Seminario de solución de problemas de Estructuras de Datos II

Clave: I5889

NRC: 59623

Calendario: 2016B

Sección: D10

Actividad 02: Diagramas de casos

Alumno: Navarro Presas Moisés Alejandro

Código del Alumno: 215861509

# main.cpp

#include <fstream>

#include "cli.hpp"

#include "email.hpp"

using namespace std;

#define EMAIL\_FILE\_PATH "email.bd"

int main(int argc, char const \*argv[])

{

short opc;

do

{

clear\_screen();

display\_menu();

opc = get\_int();

switch (opc)

{

case OPC\_EXIT:

{

break;

}

case OPC\_WRITE:

{

Email email;

fill(email);

ofstream file(EMAIL\_FILE\_PATH, ios::out);

if (!file.is\_open())

{

msg(FILE\_ERROR);

break;

}

file.seekp(email.get\_id() \* sizeof(Email), ios\_base::beg);

file.write((char\*)&email, sizeof(Email));

file.close();

msg(MSG\_DONE);

break;

}

case OPC\_READ:

{

Email email;

int id = get\_int("ID>");

ifstream file(EMAIL\_FILE\_PATH, ios::in);

if (!file.is\_open())

{

msg(FILE\_ERROR);

break;

}

file.seekg(id \* sizeof(Email), ios\_base::beg);

file.read((char\*)&email, sizeof(Email));

file.close();

if (!file.eof() && email.get\_id() != 0)

{

display(email);

}

else

{

msg(MSG\_NOT\_FOUND);

}

break;

}

case OPC\_MODIFY:

{

Email email;

int id = get\_int("ID>");

fstream file(EMAIL\_FILE\_PATH, ios::in | ios::out);

if (!file.is\_open())

{

msg(FILE\_ERROR);

break;

}

file.seekg(id \* sizeof(Email), ios\_base::beg);

file.read((char\*)&email, sizeof(Email));

if (!file.eof() && email.get\_id() != 0)

{

display(email);

msg("1) Fecha\n");

msg("2) Hora\n");

msg("3) Remitente\n");

msg("4) Destinatario\n");

msg("5) CC\n");

msg("6) BCC\n");

msg("7) Asunto\n");

msg("8) Contenido\n");

switch (get\_int())

{

case 1:

email.set\_date(get\_sring().c\_str());

break;

case 2:

email.set\_time(get\_sring().c\_str());

break;

case 3:

email.set\_from(get\_sring().c\_str());

break;

case 4:

email.set\_to(get\_sring().c\_str());

break;

case 5:

email.set\_cc(get\_sring().c\_str());

break;

case 6:

email.set\_bcc(get\_sring().c\_str());

break;

case 7:

email.set\_subject(get\_sring().c\_str());

break;

case 8:

email.set\_content(get\_text().c\_str());

break;

default:

msg(INVALID\_OPTION);

}

file.seekp(email.get\_id() \* sizeof(Email), ios\_base::beg);

file.write((char\*)&email, sizeof(Email));

file.close();

}

else

{

msg(MSG\_NOT\_FOUND);

}

break;

}

default:

{

msg(INVALID\_OPTION);

break;

}

}

pause();

} while(opc != OPC\_EXIT);

return 0;

}

# cli.hpp

#ifndef CLI\_HPP

#define CLI\_HPP

#include <iostream>

#include "email.hpp"

using namespace std;

#ifdef \_WIN32

#define CLEAR "cls"

#else

#define CLEAR "clear"

#endif

#define OPC\_EXIT 0

#define OPC\_WRITE 1

#define OPC\_READ 2

#define OPC\_MODIFY 3

#define INVALID\_OPTION "Opcion invalida\n"

#define MSG\_DONE "Listo\n"

#define MSG\_NOT\_FOUND "No encontrado\n"

#define FILE\_ERROR "Error en el archivo\n"

#define MSG\_PAUSE "Presione entrar para continuar . . ."

void display\_menu()

{

cout << OPC\_WRITE << ") Escribir" << endl;

cout << OPC\_READ << ") Leer" << endl;

cout << OPC\_MODIFY << ") Modificar" << endl;

cout << OPC\_EXIT << ") Salir" << endl;

