Міністерство освіти і науки України

Тернопільський національний економічний університет

Факультет комп’ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

КОМПЛЕКСНЕ ПРАКТИЧНЕ ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

з дисципліни «Операційні системи»

на тему:

« Робота з файлами в середовищі ОС Windows»

Студента І курсу КН-11 групи

напряму підготовки

«Комп’ютерні науки»

Дяківа Ростислава Ігоровича

Первірив:

к.т.н., доцент Комар М.П.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_

м. Тернопіль – 2017 рік

ЗМІСТ

Вступ………………………………………………….…………………………........3

1. Характеристика предметної області……………….…………………………….4

2. Алгоритмічне забезпечення ……………………………………………………...7

3. Програмна реалізація задачі ……………………………………………………..8

4. Результати роботи програми……………………………………………………...9

Висновки……………………………………………………………………………12

Додаток А……………………………………………………………………...……14

ВСТУП

Згідно з завданням на КПІЗ потрібно розробити програму для визначення кількості вільного місця на диску. Програма повинна забезпечити інтерфейс з користувачем через діалогове вікно. Назву диску користувач має вводити з клавіатури. В першому розділі потрібно розглянути основні принципи написання програм для операційної системи MS Windows, принципи програмування для багатозадачної операційної системи, WinAPI-функції. В другому розділі потрібно розробити алгоритмічне забезпечення для задачі визначення вільного місця на диску, зокрема побудувати блок-схему програми і здійснити його опис. В третьому розділі розробити програму і здійснити її опис. В четвертому розділі розглянути результати роботи програми. Для написання програми буде використана мова програмування Microsoft Visual С#

1 ХАРАКТЕРИСТИКА ПРЕДМЕТНОЇ ОБЛАСТІ

**Операційна система** (ОС) – програма, що здійснює діалог із користувачем, керує комп'ютером, його ресурсами, запускає інші програми. ОС завантажується при вмиканні комп'ютера.

Windows 10 – багатозадачна операційна система: дозволяє працювати з декількома програмами одночасно. Windows 10 створена для персональних комп’ютерів IBM PC. Дружній, інтуїтивний інтерфейс програми сприяє її швидкому опануванню. Працювати з Windows 10 значно приємніше і зручніше ніж з її попередниками. Для роботи в середовищі Windows необхідно на екрані вибирати з запропонованого набору потрібну операцію за допомогою миші.

Уся інформація ОС (програми, документи, таблиці, малюнки та ін.) зберігається в файлах. Файл – найменоване місце на диску для зберігання інформації. Кожний файл має**ім'я** і**розширення**. Розширення указує на тип файлу. Ім'я файлу може мати до 255 символів. Розширення відокремлюється від імені крапкою.

Наприклад: http://ua.textreferat.com/images/referats/7620/image001.jpgСтатут малого підприємства.doc

В імені файлу не можна використовувати такі символи:

**\* ? \ / | : < > "**

Перед ім’ям файлу стоїть його значок, що указує на розширення (тип) файлу:

|  |  |
| --- | --- |
| http://ua.textreferat.com/images/referats/7620/image002.jpg | – файли з розширенням com, exe, містять програми готові до виконання; |
| http://ua.textreferat.com/images/referats/7620/image003.jpg | – файли з розширенням bat – пакетні файли; |
| http://ua.textreferat.com/images/referats/7620/image004.jpg | – файли з розширенням doc – документи, створені у текстовому редакторі Microsoft Word; |
| http://ua.textreferat.com/images/referats/7620/image005.jpg | – файли з розширенням xls – таблиці, створені у табличному процесорі Microsoft Excel. |
| http://ua.textreferat.com/images/referats/7620/image006.jpg | – файли з розширенням bmp – малюнки, створені у графічному редакторі Paint. |
| http://ua.textreferat.com/images/referats/7620/image007.jpg | – файли з розширенням wbk – копії документів (Microsoft Word), зроблені перед їх модифікацією. |

Файли зберігають в папках. **Папка**(каталог) – найменоване місце на диску для зберігання файлів. Ім'я папки може мати до 255 символів. Кожна папка може містити в собі інші папки, документи, таблиці, малюнки та ін. Сукупність папок утворює деревоподібну структуру (мал.7). Якщо папка X входить до папки Y, то папка Х називається**укладеною папкою** папки Y. Значок звичайної папки зображено на мал.6. Подпись: 
Мал.1
Для відкриття будь-якої папки необхідно двічі натиснути на її значку. Після цього відкриється вікно, в якому буде подано вміст цієї папки.

