#include "hash\_table.h"

#include <iostream>

#include <fstream>

#include <chrono>

#include <random>

using namespace std;

void generate\_file(ofstream& of, int count\_lines);

void main() {

srand(time(0));

setlocale(LC\_ALL, "rus");

hash\_table\* table = new hash\_table(0);

string path\_in = "input.txt";

string path\_out = "output.txt";

int count\_lines = 100000;

int command\_number = 0;

ofstream wf("input.dat", ios::out | ios::binary);

generate\_file(wf,count\_lines);

do

{

patient\* tmp = new patient();

std::cout << "Enter command code: "

<< "\n0. Enter new table"

<< "\n1. Add antoher patient"

<< "\n2. Delete patient from file"

<< "\n3. Find patient"

<< "\n4. Display table"

<< "\n5. Clear table"

<< "\nexit : -1"

<< "\nYour input: ";

std::cin >> command\_number;

switch (command\_number)

{

case 0:

{

if (!table->isEmpty()) table->clear();

cout << "\nLocate the file: ";

cin >> path\_in;

ifstream fin(path\_in, ios::binary | ios::in);

table->set\_file\_path(path\_in);

int index = 0;

int prev\_card\_number = 0;

bool get\_the\_numb = true;

if (fin.is\_open()) {

while (!fin.eof()) {

fin.read((char\*)&(\*tmp), sizeof(patient));

if (prev\_card\_number == tmp->card\_number) continue;

prev\_card\_number = tmp->card\_number;

bool res = table->add(\*tmp, fin.tellg()/sizeof(patient) - 1 );

if (res)

{

index++;

if (index >= (count\_lines / 2) and get\_the\_numb) {

get\_the\_numb = false;

cout << tmp->card\_number << " " << tmp->illness\_code << " " << tmp->doctor\_surname << " " << index-1 << "\n";

}

}

}

}

fin.close();

break;

}

case 1:

{

cout << "\nEnter patient's info: ";

cin >> tmp->card\_number >> tmp->illness\_code >> tmp->doctor\_surname;

tmp->index = table->get\_current\_size();

ofstream fin(table->get\_file\_path(), ios::binary | ios::out | ios::app);

fin.write((char\*)&(\*tmp), sizeof(patient));

table->add(\*tmp, table->get\_current\_size());

break;

}

case 2:

{

int card\_number;

cout << "\nEnter patient's card number: ";

cin >> card\_number;

table->delete\_field(card\_number);

table->display\_all\_file(table->get\_file\_path());

table->clear();

ifstream fin(table->get\_file\_path() , ios::binary | ios::in);

int index = 0;

int prev\_card\_number = 0;

if (fin.is\_open()) {

while (!fin.eof()) {

fin.read((char\*)&tmp, sizeof(patient));

if (prev\_card\_number == tmp->card\_number) continue;

prev\_card\_number = tmp->card\_number;

if (table->add(\*tmp, index)) index++;

}

}

fin.close();

break;

}

case 3:

{

int card\_number;

cout << "\nEnter patient's card number: ";

cin >> card\_number;

int start\_time = clock();

auto start = chrono::high\_resolution\_clock::now();

tmp = table->get\_patient(card\_number);

if (tmp->card\_number == -1)

{

cout << "He is not our patient\n";

continue;

}

ifstream fdirect(table->get\_file\_path(), ios::in | ios::binary);

fdirect.seekg((int)(tmp->index)\*sizeof(patient), ios::beg);

fdirect.read((char\*)&(\*tmp), sizeof(patient));

fdirect.close();

auto elapsed = chrono::high\_resolution\_clock::now() - start;

long long microseconds = std::chrono::duration\_cast<std::chrono::microseconds>(elapsed).count();

cout << tmp->card\_number << " " << tmp->illness\_code << " " << tmp->doctor\_surname << " " << tmp->index << " " << " //time - " << microseconds << "\n";

break;

}

case 4:

{

table->display\_all();

break;

}

case 5:

{

table->clear();

}

case -1:

return;

}

std::cout << std::endl;

} while (command\_number != -1);

}