#include <iostream>

#include <conio.h>

#include <string>

#include <vector>

#include <fstream>

#include "MyRectangle.h"

#include "coordinates.h"

#include <Windows.h>

using namespace std;

class TextArt : public Coordinates

{

public:

int foreColor, backColor;

int highlight\_forecolor, highlight\_backcolor;

int maximumSizeOfInput;

int xpos, ypos;

int currRow;

vector <string> data;

TextArt(string filename, int x, int y, int fore, int back, int highlighted\_fore, int highlighted\_back);

void Display(bool selectFirstEntry);

void ColorRow(int entry);

void ColorNext();

void ColorPrevious();

int Getsize() { return data.size(); }

string GetEntry() { return data[currRow]; }

};

TextArt::TextArt(string filename, int x, int y, int fore, int back, int highlighted\_fore, int highlighted\_back)

{

string s;

currRow = 0;

foreColor = fore;

backColor = back;

highlight\_forecolor = highlighted\_fore;

highlight\_backcolor = highlighted\_back;

xpos = x;

ypos = y;

ifstream input(filename);

maximumSizeOfInput = 0;

while (input.peek() != -1)

{

getline(input, s);

if (s.length() > maximumSizeOfInput)

maximumSizeOfInput = s.length();

data.push\_back(s);

}

input.close();

}

void TextArt::ColorNext()

{

setForeGroundAndBackGroundColor(foreColor, backColor);

gotoxy(xpos, ypos + currRow);

cout << data[currRow];

currRow = (currRow + 1) % data.size();

setForeGroundAndBackGroundColor(highlight\_forecolor, highlight\_backcolor);

gotoxy(xpos, ypos + currRow);

cout << data[currRow];

}

void TextArt::ColorPrevious()

{

setForeGroundAndBackGroundColor(foreColor, backColor);

gotoxy(xpos, ypos + currRow);

cout << data[currRow];

currRow = currRow - 1;

if (currRow < 0)

currRow = data.size() - 1;

setForeGroundAndBackGroundColor(highlight\_forecolor, highlight\_backcolor);

gotoxy(xpos, ypos + currRow);

cout << data[currRow];

}

void TextArt::ColorRow(int entry)

{

gotoxy(xpos, ypos + entry);

setForeGroundAndBackGroundColor(highlight\_forecolor, highlight\_backcolor);

cout << data[entry];

}

void TextArt::Display(bool selectFirstEntry = true)

{

setForeGroundAndBackGroundColor(foreColor, backColor);

MyRectangle R(xpos - 1, ypos - 1, maximumSizeOfInput + 2, data.size() + 2, 219, true);

// R.Draw();

R.DrawSkelaton(xpos - 1, ypos - 1, maximumSizeOfInput + 2, data.size() + 2);

for (int i = 0; i < data.size(); i++)

{

gotoxy(xpos, ypos + i);

cout << data[i];

}

if (selectFirstEntry)

ColorRow(0);

}

class Node

{

public:

char letter;

Node\* next;

Node\* prev;

Node(char cc)

{

letter = cc;

next = prev = nullptr;

}

};

//take command used

string command;

//which letter is inputted

char parameter;

//start

Node\* start = nullptr;

//total rows create array

Node\* rows[10];

//first row

int currentRow = 0;

//Cursor

Node\* current;

int trackrow;

//deletes letters function

void DeleteLetter(int currentrow, char c)

{

if (currentrow >= 8)

{

return;

}

Node\* start = rows[currentrow]->next;

if (start == nullptr)

return;

while (1)

{

start = rows[currentrow]->next;

while (start != nullptr)

{

// delete first element

if (start->letter == c)

{

if (start->next != nullptr)

{

Node\* a1;

Node\* a2;

a1 = start->next;

a2 = start->prev;

a1->prev = a2;

a2->next = a1;

delete start;

start = a2;

}

else

{

start->prev->next = nullptr;

delete start;

return;

}

}

start = start->next;

if (start == nullptr)

return;