У разі звертання до файлу необхідно вказати його шлях.**Шлях** – це послідовність з імен логічного диску, папок та укладених папок, розділених символом «\». Наприклад, звертання до файлу Статут.doc, що знаходиться на диску С: у папці Договори**,**яка входить доп папки Мои документы (мал.7):

С:\Мои документы\Договори\Статут.doc

**Види файлів**

Найбільш простими типами файлів є документи та програми. Документи – це об’єкти, що містять ту чи іншу інформацію: тексти, рисунки, звуки тощо. Розвиток мультимедійних можливостей комп’ютера приводить до того, що в деяких документах можуть міститися декілька видів інформації одночасно, наприклад, рухоме зображення та звук. Програми служать для обробки документів – це своєрідні інструменти дії на документи. Часто їх ще називають прикладними, наприклад прикладна програма MS-DOS або Windows. Між окремими програмами та документами існує стійкий зв’язок: текстовий редактор працює з текстовими документами, програма-фонограф відтворює звуки і т.д. Windows "запам’ятовує" такі зв’язки і здатна самостійно їх використовувати при перегляді та роботі з документами.

Група однотипних документів, а також програми для їх обробки можуть бути розміщені в одній спільній папці. Папка є ще одним об’єктом у Windows. На відміну від документів та програм, які є простими та "неподільними" об’єктами, папка може містити інші об’єкти у тому числі і нові папки, зокрема папка може бути порожньою.

Незалежно від операційних систем персональних комп’ютерів всі файли можна поділити на текстові та бінарні (по іншому – двійкові) файли. Текстовими називають файли, у яких в якості інформаційних використовуються символи з десятковими кодами 32–126 та 128–254. Двійкові файли представляють собою послідовність із будь-яких символів. Їх довжина визначається із заголовку файла. Цей поділ є важливим для різних операційних систем, оскільки призначення та обробка бінарних та текстових файлів в операційних системах відрізняються.

Також файли можна поділити на виконавчі (програми) та невиконавчі (файли даних та документів). Виконавчі файли можуть запускатися операційною системою на виконання, а невиконавчі файли можуть змінити свій зміст тільки в процесі виконання програм. Далі файли можна поділити на основні, присутність яких обов’язкова для роботи операційної системи та програмних продуктів; службові, у яких зберігається конфігурація та настройки основних програмних файлів; робочі, зміст яких залежить від результатів роботи основних програмних файлів і, власне, за допомогою яких і створюються всі інші файли; а також тимчасові файли, що створюються у момент роботи основних, і у яких зберігають проміжні результати

2 АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ

Алгоритм роботи програми визначається специфікою виконання будь-якої віконної програми в середовищі ОС Windows. Після завантаження в пам’ять програма створює консольне вікно програми. Далі відбувається створення об’єктів Файла та ФайлМенеджера. Наступний крок — виклик методів Ф ФайлМенеджера, а саме вивід інформації про файл, його копіювання у ту саму папку з іншим іменем та видалення з теперішньої папки. Потім створюється новий обєкт Файлу, та викликається метод його переміщення у іншу папку. Потім у другій фазі моєї програми викликатюся статичні методи класу ФайлМенеджер, а саме RunStream(запуск потоку файлу і виведення його вмісту на консоль), RunFile(відкриття файлу та виведення його вміту на консоль), RunFileStream(видалення файлу,якщо він існує, створення файлу, його наповнення випадковими символами, виведення його вмісту на консоль).

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАДАЧІ

Програма складається з 3-ох основних частин Program.cs – вхід в програму, класи

Для роботи з файловою системо використовуються бібліотека System.IO

FileInstance та FileManager.

FileInstance має поля FilePath i FileInfo.

FileManager у своєму тілі має функції для роботи з файлами: InformationAboutFile() – реалізовує зчитування та вивід інформації про файл якщо він існує

DeleteFile() – якщо файл існує видаляє вказаний файл та виводить повідомлення у консоль про успішне видалення файлу

TransferFile() - якщо файл існує переміщає вказаний файл по заданому шляху та виводить повідомлення у консоль про успішне переміщення файлу

CopyFile() - якщо файл існує копіює вказаний файл по заданому шляху та виводить повідомлення у консоль про копіювання переміщення файлу

RunStream() – за допомогою StreamReader зчитує повністю вміст файлу у стрічку та виводить її у консоль, якщо виникає помилка то виводить повідомлення про помилку у консоль

RunFile() – за допомогою StreamReader зчитує по вміст файлу по лінійці у масив стрічкок та виводить їх по черзі у консоль, якщо виникає помилка то виводить повідомлення про помилку у консоль.

