





/\*

Program Name: theEditor.cpp

Programmer: Josh Lee

Description: Creates a text editor app through

linked lists. You are able to

\*/

#include <iostream>

#include <string>

#include <stdlib.h>

#include <fstream>

#include <conio.h>

#include <windows.h>

using namespace std;

void gotoxy(int, int);

//creates class Node

class Node {

public:

char c;

Node\* next;

Node\* prev;

Node() {

c = ' ';

next = prev = nullptr;

}

};

void write() {

system("color F0");

Node\* arr[10];

ofstream outputFile;

Node\* curr = nullptr;

Node\* start = nullptr;

Node\* end = nullptr;

char letter = '\0';

int x, y;

x = y = 0;

// Initialize the array of editor lines\linked lists

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

arr[i] = nullptr;

cout << "Please type ESC when you are done." << endl;

// Loop until ESC is entered

while (letter != 27)

{

gotoxy(x, y);

// Read what was typed

letter = \_getch();

//Special character

if (letter == -32)

{

// <- Left 75

letter = \_getch();

if (letter == 75)

{

if (x < 0)

{

//do nothing

}

else if (x != 0 && curr->prev != nullptr)

{

curr = curr->prev;

x--;

}

//left arrow at the beginning of a file

else

{

curr = start;

x = 0;

}

}

// -> Right 77

if (letter == 77)

{

//right arrow as long as there is a node after curr

if (curr->next != nullptr)

{

x++;

curr = curr->next;

}

}

// Up Arrow

if (letter == 72)

{

if (y > 0)

{

//go up

y--;

start = arr[y];

curr = start;

for (int i = 0; i < x-1; i++)

{

curr = curr->next;

}

}

}

//Down Arrow

if (letter == 80)

{

//checks to see if there is a line below and if there is

//it will go down a line

if (y <= 10 && arr[y+1] != nullptr)

{

y++;

start = arr[y];

curr = start;

for (int i = 0; i < x-1; i++)

{

curr = curr->next;

}

}

}

}

//Return

else if (letter == 13)

{

x = 0;

y++;

start = nullptr;

curr = nullptr;

}

//ESC

else if (letter == 27)

{

//do nothing

}

//Backspace

else if (letter == 8)

{

if (curr == start)

{

curr = nullptr;

start = curr;

if (x != 0)

x -= 1;

arr[y]->c = '\0';

}

//deletes carriage return

else if (arr[y] != 0 && curr == start)

{

//go up

y--;

start = arr[y];

curr = start;

while (curr->next)

{

curr = curr->next;

}

}

//deletes at end

else if ((curr->next == nullptr) && (curr->prev))

{

if (curr != nullptr)

{

x--;

curr = curr->prev;

delete(curr->next);

curr->next = nullptr;

}

}

//delete in middle

else if ((curr->next != nullptr) && (curr->prev != nullptr))

{

/\*

Node\* x = curr->next;

curr->prev->next = x;

x->prev = curr->prev;

delete curr;

x = curr;

\*/

curr->prev->next = curr->next;

curr->next->prev = curr->prev;

Node\* tmp = curr->next;

delete(curr);

curr = tmp;

x--;

}

}

//regular character (number, character, special symbols)

else

{

//first character of file

if (start == nullptr)

{

Node\* x = new Node();

x->c = letter;

start = x;

curr = start;

arr[y] = start;

}

else

{

//inserts at front after typing

if (x == 0)

{

// Create a new node and assign the character

Node\* x = new Node();

x->c = letter;

// Make the new node the 'start'

// and make its next the previous start's node

x->next = start;

start->prev = x;

x->prev = nullptr;

start = x;

arr[y] = start;

}

//insert at end

else if (curr->next == nullptr)

{

Node\* x = new Node();

x->c = letter;

curr->next = x;

x->prev = curr;

curr = curr->next; //move current

}

//insert character in middle

else

{

Node\* x = new Node();

x->c = letter;

x->next = curr->next;

curr->next->prev = x;

curr->next = x;

x->prev = curr;

curr = x;

}

}

cout << letter;

x++;

}

system("cls");

