```
/**
  ******************************************************************************
  * @file    : main.cpp
  * @brief   : Main Program
  *          : Lab 1: C++ Console Program
  *          : CS-3210/021
  * @date    : MAR 16 2021
  * @author : Julian Singkham
  ******************************************************************************
  * @attention
  *
  * The purpose of this progarm is to print the coordinates of each facet's verticies
  * given a STL file. A Facet is a triangular polygon that makes up the 3-D object.
  * Additonally at the end the total number of facets, and the max&min values for each
  * axis are printed to stdout.
  *
  * See Main comment for instructions.
  *
  ******************************************************************************
**/

#include "StlFileReader.h"
#include <string>
#include <iostream>
#include <fstream>

using namespace std;

//=====================================Methods=======================================
 /**
  * @brief The program entry point
  *
  * @param args: Pointer to the given arguements
  *        args[1] = Name of STL file
  *
  * @retval NOT USED
  */
 int main(int argc, char **argv){
    //Verify the name of the STL was inputted
    if (argc < 2){
        cout << "Error: Please input the name of the STL file " << endl;
    }
    else{
        char *filename = argv[1];
        stlFileReader fileReader(filename);
    }
    return 0;
}
```

```cpp
/**
  ******************************************************************************
  * @file    : stlFileReader.cpp
  * @brief   : STL File Reader
  *          : Lab 1: C++ Console Program
  *          : CS-3210/021
  * @date    : MAR 16 2021
  * @author : Julian Singkham
  ******************************************************************************
  * @attention
  *
  * This handles reading a STL file and prints out each facet's vertex coordinates to
  * stdout. Additionally the minimum and maximum x,y,z values and the facet count are
  * also recorded.
  *
  ******************************************************************************
**/

#include "StlFileReader.h"
#include <cfloat>
#include <string>
#include <iostream>
#include <fstream>
#include <iomanip>

using namespace std;

//=======================================Methods=====================================


/**
 * @brief Creates STL File Reader object. Additionally, this also uses a member
 *        initializer list to set the values of the stlFileReader variables as seen
 *        in the header file.
 *        DBL_MAX is the maximum value for a float.
 *
 * @param filename: Name of the STL file
 *
 * @retval NONE
 */
stlFileReader::stlFileReader(string filename)
    : max_x(-DBL_MAX), min_x(DBL_MAX), max_y(-DBL_MAX), min_y(DBL_MAX),
      max_z(-DBL_MAX), min_z(DBL_MAX), facets_count(0), filename(filename),
      line_buffer(""), filein(filename.c_str()){
          process_Stl();
      }

/**
 * @brief Reads STIL file and keeps track of how many facets there are, the
 *        max&min vertex values, and prints the vertex values for each facet
 *        to stdout.
 *
 * @param: NONE
 *
 * @retval NONE
 */
void stlFileReader::process_Stl(){
    cout << "Processing " << filename << endl;
    while(filein){
        process_Facet();
    }
    print_Summary();
}

/**
 * @brief Prints the max&min vertex values and facets count.
 *
 * @param: NONE
 *
 * @retval NONE
 */
```

```
72  void stlFileReader::print_Summary(){
73      cout << "================================================" << endl
74           << "Summary" << endl;
75
76      cout << "Total number of facets: " << facets_count << endl;
77      cout << "X Range: " << min_x << " - " << max_x <<endl;
78      cout << "Y Range: " << min_y << " - " << max_y <<endl;
79      cout << "Z Range: " << min_z << " - " << max_z <<endl;
80  }
81
82  /**
83   * @brief Helper function used to parse a facet and prints the information.
84   *
85   * @param: NONE
86   *
87   * @retval NONE
88   */
89  void stlFileReader::process_Facet(){
90      getline(filein, line_buffer);
91
92      //The first check is to see if the end of the STL file has been reached.
93      //The second check is to see if the line contains a character.