RunFileStream() – якщо вказаний файл існує, то видаляє його. Потім за допомогою FileStream створюю новий файл за шляхом вказаного файлу та додає у нього текст за допомогою статичного методу AddText. І в кінці за допомого FileStream відкриваю щойно згенерований файл і зчитую його вміст у консоль.

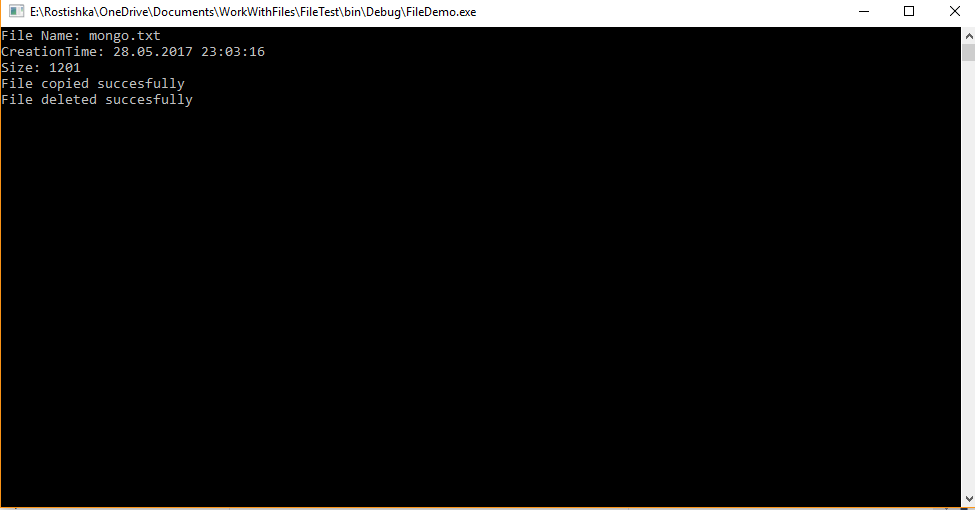
У файлі Program.cs створюються об’єкти типів FileManager та FileInstance і викликаться методи об’єкту FileManager.

РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ

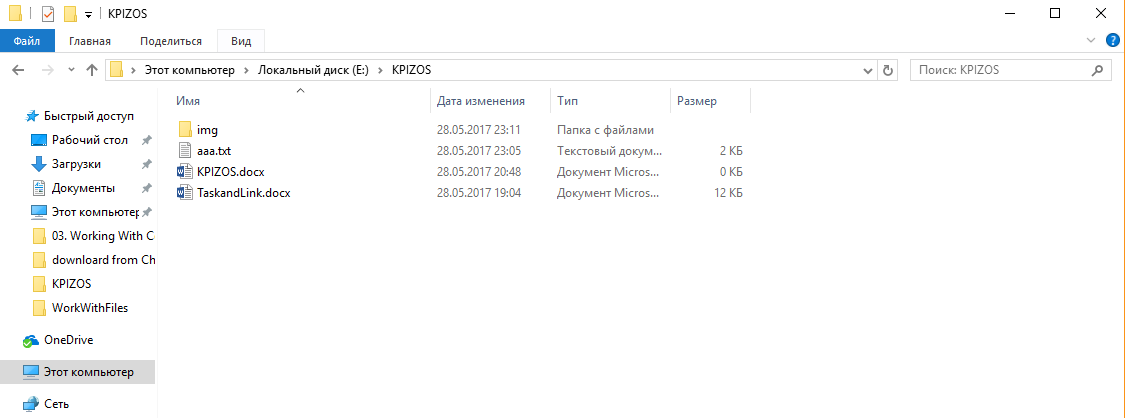
‘ Вміст початкової папки і файлу перед запуском програми:



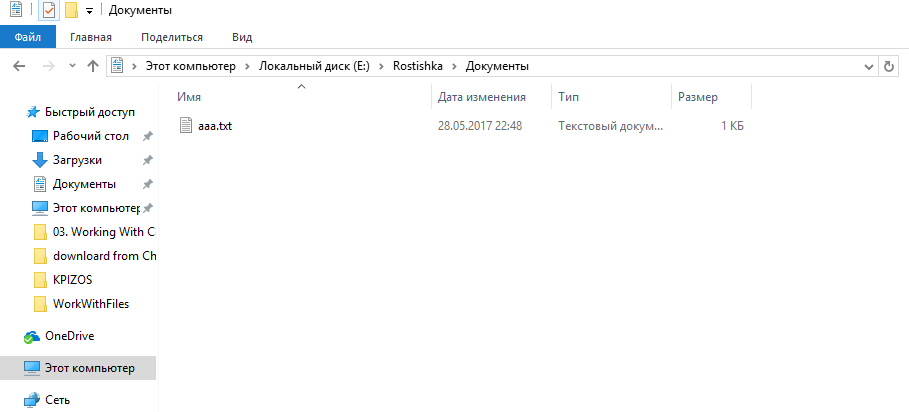
 В результаті запуску першої фази моєї програми з’являється наступне консольне вікно:



