#include<iostream>

using namespace std;

class HDD {

private:

class File {

private:

double file\_size = 0;

public:

string filename;

string content;

double getFileSize()const {

return file\_size;

}

void SetFilename(const string& filename) {

int result1 = filename.find(".txt");

int result2 = filename.find(".docx");

if (result1 >= 0 && result1 <= filename.length() || result2 >= 0 && result2 <= filename.length()) {

if (filename.length() > 5) {

this->filename = filename;

}

else {

string ex = " is not correct filename : " + filename;

throw ex;

}

}

else {

string ex = filename + " is not correct , it should be .txt or .docx file";

throw ex;

}

}

File(const string& filename, const string& content)

{

SetFilename(filename);

this->content = content;

file\_size = content.length();

}

};

File\*\* files;

int count = 0;

public:

double current\_size = 0;

static double capacity;

File\* GetFile(const string& filename) {

if (count != 0) {

for (size\_t i = 0; i < count; i++)

{

if (files[i]->filename == filename) {

return files[i];

}

}

}

return nullptr;

}

void CreateFile(const string& filename, const string& content) {

auto file = GetFile(filename);

if (file != nullptr) {

string ex = filename + " : this file already exists";

throw ex;

}

double free\_space = capacity - current\_size;

double content\_size = content.length();

if (content\_size > free\_space) {

string ex = filename + " : is bigger for our disc capacity";

throw ex;

}

File\* newfile = new File(filename, content);

AddFile(\*newfile);

system("color 2");

cout << filename + " was created successfully!" << endl;

}

void AddFile(const File& file) {

auto newfiles = new File \* [count + 1]{};

for (size\_t i = 0; i < count; i++)

{

newfiles[i] = files[i];

}

newfiles[count] = new File(file);

++count;

files = newfiles;

newfiles = nullptr;

}

bool FileExists(const string& filename)

{

for (int x = 0; x < count; x++)

{

if (files[x]->filename == filename)

{

return true;

}

}

return false;

}

int GetFileIndexByName(const string& filename)

{

for (int x = 0; x < count; x++)

{

if (files[x]->filename == filename)

{

return x;

}

}

}

void DeleteFile(const string& filename)

{

if (!FileExists(filename))

{

system("color 4");

string ex = filename + " : this file does not exist";

throw ex;

}

int index = GetFileIndexByName(filename);

auto newfiles = new File \* [count - 1]{};

for (size\_t i = 0; i < index; i++)

{

newfiles[i] = files[i];

}

for (size\_t x = index + 1; x < count + 1; x++)

{

newfiles[x - 1] = files[x];

}

--count;

files = newfiles;

newfiles = nullptr;

system("color 2");

cout << filename + " was deleted successfully" << endl;

}

void UpdateFileContent(const string& filename, const string& newcontent)

{

if (!FileExists(filename))

{

system("color 4");

string ex = filename + " : this file does not exist";

throw ex;

}

int index = GetFileIndexByName(filename);

double free\_space = capacity - current\_size - files[index]->content.length();

double content\_size = newcontent.length();

if (content\_size > free\_space) {

string ex = filename + " : is bigger for our disc capacity";

throw ex;

}

files[index]->content.clear();

files[index]->content = newcontent;

cout << filename + "\'s content was changed successfully" << endl;

}

void RenameFile(const string& oldfilename, const string& newfilename)

{

if (!FileExists(oldfilename))

{

system("color 4");

string ex = oldfilename + " : this file does not exist";

throw ex;

}

int result1 = newfilename.find(".txt");

int result2 = newfilename.find(".docx");

if (result1 >= 0 && result1 <= newfilename.length() || result2 >= 0 && result2 <= newfilename.length()) {

if (newfilename.length() > 5)

{

int index = GetFileIndexByName(oldfilename);

files[index]->filename.clear();

files[index]->filename = newfilename;

system("color 2");

cout << oldfilename << " was changed to " << newfilename << endl;

}

else {

string ex = " is not correct filename : " + newfilename;

throw ex;

}

}

else {

string ex = newfilename + " is not correct , it should be .txt or .docx file";

throw ex;

}

}

};

double HDD::capacity = 2 \* 1024 \* 1024;

class Control {

public:

static void Run() {

HDD disc;

try

{

disc.CreateFile("myfile.txt", "Hello World");

disc.CreateFile("yourfile.docx", "Hello World");

// Task 1

disc.DeleteFile("myfile.txt");

//disc.DeleteFile("myfile");

// Task 2

disc.UpdateFileContent("yourfile.docx", "Salam Dunya");

// Task 3

disc.RenameFile("yourfile.docx", "newname.txt");

}

catch (const string& ex)

{

system("color 4");

cout << "Error : " << ex << endl;

}

}

};

void main()

{

Control::Run();

}