Editor.h

```
#ifndef EDITOR_H
#define EDITOR_H

#include <iostream>
#include <string>
#include <list>
#include <fstream>
#include <ostream>
#include <iostream>
using namespace std;

class Editor {
public:
    Editor();//后置条件，编辑器为空
    Editor(const string &in, const string &out);
    string parse(const string& line);//如果line为合法命令，则该命令被执行，执行结果被返回。若line是插入的文本行，则该文本行被插入并返回结果；否则返回非法命令错误
    bool run_command();

    /*
    Editor(char *in, char *out);

    }

    */
protected:
    string command_check(const string& line);//后置条件：检查line是否有错误。如果不存在错误，则命令被执行，错误被返回；否则，一个错误信息被返回
    void insert_command(const string& line);//如果line不是很长，则将其插入编辑器，并返回一个空行；否则，返回错误信息。
    void delete_command(int k, int m);//如可能，删除第k至m行，并返回一个空行，否则返回错误信息。
    void line_command(int m);//后置条件：如可能，第m行成为当前行，并返回一个空行；否则，返回错误信息
    void done_command();//编辑器运行结束，并返回文本
    bool str_to_int(string s, int a[], int n);//将s中的数字提取出来放到a[]中，n为数字的个数，如果含有非数字的字符，返回false
    void print_line(int start, int end);//打印从start到end行
    void quit();//退出
    void save();//将文本存盘
    void open();//打开一个新文件
```

```
public:
    list<string> text;//文本列表
    list<string>::iterator current;//当前行的迭代器
    int currentLineNumber;//当前文本行数
    bool inserting;//判断当前操作是否是插入操作
    string in, out;//输入输出文件名
    std::ifstream infile;//输入文件流
    std::ofstream outfile;//输出文件流
};

#endif // !EDITOR.H
```

Editor.cpp

```
#include "Editor.h"

const string PROMPT = "Please enter a line:";
const string NOT_COMMAND = "***Error:This is not a command.";
const string COMMAND_ERROR = "***Error:Command Error";
const string COMMAND_ISVALID = "isValid";
const string COMMAND_QUIT = "command_quit";

Editor::Editor(const string& input,const string& output) {
    /*
    constructor with input and output files
    */
    text = list<string>();
    current = text.begin();
    inserting = true;
    currentLineNumber = -1; //text is empty
    this->in = input;
    this->out = output;
    infile.open(in, ios_base::in); //open the input file

    if (in != "") {
        if (infile.fail()) cout << "***Error:The file does not exist!" << endl;
        else { //read the contents in the input file into text
            string temp;
            while (!infile.eof()) {
                infile >> temp;
                currentLineNumber++;
                text.insert(current, temp);
```

```
                }
                infile.close();
                }
        }
}

bool Editor::run_command() {
        /*
        读取用户命令，并把命令传给调用的函数
        */
        string line;
        do {
                cout << PROMPT << endl;
                getline(cin, line);
        } while (parse(line) != COMMAND_QUIT); //exit when line = $quit
        return true;
}

string Editor::parse(const string& line) {
        /*
        判断字符串是否符合命令格式，是否为Quit命令，若是，返回false
        */
        string s = line;
        //check if it is a command
        if (s[0] != '$') {
                //check if it is inserting
                if (inserting) { //if so, insert line
                        insert_command(line);
                        return COMMAND_ISVALID;
                } else { //if not, invalid
                        cout << NOT_COMMAND << endl;
                        return NOT_COMMAND;
                }
        } else {
                int len = s.length();
                for (int i = 1; i < len; i++) s[i] = tolower(s[i]); //convert the command to lower
case

                switch(s[1]) { //determine which command it is
                case 'i':
                        {
                                if (s != "$insert") {
                                        cout << NOT_COMMAND << endl;
                                        return NOT_COMMAND;
```

```
                }
                //s == $insert
                inserting = true;
                return COMMAND_ISVALID;
            }
            break;
        case 'd':
            {
                if (s.substr(0,8) == "$delete ") {
                    int a[2]; //to store the first and second line number
                    string str = s.substr(8, s.length() - 8);
                    string res = command_check(s); //check if the command is valid
                    if (res != COMMAND_ISVALID) { //if not, output the error
information
                        cout << res << endl;
                        return res;
                    }
                    else if (str_to_int(str, a, 2)) {
