if(choice == '1')
    {
```

```
189         ifstream data_file;
190         string fname = "";
191
192         cout << "what's the name of input file?  ";
193         cin >> fname;
194         // }
195         data_file.open(fname);
196
197         while (!data_file)
198         {
199             data_file.close();
200             data_file.clear();
201             cerr << "\n\aCould not open file '" << fname << "'!\n\n";
202             cout << "what's the name of input file?  ";
203             getline(cin, fname);
204             data_file.open(fname);
205         }
206
207         /*
208         if(choice2 == '1')
209         {
210             outputfile = openOutputfile();
211         }
212         */
213
214         data_file >> ws;
215
216         while(!data_file.eof())
217         {
218
219             frame tempy = source;
220
221             if(choice2 == '1')
222             {
223                 data_file >> tempy;
224
225                 outputfile << tempy;
226
227                 data_file >> ws;
228
229             }
230             else
231             {
232                 data_file >> tempy;
233
234                 cout << tempy;
235
236                 data_file >> ws;
237             }
238
239         }
240
241         //closeOutputFile(outputfile);
242         data_file.clear();
243         data_file.close();
244     }
245     else if(choice != '1')
246     {
247
248         if(choice2 == '1')
249         {
250             cout << "Enter Phrase : ";
251             frame k = source;
252
253             cin >> k;
254
255             outputfile << k;
256         }
257         else
258         {
259             cout << "Enter Phrase: ";
260             frame k = source;
261
262             cin >> k;
263
264             cout << k;
265
266         }
267
268     }
269
270     closeOutputFile(outputfile);
271
272     ofstream save_fileOut;
273
274     save_fileOut.open(savename,std::ofstream::trunc);
275
276     /*
277            cout << framechoice << "\n";
278            cout << indentchoice<< "\n";
279            cout << choice<< "\n";
280            cout << choice2<< "\n";
281            cout << borderChoice<< "\n";
282      */
283
284     save_fileOut << framechoice;
285     save_fileOut << indentchoice;
286     save_fileOut << choice;
287     save_fileOut << choice2;
288     save_fileOut << borderChoice;
289
290     save_fileOut.close();
291     save_fileOut.clear();
292
293
294
295 }
296
```

```
297
298 ofstream openOutputfile()
299 {
300     ofstream file;
301     string fname;
302     cout << "What should the Output file name be?  ";
303     //getline(cin, fname);
304     cin >> fname;
305     file.open(fname);
306
307
308     while (!file)
309     {
310         file.close();
311         file.clear();
312         cout << "\nCannot write to '" << fname
313              << "'!!\a\n\nPlease choose "
314              "another name (and/or location):  ";
315         //getline(cin, fname);
316         cin >> fname;
317         file.open(fname);
318     }
319
320     return file;
321
322
323 }
324
325 void closeOutputFile(ofstream & file)
326 {
327     file.close();
328     file.clear();
329
330 }
```

```
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ ls
frame.h  FrameInfo.txt  frameMe.txt  frameOut.txt  FrameProjectDriver.cpp  saveOp.t
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ show-code frame.h
```

frame.h:

```
1  /*
2
3  Frame Lib
4
5
6  */
7  #include <string>
8  #include <sstream>
9  #include <iomanip>
10
11 #pragma once
12
13
14 class frame
15 {
16     std::string val;
17     char borderChar;
18
19
20 public:
21
22     enum class ChoiceOri {left,right,center};
23
24     ChoiceOri thisChoice = ChoiceOri::left; //okay its public
25
26     enum class ChoiceBorder {defaultB,singleL,doubleL,shaded};
27
28     ChoiceBorder thisBorder = ChoiceBorder::defaultB;
29
30     frame(std::string valIn = "", char borderIn = '*')
31         :
32         val(valIn),
33         borderChar(borderIn)
34     {
35
36     }
37
38     void set_border(char b)
39     {
40         borderChar = b;
41     }
42
43     char get_border()
44     {
45         return borderChar;
46     }
47
48
49
50
51
52
53
54     friend std::istream & operator>>(std::istream & in, frame & r)
55     {
56
57         //takes care of reading in whitespace
58         if (in.peek() == '\n')
59         {
60             in.ignore();
61         }
62
63
64         std::getline(in,r.val);
65
66         return in;
```