}

}

}

//Delete current row function

void DeleteRow()

{

int size = sizeof(rows) / sizeof(rows[0]);

for (int i = currentRow; i < size; i++) {

rows[i] = rows[i + 1];

}

Node\* start = rows[currentRow]->next;

if (start == nullptr)

return;

while (start->next->next != NULL)

{

start = start->next;

}

current = start->next;

}

void main()

{

//set x and y values for animation

int xCoord = 23, yCoord = 2;

Coordinates C;

//create array of 10

for (int i = 0; i < 10; i++)

{

rows[i] = new Node('.');

rows[i]->next = nullptr;

rows[i]->prev = nullptr;

}

//Open input file

ifstream input("c:\\temp\\input.txt");

while (input.peek() != -1)

{

//Display input file contents

TextArt A("c:\\temp\\input.txt", 30, 2, 2, 0, 2, 0); //(X int, y int, color of letter in box, Color of box, Color of linked list output, background)

A.Display();

//Display arrow for which command is being done

C.gotoxy(xCoord, yCoord);

cout << "--->";

//Show arrow for 300

Sleep(300);

//clear screen

system("CLS");

//take command

input >> command;

if (command == "NL") //make a new line

{

currentRow++;

trackrow++;

}

else if (command == "DL") //delete 1 specific letter

{

input >> parameter;

DeleteLetter(currentRow, parameter);

}

else if (command == "A") //Add letter to line

{

input >> parameter;

Node\* newnode = new Node(parameter);

if (rows[currentRow]->next == nullptr) // if this is our first letter

{

rows[currentRow]->next = newnode;

newnode->prev = rows[currentRow];

current = newnode;

}

else if (current->next == NULL) //add letter to end

{

Node\* end;

end = rows[currentRow]->next;

while (end->next != nullptr)

end = end->next;

end->next = newnode;

newnode->prev = end;

current = newnode; //current follows newnode

}

else //Follow Cursor insert in middle

{

Node\* temp = current->next;

current->next = newnode;

newnode->prev = current;

if (temp != NULL) {

newnode->next = temp;

}

else {

newnode->next = NULL;

}

current = newnode;

}

}

else if (command == "B") //move cursor back (no parameter)

{

//If previous is not null continue

if (current->prev != NULL) {

current = current->prev;

} //crahses when go back at beginning

}

else if (command == "F") //Move cursor foward (no parameter)

{

//If next is not null continue

if (current->next != NULL) {

current = current->next;

}

}

else if (command == "DR")//delete entire row move cursor to previous row (no parameter)

{

DeleteRow();

trackrow--;

}

else if (command == "BS") //backspace (no parameter)

{

Node\* p = rows[currentRow]->next;

Node\* store\_next = rows[currentRow]->next;

Node\* store\_prev = rows[currentRow]->next;

while (p->next != current) //go to current

{

p = p->next;

}

store\_next = p->next; //set to next

store\_prev = p->prev; //set to prev

store\_next->prev = store\_prev; //connect together

store\_prev->next = store\_next;

delete p; //delete node

}

else if (command == "DX") //deletes all letters in document (needs parameter for which letter)

{

input >> parameter;

for (int i = 0; i < 10; i++)

{

DeleteLetter(i, parameter);

}

}

// display linked list

cout << endl;

cout << endl;

cout << endl;

for (int i = 0; i < 8; i++) //loop through linked list

{

Node\* t = rows[i]->next;

if (t != nullptr)

{

while (t != nullptr)

{

cout << t->letter;

t = t->next;

//output cursor

if (t == current )

{

cout << "|";

}

}

cout << endl;

}

}

//Display input file

TextArt B("c:\\temp\\input.txt", 30, 2, 2, 0, 2, 0); //(X int, y int, color of letter in box, Color of box, Color of linked list output, background)

B.Display();

//Increase Y for arrow

yCoord++;

//Slow down loop 500

Sleep(500);

}

input.close();

while (1);

}