**Header file.h**

#include<iostream>

#include<string.h>

#include<windows.h>

#include<limits.h>

using namespace std;

class NODE

{

public:

int value;

string DATA;

NODE \*LEFT;

NODE \*RIGHT;

NODE \*TOP;

NODE \*BOTTOM;

NODE(); //CONSTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

~NODE(); //DESTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

};

class NOTEPAD

{

public:

NODE \*ROOT;

string arr[1000];

NOTEPAD();//CONSTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void filling();

void insert(int X, int Y); //ADD A TEXT FUNCTION

void delet(string x);

void search();

string copy(int X, int Y);

void paste(int X, int Y, string word);

void find(string word);

int count();

void filled();

void display();

void undo();

void push();

~NOTEPAD(); //DESTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

};

void maker();

**IMPLEMENTATION FILE.CPP**

#include"SOURCE.h"

//CONSTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NODE::NODE()

{

value = 0;

DATA = " ";

LEFT = NULL;

RIGHT = NULL;

TOP = NULL;

BOTTOM = NULL;

}

//DESTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NODE::~NODE()

{

free(LEFT);

free(RIGHT);

free(TOP);

free(BOTTOM);

}

//CONSTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NOTEPAD::NOTEPAD()

{

ROOT =NULL;

for (int i = 0; i < 1000; i++)

{

arr[i] = " ";

}

}

//FILLING NOTEPAD WITH NODES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void NOTEPAD::filling()

{

int num = 1;

for (int i = 1; i <= 1000; i++)

{

NODE \*n1 = new NODE();

n1->value = num;

num++;

if (ROOT == NULL)

{

ROOT = n1;

}

else

{

NODE\* temp = ROOT;

while (temp->RIGHT!=NULL)

{

temp = temp->RIGHT;

}

temp->RIGHT = n1;

n1->LEFT = temp;

}

}

NODE\* ptr = ROOT;

for (int i = 1; i <= 50; i++)

{

ptr = ptr->RIGHT;

}

NODE\* temp = ROOT;

while (ptr != NULL)

{

temp->BOTTOM = ptr;

ptr->TOP = temp;

temp = temp->RIGHT;

ptr = ptr->RIGHT;

}

}

//ADD A TEXT FUNCTION

void NOTEPAD::insert(int X, int Y)

{

string str;

cout << "Enter the word to be stored inside the node: ";

cin >> str;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> str;

}

NODE\* cursor = ROOT;

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

{

cursor = cursor->RIGHT;

}

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

{

cursor = cursor->BOTTOM;

}

cursor->DATA = str;

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT INSERTED.\n";

Sleep(1000);

}

//FUNCTION TO DELETE USER DESIRED WORD

void NOTEPAD::delet(string x)

{

bool flag = false;

NODE\* temp = ROOT;

while (temp->RIGHT != NULL)

{

if (x == temp->DATA)

{

temp->DATA = " ";

flag = true;

}

temp = temp->RIGHT;

}

if (flag == false)

{

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT NOT FOUND.\n";

}

else

{

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT DELETED.\n";

}

Sleep(1000);

}

//FUNCTION TO SERCH THE USER DESIRED WORD

void NOTEPAD::search()

{

bool fon = false;

string str;

cout << "Enter Data To Search : ";

cin >> str;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> str;

}

NODE\* ptr = ROOT;

cout << endl;

while (ptr != NULL)

{

if (ptr->DATA == str)

{

fon = true;

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT FOUND.\n";

}

ptr = ptr->RIGHT;

}

if (fon == false)

{

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT NOT FOUND.\n";

}

Sleep(2000);

}

//function to copy the word present at user enter position

string NOTEPAD::copy(int X, int Y)

{

NODE\* cursor = ROOT;

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

{

cursor = cursor->RIGHT;

}

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

{

cursor = cursor->BOTTOM;

}

return cursor->DATA;

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT COPPIED.\n";

Sleep(1000);

}

//FUNCTION TO PASTE THE USED COPIED TEXT AT HIS GIVEN LOCATION

void NOTEPAD::paste(int X, int Y, string word)

{

NODE\* cursor = ROOT;

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

{

cursor = cursor->RIGHT;

}

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

{

cursor = cursor->BOTTOM;

}

cursor->DATA=word;

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TEXT PASTED SUCCESSFULLY.\n";

Sleep(1000);

}

//FUNCTION TO FIND AND REPLACE THE WORD

void NOTEPAD::find(string word)

{

char choice;

NODE\* ptr = ROOT;

bool foun = false;

while (ptr != NULL && foun!=true)

{

if (ptr->DATA == word)

{

foun = true;

}

ptr = ptr->RIGHT;

}

if (foun == true)

{

ptr = ROOT;

string replace;

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*WORD FOUND.\n";

cout << "ENTER ITS REPLACEMENT: ";

cin >> replace;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> replace;

}

cout << "ENTER (Y / y) IF YOU WANT TO REPLACE THE WORD IN THE WHOLE FILE\n"

<< " & (N / n) IF YOU WANT TO JUST REPLACE THE FIRST WORD : ";

cin >> choice;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> choice;

}

if (choice == 'N' || choice == 'n')

{

while (ptr != NULL)

{

if (ptr->DATA == word)

{

ptr->DATA = replace;

break;

}

ptr = ptr->RIGHT;

}

}

else if (choice == 'Y' || choice == 'y')

{

while (ptr != NULL)

{

if (ptr->DATA == word)

{

ptr->DATA=replace;

}

ptr = ptr->RIGHT;

}

}

}

}

//FUNCTION TO COUNT THE TOTAL NUMBER OF WORDS

int NOTEPAD::count()

{

int num = 0;

NODE \*ptr = ROOT;

while (ptr!=NULL)