                        delete_command(a[0], a[1]);
                        inserting = false;
                        return COMMAND_ISVALID;
                    } else { //str_to_int(str, a, 2) = false
                        cout << COMMAND_ERROR << endl;
                        return COMMAND_ERROR;
                    }

                } else if (s == "$done") {
                    //check if the text is empty
                    if (text.size() != 0) //if not, execute done_command
                        done_command();
                    //if so, print an empty line
                    else cout << endl;
                    inserting = false;
                    return COMMAND_ISVALID;
                } else {
                    cout << NOT_COMMAND << endl;
                    return NOT_COMMAND;
                }
            } break;

        case 'l':
            {
                if (s.substr(0,6) == "$line ") {
                    if (s[6] == '-' && s[7] == '1' && s.length() == 8) { //case -1
```

```
                    line_command(-1);
                    return COMMAND_ISVALID;
                }
                int m[1]; //to store the line number
                string str = s.substr(6, s.length() - 6);
                string res = command_check(s); //check if the command is valid
                if (res != COMMAND_ISVALID) { //if not, output the error
information
                    cout << res << endl;
                    return res;
                }
                else if (str_to_int(str, m, 1)) {
                    line_command(m[0]);
                    return COMMAND_ISVALID;
                } else { //str_to_int(str, m, 1) = false
                    cout << COMMAND_ERROR << endl;
                    return COMMAND_ERROR;
                }
            } else {
                cout << NOT_COMMAND << endl;
                return NOT_COMMAND;
            }
        } break;

    case 'p':
        {
            if (s.substr(0,7) == "$print ") {//命令$print
                int a[2]; //to store the first and second line number
                string str = s.substr(7, s.length() - 7);
                string res = command_check(s); //check if the command is valid
                if (res != COMMAND_ISVALID) { //if not, output the error
information
                    cout << res << endl;
                    return res;
                }
                else if (str_to_int(str, a, 2)) {
                    print_line(a[0], a[1]);
                    inserting = false;
                    return COMMAND_ISVALID;
                } else { ////str_to_int(str, a, 2) = false
                    cout << COMMAND_ERROR << endl;
                    return COMMAND_ERROR;
                }
            } else {
```

```
                cout << NOT_COMMAND << endl;
                return NOT_COMMAND;
            }
        } break;

    case 'q':
        {
            if (s == "$quit") {
                quit();
                inserting = false;
                return COMMAND_QUIT;
            } else {
                cout << NOT_COMMAND << endl;
                return NOT_COMMAND;
            }
        } break;

    case 'o':
        {
            if (s == "$open") {
                open();
                return COMMAND_ISVALID;
            } else {
                cout << NOT_COMMAND << endl;
                return NOT_COMMAND;
            }
        } break;

    case 's':
        {
            if (s == "$save") {
                save();
                return COMMAND_ISVALID;
            } else {
                cout << NOT_COMMAND << endl;
                return NOT_COMMAND;
            }
        } break;

    default: //default case: invalid
        {
            cout << NOT_COMMAND << endl;
            return NOT_COMMAND;
        } break;
```

```
            }
        }
}

bool Editor::str_to_int(string s, int a[], int n) {
    /*
    将s中的数字提取出来放到a[]中，n为数字的个数，如果含有非数字的字符，返回false
    */
    int len = s.length();
    int i = 0, j = 0, count = 0;
    for (;j < n && i < len; j++) {
        count++; //count the number
        a[j] = 0;
        while (s[i] != ' ' && i < len) { //compute each number seperated by space
            if (!isdigit(s[i])) return false;
            a[j] = a[j] * 10 + (s[i] - '0');
            i++;
        }
        i++;
    }
    if (count != n) return false;
    //count = n
    return true;
}