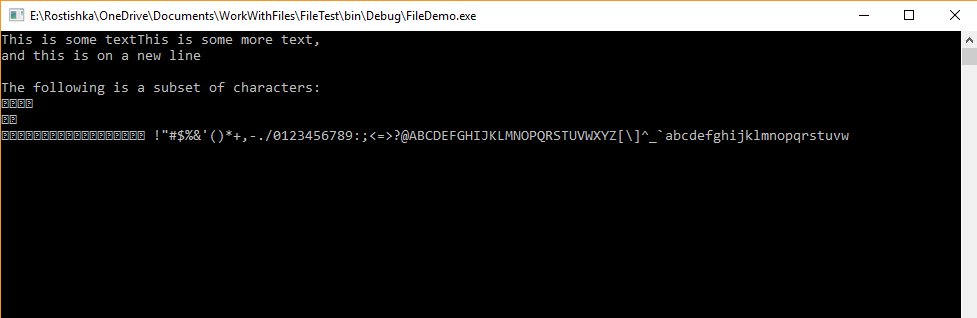
Вміст початкової папки і файлу після запуску програми:



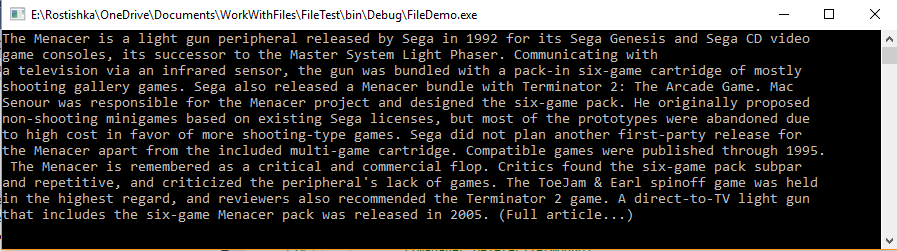
Вміст іншої папки піcля запуску програми:



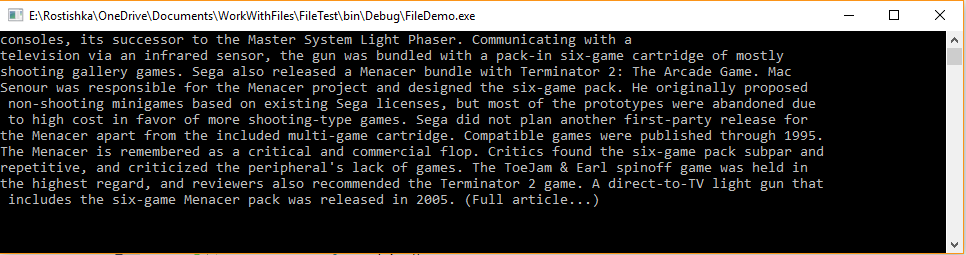
Консольне вікно після запуску методу RunFileStream() другої фази програми



Консольне вікно після запуску методу RunFile() другої фази програми



Консольне вікно після запуску методу RunStream() другої фази програми



ВИСНОВОК

У своїй роботі я висвітлив тему видів файлів та роботи з ними в ОС Windows. Розповів про алгоритми роботи з файлами. Навчився застосовувати знання на практиці. Крім того, ознайомимося з основними принципами об’єктно-орієнтованого програмування. Результатом написання даного КПІЗ є розробка програми для роботи з файлами в середовищі ОС Windows.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ресурсы Microsoft Windows NT Workstation 4.0. / Пер. с. Англ. – СПб.: Питер, 2000. – 384 с.

2. Системное программное обеспечение / А.В. Гордеев, А.Ю. Молчанов. – СПб.: Питер, 2001. – 736 с.

3. Джордейн Р. Справочник программиста персональных компьютеров. — М.: Финансы и статистика, 1995. - 592 с.

4. Скэнлон Л. Персональные ЭВМ IBM PC и XT. Программирование на языке ассемблера: Пер. с англ. - М.: Радио и связь, 1989. – 336 с.

5. Пратт Т., Зелковиц М. Языки программирования: реализация и разработка. – СПб.: Питер, 2001 6. Соловьев Г.Н., Никитин В.Д. Операционные системы ЭВМ. - М.: Высшая школа, 1989. - 255 с.

