Сделать скрины результатов выполнения работы.

//main1.cpp

#include <iostream>

#include <climits> //for INT\_MAX

#include <ctime> //for time()

#include <cstdlib> //for srand(),rand()

using namespace std;

int \*\* make\_matr(int n);

void print\_matr(int \*\* matr, int n);

void delete\_matr(int \*\* matr, int n);

void fill\_matr(int \*\* matr, int n);

int \*\* delete\_min(int \*\* matr, int n);

int main() {

int \*\* oMatr, \*\*fMatr;

int n;

cout << "Enter size of matr(NxN): ";

cin >> n;

oMatr = make\_matr(n);

fill\_matr(oMatr,n);

cout << "Original Matrix: \n";

print\_matr(oMatr,n);

fMatr = delete\_min(oMatr, n);

cout << "Final Matrix(after deleting row and col that contains min value):\n";

print\_matr(fMatr,n-1);

delete\_matr(oMatr,n);

delete\_matr(fMatr,n-1);

}

int \*\* make\_matr(int n) {

int \*\* matr;

matr = new int\*[n];

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

matr[i] = new int[n];

return matr;

}

void print\_matr(int \*\* matr, int n) {

for (int i = 0; i < n; ++i) {

for (int j = 0; j < n; ++j) {

cout << matr[i][j] << " ";

}

cout << '\n';

}

}

void delete\_matr(int \*\* matr, int n) {

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

delete matr[i];

delete matr;

matr = NULL;

}

void fill\_matr(int \*\* matr, int n) {

srand(time(NULL));

int mod = 90;

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

for (int j = 0; j < n; ++j)

matr[i][j] = rand()%mod + 10;

}

int \*\* delete\_min(int \*\* matr, int n) {

int \*\* rezult = make\_matr(n-1);

int imin=0, jmin=0, min = INT\_MAX;

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

for (int j = 0; j < n; ++j)

if (matr[i][j] < min) {

min = matr[i][j];

imin = i;

jmin = j;

}

for (int i = 0, ri = 0; i < n; ++i) {

if (i == imin) continue;

for (int j = 0, rj = 0; j < n; ++j) {

if (j == jmin) continue;

rezult[ri][rj] = matr[i][j];

++rj;

}

++ri;

}

return rezult;

}

//main2.cpp

#include <iostream>

#include <cstring>

#include <fstream>

using namespace std;

struct Node {

char \* key;

Node \* next;

Node \* prev;

Node(const char \* str):next(NULL),prev(NULL) {

int len = strlen(str);

key = new char[len+1];

strcpy(key,str);

}

~Node() { delete [] key; }

};

struct List {

int size;

Node \* first;

Node \* last;

List():size(0),first(NULL),last(NULL) {} //конструктор по умолчанию. Присваиваем элементам структуры нулевые значения

~List();

void add(int position, char \* str);

void del(int position);

void print();

void printToFile(ofstream & fout);

void restorFromFile(ifstream & fin);

};

List::~List() {

Node \* temp;

while (size > 0) {

temp = first;

first = first->next;

delete temp;

size--;

}

first = NULL;

last = NULL;

}

void List::add(int position, char \* str) {

Node \*el = new Node(str);

if (position == 0) {

el->next = first;

if (size != 0) first->prev = el;

first = el;

if (size == 0) last = first;

}

else if (position == size) {

el->prev = last;

last->next = el;

last = el;

}

else {

Node \* temp = first;

int p = 0;

while (p<position) {

temp = temp->next;

++p;

}

temp->prev->next = el;

el->prev = temp->prev;

temp->prev = el;

el->next = temp;

}

++size;

}

void List::del(int position) {

Node \* temp;

if (position == 0) {

temp = first;

first = first->next;

first->prev = NULL;

delete temp;

}

else if (position == size-1) {

temp = last;

last = last->prev;

last->next = NULL;

delete temp;

}

else {

int p = 0;

temp = first;

while (p < position) {

temp = temp->next;

++p;

}

temp->next->prev = temp->prev;

temp->prev->next = temp->next;

delete temp;

}

--size;

}

void List::print() {

Node \*temp = first;

for (int i = 0; i < size; ++i) {

cout << i <<": " << temp->key << "\n";

temp = temp->next;

}

if (size == 0) cout << "List is Empty!\n";

}

void List::printToFile(ofstream & fout) {

Node \* temp = first;

while (temp != NULL) {

fout << temp->key <<'\n';

temp = temp->next;

}

}

void List::restorFromFile(ifstream & fin) {

this->~List(); //удаляем содержимое списка

char temp[256];

while (fin.getline(temp,256)) {

add(size,temp);

}

}

int main() {

List list;//создается пустой список

list.print(); //печать пустого списка

int k, pos;

char temp[256];

cout << "\nHow many elements do you want to add to list?: ";

cin >> k;

while (k--) {

cout << "Enter pos to add(from 0 to " << list.size << "):";

cin >> pos;

cin.get();

cout << "Enter string:\n";

cin.getline(temp,256);

list.add(pos,temp);

}

list.print();

cout << "\nHow many elements do you want to delete from list? ";

cin >> k;

cout << "Enter " << k <<" numbers in ascending order(from 0 to " << list.size-1 << "):\n";

int mas[256];

for (int i = 0; i < k; ++i) cin >> mas[i];

for (int i = 0; i < k; ++i) list.del(mas[i] - i);

cout << "\nList after remove operations:\n";

list.print();

char filename[] = "data.txt";

ofstream fout(filename);

cout << "\nRecording list to "<< filename << "...\n";

list.printToFile(fout);

list.~List(); //удаление списка

list.print();

fout.close();

ifstream fin(filename);

cout << "\nRestoring list from " << filename << "...\n";

list.restorFromFile(fin);

list.print();

list.~List();

}