//prints linked list

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

{

if (arr[i] != nullptr)

{

Node\* p;

p = arr[i];

while (p != nullptr)

{

cout << p->c;

p = p->next;

}

cout << endl;

}

}

gotoxy(x, y);

}

//save file

char input;

cout << endl << endl << "Would you like to save? (y/n)" << endl;

cin >> input;

if (input == 'y')

{

string fileName;

cout << "Name of file: ";

cin >> fileName;

ofstream myFile;

myFile.open(fileName, ios::app);

Node\* nodePtr;

nodePtr = start;

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

{

if (arr[i] != nullptr)

{

Node\* p;

p = arr[i];

while (p != nullptr)

{

myFile << p->c;

p = p->next;

}

myFile << endl;

}

}

gotoxy(x, y);

myFile.close();

}

else {

exit(0);

}

}

void ShowConsoleCursor(bool showFlag)

{

HANDLE out = GetStdHandle(STD\_OUTPUT\_HANDLE);

CONSOLE\_CURSOR\_INFO cursorInfo;

GetConsoleCursorInfo(out, &cursorInfo);

cursorInfo.bVisible = showFlag; // set the cursor visibility

SetConsoleCursorInfo(out, &cursorInfo);

}

//displays menu

int displayMenu() {

//Press Spacebar to Select Option

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 143);

cout << "[FILE]" << " " << "[EDIT]" << " " << "[FORMAT]" << " " << "[VIEW]" << " " << "[HELP]" << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " " << " ";

//New = 1, Open.. = 2, X Close = 3

int option = 1;

char c;

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

cout << endl;

gotoxy(56, 13);

cout << " New " << endl;

gotoxy(54, 14);

cout << " Open... " << endl;

gotoxy(55, 15);

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 252);

cout << " X Close " << endl;

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

gotoxy(43, 20);

cout << "Press Spacebar to select an option";

while (1) {

if (\_kbhit()) {

c = \_getch();

if (c == 32) {

break;

}

if (c == -32) {

c = \_getch();

if (c == 72) {

if (option > 1) {

gotoxy(56, 13);

cout << " New " << endl;

gotoxy(54, 14);

cout << " Open... " << endl;

gotoxy(54, 15);

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 252);

cout << " X Close " << endl;

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

option--;

}

}

else if (c == 80) {

if (option < 3) {

gotoxy(56, 13);

cout << " New " << endl;

gotoxy(54, 14);

cout << " Open... " << endl;

gotoxy(54, 15);

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 252);

cout << " X Close " << endl;

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

option++;

}

}

}

}

//creates blinking animation for option when user navigates to the chosen option

if (option == 1) {

gotoxy(56, 13);

cout << " ";

Sleep(500);

gotoxy(56, 13);

cout << "> New <";

Sleep(500);

}

if (option == 2) {

gotoxy(54, 14);

cout << " ";

Sleep(500);

gotoxy(54, 14);

cout << "> Open... <";

Sleep(500);

}

if (option == 3) {

gotoxy(54, 15);

cout << " ";

Sleep(500);

gotoxy(54, 15);

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 252);

cout << "> X Close <";

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

Sleep(500);

}

}

return option;

}

void open() {

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

char c;

string fileName;

cout << "What is the path of your file?" << endl;

cin >> fileName;

system("cls");

ifstream openFile;

openFile.open(fileName, ios::app);

Node\* nodePtr;

Node\* arr[10];

Node\* newNode = new Node();

Node\* head = nullptr;

string line, text;

newNode->next = NULL;

nodePtr = NULL;

getline(openFile, text);

while (!openFile.eof())

{

while (openFile >> text)

cout << text << " ";

}

cout << text << endl;

write();

//convertString();

}

// Driver program

int main()

{

system("color F0");

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), 240);

int option = displayMenu();

switch (option)

{

case 1:

system("cls");

write();

break;

case 2:

system("cls");

open();

break;

case 3:

exit(0);

}

system("pause");

return 0;

}

void gotoxy(int x, int y)

{

COORD pos = { x, y };

HANDLE output = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleCursorPosition(output, pos);

}