string Editor::command_check(const string& line) { //check if the command is valid
    if (line[2] == 'i') {//命令为$line
        int m;
        int len = line.length() - 6;
        string str = line.substr(6, len);
        int i = 0, count = 0;
        for (; i < len; i++) {
            count++; //count the number
            m = 0;
            while (str[i] != ' ' && i < len) {
                if (!isdigit(str[i]))
                    return "***Error:The command is not followed by a nonnegative integer.";
                //isdigit(str[i]) = true
                m = m * 10 + (str[i] - '0');
                i++;
            }
        }
```

```
            if (count != 1)
                return "***Error:The command is not followed by exactly one integer.";
            else if (m > text.size() - 1)
                return "***Error:The line number is > the last line number.";
            // count = 1 && m < text.size()
            return COMMAND_ISVALID;

        }else if (line[2] == 'e'|| line[2] == 'r') {//命令为$delete或$print
            int ans = 0;
            if (line[2] == 'e') ans = 8;
            //line[2] == 'r'
            else ans = 7;
            int a[2];
            int len = line.length() - ans; //length starting from number
            string str = line.substr(ans, len);
            int i = 0, j = 0, count = 0;
            for (; j < 2 && i < len; j++) {
                count++; //count the number
                a[j] = 0;
                while (str[i] != ' ' && i < len) {
                    if (!isdigit(str[i]))
                        return "***Error:The command is not followed by two
nonnegative integers.";
                    //isdigit(str[i]) = true
                    a[j] = a[j] * 10 + (str[i] - '0');
                    i++;
                }
                i++;
            }

            if (count != 2) return "***Error:The command is not followed by two
nonnegative integers.";
            else if (a[0] > a[1]) return "***Error:The first line number > the second.";
            else if (a[1] > text.size()-1) return "***Error:The second line number > the
last line number.";
            return COMMAND_ISVALID;
        }
        return NOT_COMMAND;
}

void Editor::insert_command(const string& line) {
        currentLineNumber++;
        text.insert(current, line);
}
```

```
void Editor::delete_command(int k, int m) {
	/*
	删除第k行到第m行
	*/
	list<string>::iterator po2 = text.begin();
	list<string>::iterator po3 = text.begin();

	for (int l = 0; l < k; l++) po2++; //refer to the position of k
	for (int l = 0; l < m; l++) po3++; //refer to the position of m
	list<string>::iterator p1 = po2;
	p1 = text.erase(po2, po3); //delete from position k to m-1 if k != m

	if (k != m)
		p1 = text.erase(po3);

	if (k > 0 && currentLineNumber >= k && currentLineNumber <= m) { //current
line is in the lines to delete and k > 0
		current = p1;
		currentLineNumber -= (m-k);
	} else if (k == 0 && currentLineNumber >= k && currentLineNumber <= m)
{ //current line is in the lines to delete and k = 0
		currentLineNumber = -1;
		current = text.begin();
	} else { //current line is not in the lines to delete
		currentLineNumber -= (m-k)+1;
	}
}

void Editor::done_command() {
	/*
	打印文本
	*/
	cout << "Here is the final text:" << endl << endl;
	if (text.size() != 0)
		print_line(0, text.size()-1);

}

void Editor::line_command(int m) {
	list<string>::iterator p = text.begin();
	for (int i = 0; i < m && p != text.end(); i++) {p++;};
	current = ++p;
	currentLineNumber = m;
```

```
    if (m == -1) {
        current = text.begin();
        currentLineNumber = -1;
    }
    inserting = false;
}

void Editor::print_line(int start, int end) {
    list<string>::iterator p = text.begin();
    list<string>::iterator p2 = text.begin();
    for (int i = 0; i < start; i++) p++; //refer to the start position
    for (int i = 0; i < end; i++) p2++; //refer to the end position

    int count = 0;
    while(1) {
        if ( count++    == currentLineNumber - start) cout << ">"; //print > before
current line
        cout << *p << endl;
        if (p != p2) p++;
        //p = p2
        else break;
    }

    return ;
}

void Editor::quit() {//退出
    return;
}

void Editor::save() {//将文本存盘

    if (out == "") {
        cout << "Are you sure to save all the text to the file:(Y/N) " << endl;
        char c;
        cin >> c; //input answer
        if (tolower(c) == 'y') { //save