}

int get\_int(string msg = ">", int def = -1) {

int i;

cout << msg;

if(!(cin >> i)) {

cin.clear();

cin.ignore();

i = def;

}

cin.ignore();

return i;

}

string get\_sring(string msg = ">")

{

string s;

std::cout << msg;

cin >> s;

return s;

}

string get\_text(string msg = ">")

{

string s, buffer;

std::cout << msg << endl;

int n = get\_int(" Lienas>");

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

{

getline(cin, buffer);

s += buffer + '\n';

}

return s;

}

void clear\_screen() {

system(CLEAR);

}

void msg(string m) {

cout << m;

}

void pause() {

cout << MSG\_PAUSE;

cin.ignore();

}

/\* ---------------------------------------------------------------------------\*/

void fill(Email& email)

{

email.set\_id(get\_int("ID>"));

email.set\_date(get\_sring("Fecha (yyyymmdd)>").c\_str());

email.set\_time(get\_sring("Hora (hhmm)>").c\_str());

email.set\_from(get\_sring("Remitente>").c\_str());

email.set\_to(get\_sring("Destinatario>").c\_str());

email.set\_cc(get\_sring("CC>").c\_str());

email.set\_bcc(get\_sring("BCC>").c\_str());

email.set\_subject(get\_sring("Asunto>").c\_str());

email.set\_content(get\_text("Contenido>").c\_str());

}

void display(Email& email)

{

cout << "ID: " << email.get\_id() << endl;

cout << "Fecha: " << email.get\_date() << endl;

cout << "Hora: " << email.get\_time() << endl;

cout << "Remitente: " << email.get\_from() << endl;

cout << "Destinatario: " << email.get\_to() << endl;

cout << "CC: " << email.get\_cc() << endl;

cout << "BCC: " << email.get\_bcc() << endl;

cout << "Asunto: " << email.get\_subject() << endl;

cout << "Contenido: " << email.get\_content() << endl;

}

#endif

# email.hpp

#ifndef EMAIL\_HPP

#define EMAIL\_HPP

#include <ctime>

#include <cstring>

class Email

{

private:

int id;

char date[8];

char time[4];

char from[256];

char to[256];

char cc[256];

char bcc[256];

char subject[78];

char content[512];

public:

void set\_id(const int id);

int get\_id();

void set\_date(const char\* date);

char\* get\_date();

void set\_time(const char\* time);

char\* get\_time();

void set\_from(const char \*from);

char\* get\_from();

void set\_to(const char \*to);

char\* get\_to();

void set\_cc(const char \*cc);

char\* get\_cc();

void set\_bcc(const char \*bcc);

char\* get\_bcc();

void set\_subject(const char \*subject);

char\* get\_subject();

void set\_content(const char \*content);

char\* get\_content();

};

void Email::set\_id(const int id)

{

this->id = id;

}

int Email::get\_id()

{

return id;

}

void Email::set\_date(const char\* date)

{

strcpy(this->date, date);

}

char\* Email::get\_date()

{

return date;

}

void Email::set\_time(const char\* time)

{

strcpy(this->time, time);

}

char\* Email::get\_time()

{

return time;

}

void Email::set\_from(const char\* from)

{

strcpy(this->from, from);

}

char\* Email::get\_from()

{

return from;

}

void Email::set\_to(const char\* to)

{

strcpy(this->to, to);

}

char\* Email::get\_to()

{

return to;

}

void Email::set\_cc(const char\* cc)

{

strcpy(this->cc, cc);

}

char\* Email::get\_cc()

{

return cc;

}

void Email::set\_bcc(const char\* bcc)

{

strcpy(this->bcc, bcc);

}

char\* Email::get\_bcc()

{

return bcc;

}

void Email::set\_subject(const char\* subject)

{

strcpy(this->subject, subject);

}

char\* Email::get\_subject()

{

return subject;

}

void Email::set\_content(const char\* content)

{

strcpy(this->content, content);

}

char\* Email::get\_content()

{

return content;

}

#endif