Додаток А

Текст програми „ Робота з файлами в середовищі ОС Windows ”

**Файл Program.cs**

using System;

namespace FileDemo

{

class Program

{

static void Main(string[] args)

{

FileInstance mongo = new FileInstance(@"E:\KPIZOS\mongo.txt");

FileManager manager = new FileManager();

manager.InformationAboutFile(mongo);

manager.CopyFile(mongo, @"E:\KPIZOS\aaa.txt");

manager.DeleteFile(mongo);

FileInstance aaa = new FileInstance(@"E:\KPIZOS\aaa.txt");

manager.TransferFile(aaa, @"E:\Rostishka\Documents\aaa.txt");

|  |
| --- |
| FileManager.RunStream(aaa); |
|  | FileManager.RunFile(aaa); |
|  | FileManager.RunFileStream(aaa); |

}

}

}

**Файл FileManager.cs**

using System;

using System.IO;

using System.Text;

namespace FileDemo

{

public class FileManager

{

public FileManager(){}

public void InformationAboutFile(FileInstance fileInstance)

{

if (fileInstance.FileInfo.Exists)

{

Console.WriteLine("File Name: {0}", fileInstance.FileInfo.Name);

Console.WriteLine("CreationTime: {0}", fileInstance.FileInfo.CreationTime);

Console.WriteLine("Size: {0}", fileInstance.FileInfo.Length);

}

}

public void DeleteFile(FileInstance fileInstance)

{

if (fileInstance.FileInfo.Exists)

{

fileInstance.FileInfo.Delete();

Console.WriteLine("File deleted succesfully");

}

}

public void TransferFile(FileInstance fileInstance, string newFilePath)

{

if (fileInstance.FileInfo.Exists)

{

fileInstance.FileInfo.MoveTo(newFilePath);

Console.WriteLine("File transfered succesfully");

}

}

public void CopyFile(FileInstance fileInstance, string newFilePath)

{

if (fileInstance.FileInfo.Exists)

{

fileInstance.FileInfo.CopyTo(newFilePath, true);

Console.WriteLine("File copied succesfully");

}

}

public static void RunStream(FileInstance fileInstance)

{

try

{ // Open the text file using a stream reader.

using (StreamReader sr = new StreamReader(fileInstance.FilePath, System.Text.Encoding.Default))

{

// Read the stream to a string, and write the string to the console.

String line = sr.ReadToEnd();

Console.WriteLine(line);

}

}

catch (Exception e)

{

Console.WriteLine("The file could not be read:");

Console.WriteLine(e.Message);

}

}

public static void RunFile(FileInstance fileInstance)

{

try

{

string[] sr = System.IO.File.ReadAllLines(fileInstance.FilePath, System.Text.Encoding.Default);

foreach (string line in sr)

{

Console.WriteLine(line);

}

}

catch (Exception e)

{

Console.WriteLine("The file could not be read:");

Console.WriteLine(e.Message);

}

}

public static void RunFileStream(FileInstance fileInstance)

{

if (File.Exists(fileInstance.FilePath))

{

File.Delete(fileInstance.FilePath);

}

//Create the file.

using (FileStream fs = File.Create(fileInstance.FilePath))

{

AddText(fs, "This is some text");

AddText(fs, "This is some more text,");

AddText(fs, "\r\nand this is on a new line");

AddText(fs, "\r\n\r\nThe following is a subset of characters:\r\n");

for (int i = 1; i < 120; i++)

{

AddText(fs, Convert.ToChar(i).ToString());

}

}

//Open the stream and read it back.

using (FileStream fs = File.OpenRead(fileInstance.FilePath))

{

byte[] b = new byte[1024];

UTF8Encoding temp = new UTF8Encoding(true);

while ((fs.Read(b, 0, b.Length) > 0))

{

Console.WriteLine(temp.GetString(b));

}

}

}

private static void AddText(FileStream fs, string value)

{

byte[] info = new UTF8Encoding(true).GetBytes(value);

fs.Write(info, 0, info.Length);

}

}

}

**Файл FileInstance.cs**

using System;

using System.IO;

namespace FileDemo

{

public class FileInstance

{

private String \_filePath;

private FileInfo \_fileInfo;

public String FilePath

{

get => \_filePath;

set => \_filePath = value;

}

public FileInfo FileInfo

{

get => \_fileInfo;

set => \_fileInfo = value;

}

public FileInstance(String filePath)

{

FilePath = filePath;

\_fileInfo = new FileInfo(\_filePath);

}

}

}