            cout << "Please enter the file name: " << endl;
            string file;
            cin >> file;
            out = file;
            outfile.open(file);
            if (outfile.fail()) {
                out = "";
```

```
                cout << "***Error:The file does not exist!" << endl;
            } else {
                list<string>::iterator pp = text.begin();
                while (pp != text.end()) {
                    outfile << *pp << endl;
                    pp++;
                }
                outfile.close();
                cin.get();
            }
        }
    }
    else {
        cout << "Are you sure to save all the text to the file:(Y/N) " << endl;
        char c;
        cin >> c; //input answer
        if (tolower(c) == 'y') { //save
            outfile.open(out);
            if (outfile.fail()) {
                cout << "***Error:The file does not exist!" << endl;
            } else {
                list<string>::iterator p = text.begin();
                while (p != text.end()) {
                    outfile << *p << endl;
                    p++;
                }
                outfile.close();
                cin.get();
            }
        }
    }
}


void Editor::open() {//打开一个新文件
    cout << "Are you sure to open a text file:(Y/N) " << endl;
    char c;
    if (cin >> c && tolower(c) == 'y') { //input answer y
        string file;
        if (in == "") {
            cout << "Please enter the file name: " << endl;
            cin >> file;
            in = file;
        }
```

```
        cout << "Would you like to add contents to edit texts or quit and open a
new file? (Add/Open)" << endl;
        string ans;
        cin >> ans; //input answer

        if (tolower(ans[0]) == 'a') {
            if (ans[1] == 'd' && ans[2] == 'd') { //add
                infile.open(in);
                if (infile.fail()) {
                    cout << "***Error:The file does not exist!" << endl;
                }
                else {
                    string temp;
                    while(getline(infile, temp)) {//读取本地文件的内容，添加到list
                        currentLineNumber++;
                        text.insert(current, temp);
                    }
                    infile.close();
                    cin.get();
                }
            } else {
                cout << NOT_COMMAND << endl;
                return;
            }
        } else if (tolower(ans[0] == 'o')){
            if (ans[1] == 'p' && ans[2] == 'e' && ans[3] == 'n') { //open
                currentLineNumber = -1;
                text.clear();
                current = text.begin();
                inserting = true;

                infile.open(in);
                if (infile.fail()) {
                    cout << "***Error:The file does not exist!" << endl;
                }
                else {
                    string temp;
                    while(getline(infile, temp)) {//读取本地文件的内容，添加到list
                        currentLineNumber++;
                        text.insert(current, temp);
                    }
                    infile.close();
                    cin.get();
                }
```

```
            } else {
                cout << NOT_COMMAND << endl;
            }
        } else {
            cout << NOT_COMMAND << endl;
        }
    }
}
```

test.cpp

```
#include <iostream>
#include <cstring>
#include "Editor.h"
using namespace std;

int main() {
    cout
<<"************************************************************" <<
endl;
    cout <<"*
*" << endl;
    cout <<"*
*" << endl;
    cout <<"*                    Simple Text Editor                    *"
<< endl;
    cout <<"*
*" << endl;
    cout <<"*                  Welcome to have a try  !                 *" <<
endl;
    cout <<"*
*" << endl;
    cout <<"*                              Numb: 14346009 14346022
*" << endl;
    cout <<"*                              Name:   李志容     谭笑
*" << endl;
    cout <<"*                              Date:      2016.3.19        *"
<< endl;
    cout
<<"************************************************************" <<
endl;
    cout << endl;
```

```
    //Editor myEditor("E:\\2333.txt", "E:\\2333.txt");
    //string file = "E:\\2333.txt";
    //string in, out;
    //cin >> in >> out;
    //cin.get();
    Editor myEditor("", "");
    myEditor.run_command();

    system("pause");
    return 0;
}
```