94      //The third check is to see if we reached the end of the file
95      if (line_buffer.find("endsolid") == string::npos &&
96          line_buffer.find_first_not_of(" \t\n\v\f\r") != string::npos && !filein.eof()){
97
98          string first_word;
99          stringstream(line_buffer) >> first_word;
100
101         //Skip until facet is found
102         while(first_word != "facet" && !filein.eof()){
103             getline(filein, line_buffer);
104             stringstream(line_buffer) >> first_word;
105         }
106
107         //Code can be added here to deal with facet shading.
108         //Code can be added here to deal with special shapes (like shape in shape)
109
110         //Skip until vertex is found. Stop working if end of file.
111         while(first_word != "vertex" && !filein.eof()){
112             getline(filein, line_buffer);
113             stringstream(line_buffer) >> first_word;
114         }
115
116         //Stop working if end of file.
117         if (!filein.eof()){
118             facets_count++;
119             cout << "Facet #" << facets_count << endl;
120             //Iterate through each axis.
121             for (int i = 0; i < 3; i++){
122                 string vertex;
123                 double x, y, z;
124                 stringstream(line_buffer) >> vertex >> x >> y >> z;
125
126                 cout << "  Vertex " << i+1 << ": (" << x << ", " << y << ", " << z << ")" << endl
    ;
127
128                 max_x = max(x, max_x);
129                 min_x = min(x, min_x);
130
131                 max_y = max(y, max_y);
132                 min_y = min(y, min_y);
133
134                 max_z = max(z, max_z);
135                 min_z = min(z, min_z);
136                 getline(filein, line_buffer);
137             }
138         }
139         cout << endl;
140     }
141 }
```

```
142
143 stlFileReader::~stlFileReader() {
144     filein.close();
145 }
```

```
/**
  ********************************************************************************
  * @file    : stlFileReader.h
  * @brief   : Outline for STL file reader
  *          : Lab 1: C++ Console Program
  *          : CS-3210/021
  * @date    : MAR 16 2021
  * @author : Julian Singkham
  ********************************************************************************
**/

#ifndef STLFILEREADER
#define STLFILEREADER

#include <fstream>
#include <string>
#include <cfloat>

using namespace std;

class stlFileReader{
    public:
        //==================================Methods================================
        /**
         * @brief Creates STL File Reader object
         *
         * @param filename: Name of the STL file
         *
         * @retval NONE
         */
        stlFileReader(string filename);

        ~stlFileReader();

    private:
        //=================================Variables===============================
        double max_x, min_x, max_y, min_y, max_z, min_z;
        int facets_count;
        string filename, line_buffer;
        ifstream filein; //Input only file stream

        //=================================Methods================================

        /**
         * @brief Reads STL file and keeps track of how many facets there are, the
         *        max&min vertex values, and prints the vertex values for each facet
         *        to stdout.
         *
         * @param: NONE
         *
         * @retval NONE
         */
        void process_Stl();

        /**
         * @brief Helper function used to parse a facet and prints the information.
         *
         * @param: NONE
         *
         * @retval NONE
         */
        void process_Facet();

        /**
         * @brief Prints the max&min vertex values and facets count.
         *
         * @param: NONE
         *
         * @retval NONE
         */
        void print_Summary();
```

```
72  };
73  #endif
```

```
CC = g++
CFLAGS = -c -MMD
LFLAGS = -Wall -Wextra -g
LDFLAGS ?= -lglut -lGLU  -lGL
SOURCES = $(wildcard *.cpp)
OBJECTS = $(SOURCES:.cpp=.o)
EXECUTABLE = ex

all: $(EXECUTABLE) clean
$(EXECUTABLE): $(OBJECTS)
    $(CC) $(LFLAGS) -o $@ $(OBJECTS) $(LDFLAGS)

%.o:%.cpp
    $(CC) $(CFLAGS) $<

-include *.d

clean:
    rm -f *.d
    rm -f *.o
```

**Table of Contents**