```
67
68
69      }
70
71      friend std::ostream & operator<<(std::ostream & out,
72                                       const frame & r)
73      {
74          std::string a = "";
75          std::string largestWord = "";
76
77          std::string main = r.val;
78
79          std::istringstream iss;
80
81          // std::string borderStringSide = r.borderChar;
82          std::string borderStringSide(1, r.borderChar);
83
84          iss.str(main);
85
86          do //first pass
87          {
88              iss >> a;
89
90              if(a.length() > largestWord.length())
91              {
92                  largestWord = a;
93
94              }
95
96          }
97          while (!iss.eof());
98
99          iss.clear();
100         iss.str (main);
101
102         std::string borderString;
103
104         switch(r.thisBorder)
105         {
106
107         case ChoiceBorder::defaultB:
108
109             //out << r.borderChar <<" ";
110             borderString =
111                 std::string(largestWord.length()+4, r.borderChar);
112             out << borderString << '\n';
113
114             break;
115
116         case ChoiceBorder::singleL:
117             borderString = "+"
118                            + std::string(largestWord.length()+2, '-') + "+'
119             out << borderString << '\n';
120             borderStringSide = "|";
```

```
121
122
123             break;
124
125         case ChoiceBorder::doubleL:
126             borderString = "+"
127                            + std::string(largestWord.length()+4, '=') + "+'
128             out << borderString << '\n';
129             borderStringSide = "||";
130
131             break;
132
133         case ChoiceBorder::shaded:
134             borderString = "+"
135                            + std::string(largestWord.length()+2, '-') + "+'
136             out << borderString << '\n';
137             borderStringSide = "|";
138
139             break;
140         }
141
142         do
143         {
144             iss >> a;
145
146             out << borderStringSide <<" ";
147
148             switch(r.thisChoice)
149             {
150
151             case ChoiceOri::left:
152
153                 out << std::left
154                     << std::setw(static_cast<int>(largestWord.length()))
155                     << a;
156                 break;
157
158             case ChoiceOri::right:
159
160                 out  << std::right
161                      << std::setw(static_cast<int>(largestWord.length()))
162                      << a;
163                 break;
164
165             case ChoiceOri::center:
166                 out << std::setw(static_cast<int>(
167                                      (largestWord.length()-a.length())/2))
168                     << "" << a
169                     << std::setw(static_cast<int>(
170                                      largestWord.length()-a.length()
171                                      -(largestWord.length()-a.length())/2))
172                     << "";
173                 break;
174             }
```

```
   175
   176              if( r.thisBorder == ChoiceBorder::shaded)
   177              {
   178                  out << " "<< borderStringSide << "*" << '\n';
   179              }
   180              else
   181              {
   182                  out << " "<< borderStringSide  << '\n';
   183              }
   184
   185
   186          }
   187          while (!iss.eof());
   188
   189          if( r.thisBorder == ChoiceBorder::shaded)
   190          {
   191              out << borderString << "*" << '\n';
   192              out << std::string(largestWord.length()+4, '*') << "*" << '\n'
   193          }
   194          else
   195          {
   196              out << borderString <<'\n';
   197          }
   198
   199
   200
   201          return out;
   202
   203      }
   204
   205  };
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ ls
frame.h  FrameInfo.txt  frameMe.txt  frameOut.txt  FrameProjectDriver.cpp  saveOp.t
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ CPP FrameProjectDriver.cpp frame
FrameProjectDriver.cpp***


j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ ./FrameProjectDriver.out
Frame Project:
Edit/create options executing program?
Press y, otherwise press any key
to run program based on previous settings
y
Press 1 for single line frame
Press 2 double line frame
Press 3 for shaded line frame
Press 4 for custom frame from one char
1
Press 1 for Left indentation
Press 2 for Right indentation
Press 3 for Centered indentation
1
Press 1 to enter phrases from file, any other number to input manually: 1
Press 1 to enter print into a file, any other number to print onto screen: 1
What should the Output file name be?  frameOut.txt
what's the name of input file?  frameMe.txt