{

if (ptr->DATA != " ")

{

num++;

}

ptr = ptr->RIGHT;

}

return num;

}

//FUNCTION TO DISPLAY ALL THE ORDERS PAIRS CONTAINING A WORD

void NOTEPAD::filled()

{

int num = 0;

cout << "\n--------------------------------------------------------------------------\n";

cout << "FOLLOWING ORDERED PAIRS ARE FILLED.\n";

NODE\* ptr = ROOT;

int x=0, y=0;

while (ptr != NULL)

{

if (x >= 50)

{

x = 0;

y++;

}

if (ptr->DATA != " ")

{

num++;

cout << " ( " << x << " , " << y << " ) ";

}

x++;

ptr = ptr->RIGHT;

}

if (num == 0)

{

cout << "\tNONE.\n";

}

}

//DISPLAY FUNCTION

void NOTEPAD::display()

{

NODE \*TEMP =ROOT ;

for (int i = 1; i <= 1000; i++)

{

//cout << TEMP->value;

cout << TEMP->DATA;

cout << " ";

TEMP = TEMP->RIGHT;

if (i > 0 && i % 50 == 0)

{

cout << endl;

}

}

}

//FUNCTION TO PUSH ALL THE ELEMENTS FROM THE NOTEPAD TO A STRING ARRAY

void NOTEPAD::push()

{

NODE \*ptr = ROOT;

int i = 0;

while (ptr != NULL)

{

arr[i] = ptr->DATA;

ptr = ptr->RIGHT;

i++;

}

}

//UNDO FUNCTION

void NOTEPAD::undo()

{

NODE\* ptr = ROOT;

int i = 0;

while (ptr != NULL)

{

ptr->DATA=arr[i];

ptr = ptr->RIGHT;

i++;

}

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*UNDO OPERATION COMPLETED SUCCESSFULLY.\n";

Sleep(1000);

}

//DESTRUCTOR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NOTEPAD::~NOTEPAD()

{

free(ROOT);

}

void maker()

{

cout << "\n\n\n\n\n";

cout << "\t\t\t\t---------------------------------------------------\n";

cout << "\t\t\t\t\t\tNOTEPAD PROJECT\n\n";

cout << "\t\t\t\t\t20F-0329\tZAEEM MUHAMMAD YASEEN\n\n";

cout << "\t\t\t\t\t20F-0297\tHAMZA SAJJAD\n\n";

cout << "\t\t\t\t\t20F-0242\tAFFAN SHAHID CHOUDARY\n";

cout << "\t\t\t\t---------------------------------------------------\n";

}

**MAIN.CPP file**

#include"SOURCE.h"

int main()

{

maker();

Sleep(1000);

NOTEPAD p;

string find, copy, n;

bool found;

int userwill=0, x=0, y=0;

p.filling();

do

{

Sleep(1000);

system("cls");

cout << "NOTEPAD.\n";

p.display();

p.filled();

cout << "\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << "\n\t\tWhat action do u want to perform & press '0' to EXIT:\n";

cout << "\t\t1) ADD TEXT.\t";

cout << "2) DELETE TEXT.\n";

cout << "\t\t3) SEARCH TEXT.\t";

cout << "4) COPY TEXT.\n";

cout << "\t\t5) PASTE TEXT.\t";

cout << "6) FIND & REPLACE TEXT.\n";

cout << "\t\t7) UNDO TEXT.\t";

cout << "8) COUNT TOTAL TEXT.\n";

cin >> userwill;

switch (userwill)

{

case 0:

break;

case 1:

p.push();

cout << "Enter the (x,y) cordinates [minimum (0,0), maximum(49,19)] where you want to put the text;\n";

cout << "x: ";

cin >> x;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> x;

}

while (x < 0 || x > 49)

{

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> x;

}

cout << "y: ";

cin >> y;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> y;

}

while (y < 0 || y > 19)

{

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> y;

}

p.insert(x, y);

break;

case 2:

cout << "Enter the string you want to delete " << endl;

cin >> n;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> n;

}

p.delet(n);

break;

case 3:

p.search();

break;

case 4:

cout << "Enter the (x,y) cordinates whose text you want to copy;\n";

cout << "x: ";

cin >> x;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> x;

}

while (x < 0 || x > 49)

{

cout << "OUT OF BOUND. ENTER AGAIN.";

cin >> x;

}

cout << "y: ";

cin >> y;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> y;

}

while (y < 0 || y > 19)

{

cout << "OUT OF BOUND. ENTER AGAIN.";

cin >> y;

}

copy = p.copy(x, y);

break;

case 5:

p.push();

if (copy != "")

{

cout << "Enter the (x,y) cordinates where you want to paste the copied text;\n";

cout << "x: ";

cin >> x;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> x;

}

while (x < 0 || x > 49)

{

cout << "OUT OF BOUND. ENTER AGAIN.";

cin >> x;

}

cout << "y: ";

cin >> y;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> y;

}

while (y < 0 || y > 19)

{

cout << "OUT OF BOUND. ENTER AGAIN.";

cin >> y;

}

p.paste(x,y,copy);

}

else

cout << "\nERROR! NOTHING COPIED YET.\n";

break;

case 6:

p.push();

cout << "\nEnter the text which you want to find: ";

cin >> find;

while (cin.fail())

{

cin.clear();

cin.ignore();

cout << "ERROR! ENTER AGAIN.\n" << endl;

cin >> find;

}

p.find(find);

break;

case 7:

p.undo();

break;

case 8:

cout << "TOTAL NUMBER OF WORDS ARE: " << p.count();

break;

default:

cout << "------------------------------ERROR: WRONG INPUT.------------------------------\n";

break;

}

} while (userwill != 0);

system("pause");

return 0;

}