Press q to quit, any other char to continue
h
Edit/create options executing program?
Press y, otherwise press any key
to run program based on previous settings
y
Press 1 for single line frame
Press 2 double line frame
Press 3 for shaded line frame
Press 4 for custom frame from one char
2
Press 1 for Left indentation
Press 2 for Right indentation
Press 3 for Centered indentation
2
Press 1 to enter phrases from file, any other number to input manually: 1
Press 1 to enter print into a file, any other number to print onto screen: g
what's the name of input file?  frameMe.txt
+=========+
|| hello ||
|| world ||
+=========+
+===========+
||     the ||
||     sky ||
||      is ||
|| falling ||
+===========+
+=======+
||   O ||
||   | ||
|| -+- ||
||   | ||
|| /-\ ||
+=======+

Press q to quit, any other char to continue
h
Edit/create options executing program?
Press y, otherwise press any key
to run program based on previous settings
h
what's the name of input file?  frameMe.txt
+=========+
|| hello ||
|| world ||
+=========+
+===========+
||     the ||
||     sky ||
||      is ||
```

```
|| falling ||
+============+
+=======+
||   O ||
||   | ||
|| -+- ||
||   | ||
|| /-\ ||
+=======+

Press q to quit, any other char to continue
h
Edit/create options executing program?
Press y, otherwise press any key
to run program based on previous settings
y
Press 1 for single line frame
Press 2 double line frame
Press 3 for shaded line frame
Press 4 for custom frame from one char
3
Press 1 for Left indentation
Press 2 for Right indentation
Press 3 for Centered indentation
3
Press 1 to enter phrases from file, any other number to input manually: 1
Press 1 to enter print into a file, any other number to print onto screen: g
what's the name of input file?  frameMe.txt
+-------+
| hello |*
| world |*
+-------+*
**********
+---------+
|   the   |*
|   sky   |*
|   is    |*
| falling |*
+---------+*
***********
+-----+
|  O  |*
|  |  |*
| -+- |*
|  |  |*
| /-\ |*
+-----+*
********

Press q to quit, any other char to continue
h
Edit/create options executing program?
Press y, otherwise press any key
to run program based on previous settings
y
Press 1 for single line frame
Press 2 double line frame
Press 3 for shaded line frame
Press 4 for custom frame from one char
4
Enter a char for it's border: %
Press 1 for Left indentation
Press 2 for Right indentation
Press 3 for Centered indentation
3
Press 1 to enter phrases from file, any other number to input manually: g
Press 1 to enter print into a file, any other number to print onto screen: g
Enter Phrase: Don't Frame Me!
%%%%%%%%%
% Don't %
% Frame %
%  Me!  %
%%%%%%%%%

Press q to quit, any other char to continue
h
Edit/create options executing program?
Press y, otherwise press any key
to run program based on previous settings
y
Press 1 for single line frame
Press 2 double line frame
Press 3 for shaded line frame
Press 4 for custom frame from one char
1
Press 1 for Left indentation
Press 2 for Right indentation
Press 3 for Centered indentation
1
Press 1 to enter phrases from file, any other number to input manually: 1
Press 1 to enter print into a file, any other number to print onto screen: g
what's the name of input file?  frameMe.txt
+-------+
| hello |
| world |
+-------+
+---------+
| the     |
| sky     |
| is      |
| falling |
+---------+
+-----+
| O   |
| |   |
| -+- |
| |   |
| /-\ |
```

```
+-----+

Press q to quit, any other char to continue
q
j_pec2@ares:~/JPMainDir/CSC122/Port3/frameProject$ exit
exit

Script done on 2022-12-08 20:57:03-06:00 [COMMAND_EXIT_CODE="0"]
```