|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |
| **Институт комплексной безопасности и специального приборостроения** | | |
|  |  |  |

Кафедра КБ-4 «Управление и моделирование систем»

**Отчет**

По проекту:  
Система оптимизации данных SmartFileLinker

Выполнил:

Студент 3 курса

Группа ББСО-02-19

Шифр 19Б1192

Мысин Никита Олегович

Москва, 2021

Мой проект называется “Система оптимизации данных SmartFileLinker”. В нём я реализую оптимизацию данных в указанной папке.

Опишу по пунктам принцип действия программы и её логику:

1. Меню команд

С помощью меню команд можно выбирать, что будет делать программа.

elif command=="commands":

print("Список команд:")

print("exit - выход из программы")

print("stop - выход из программы")

print("commands - список всех команд")

print("info - информация о программе")

print("instruction - инструкция к программе")

print("eval - выполнение однострочной команды интерпретатором Python")

print("system - выполнение однострочной команды системой")

print("getram - вывести на экран размер буфера")

print("setram - установить размер буфера")

print("encode - оптимизировать папку")

print("decode - деоптимизировать папку")

print("iscopy - определение схожести двух файлов")

print("size - определение размера файла или папки")

print("exists - проверка файла или папки на существование")

print("isdir - проверка принадлежности указанного пути к папке")

print("isfile - проверка принадлежности указанного пути к файлу")

print("getabsolutepath - получение абсолютного пути к папке или файлу")

print("createfile - создать новый файл")

print("createdir - создать новую папку")

print("clear - очистить файл")

print("rewrite - перезаписать файл")

print("write - дозаписать данные в файл")

print("append - дозаписать данные в файл")

print("delete - удалить файл или папку")

print("read - прочитать файл")

print("rename - переименовать файл или папку")

print("copy - скопировать файл или папку")

print("move - переместить файл или папку")

print("! - игнорирование следующего сообщения")

2. Проверка на копию

В алгоритме оптимизации информации используется алгоритм проверки схожести файлов.

def isCopy(path1, path2): #побайтово сравниваем содержимое двух файлов

if file.exists(path1) and file.exists(path2):

result = False

if file.size(path1)==file.size(path2):

result = True

file1 = open(path1, "rb") #rb - read byte или read binary

file2 = open(path2, "rb")

temp = file.size(path1)

while temp>0:

if temp<file.memory: #memory – максимальный размер буфера

data1 = file1.read(temp)

data2 = file2.read(temp)

temp-=temp

else:

data1 = file1.read(file.memory)

data2 = file2.read(file.memory)

temp-=file.memory

x=0

while x < len(data1) and data1[x] == data2[x]:

x+=1

if x!=len(data1):

return False

return result

3. Оптимизация

Оптимизация происходит следующим образом: в указанной папке создаётся файл-таблица. Далее в неё заносится путь основного файла и пути к его копиям. Копии все удаляются.

Например у нас есть три видеофайла video1.mp4, video2.mp4 и video3.mp4. Все они являются полностью одинаковыми по содержанию. В файл-таблицу под названием по умолчанию «list.list» данные занесутся следующим образом:

video1.mp4:

video2.mp4

video3.mp4

То есть, получается, что в таблицу заносится путь к ориганалу, далее следует двоеточие и в следующих строках перечисление путей к копиям.

Одиночные файлы — файлы без копий — в таблицу вноситься не будут.

Возьмём другой пример, с внутренними папками. У нас есть три полностью одинаковых по содержанию видеофайла: video1.mp4, papka1/video2.mp, papka2/video3.mp4, столько же одинаковых аудиофайлов: papka1/audio1.mp3, papka2/audio2.mp3, audio3.mp3 и один текстовый файл text.txt. Результат будет следующим:

audio3.mp3:

papka1/audio1.mp3

papka2/audio2.mp3

video1.mp4:

papka1/video2.mp4

papka2/video3.mp4

def encode(path):

for subdir1, dirs1, files1 in os.walk(path):

if not (file.exists(path + "\\list.list")):

file.createFile(path + "\\list.list")

l = open(path+"\\list.list", "a")

for f1 in files1:

swing = True

for subdir2, dirs2, files2 in os.walk(path):

for f2 in files2:

if not (subdir1 + "\\" + f1==subdir2 + "\\" + f2):

if (file.isCopy(subdir1 + "\\" + f1, subdir2 + "\\" + f2)) and subdir1+"\\"+f1!=path+"\\"+"list.list" and subdir2+"\\"+f2!=path+"\\"+"list.list":

if (swing):

l.write(subdir1 + "\\" + f1 + ":\n")

swing = False

l.write(subdir2 + "\\" + f2 + "\n")

file.delete(subdir2 + "\\" + f2)

print("Оптимизация " + subdir2 + "\\" + f2+" ")

l.close()

4. Деоптимизация

Деоптимизация происходит по прямо противоположному к алгоритму оптимизации алгоритму. Оригинал файла копируется под именами копий по их пути, после чего файл-таблица list.list удаляется.

def decode(path):

src = ""

if not (file.exists(path + "\\list.list")):

print("Error")

else:

with open(path+"\\list.list", "r") as l:

for line in l:

if line.endswith(":\n"):

src = line[:-2]

else:

print ("Восстановление " + line[:-1])

file.createFile(line[:-1])

shutil.copy2(src, line[:-1])

file.delete(path+"\\list.list")

5. Интерфейс

Для взаимодействия пользователя программы используется интерфейс, часть которого выглядит так:

def interface(): #здесь находится весь интерфейс

print("Добро пожаловать в программу оптимизации данных SmartFileLinker!")

print("Посмотреть список команд можно командой \"commands\"")

run = True

command = ""

while run:

command = input("\n>>> ")

print()

if command=="exit" or command=="stop":

run=False

print("Выход из программы SmartFileLinker...\n")

time.sleep(5)

elif command=="info":

print("Информация:")

print(" Версия: 1.0")

print(" Почта: n.mysin@list.ru")

print(" Задача:")

print(" Основной задачей данного проекта является оптимизация дискового пространства, методом замены копий файлов в указанной папке на их пути с учётом наименования в таблице ссылок. Файл таблицы ссылок создаётся с наименованием list.list в указанной для оптимизации папке. При восстановлении всех файлов файл таблицы ссылок автоматически удаляется.")

elif command=="instruction":

print("Инструкция:")

print(" После запуска программы пользователь, при необходимости, читает инструкцию и список команд, после чего работает в программе, вводя необходимые команды. Удалять файлы вручную из оптимизированной папки крайне не рекомендуется, так как существует вероятность удаления оригиналов файлов, после чего ссылки других оптимизированных файлов в этой папке перестанут быть действительными. При необходимости добавить, удалить, переименовать или перезаписать любой файл в оптимизируемой папке, рекомендуется деоптимизировать папку, произвести необходимые операции и снова оптимизировать папку.")

elif command=="eval":

eval(input("eval>>> "))

elif command=="system":

print(os.system(input("system>>> ")))

elif command=="setram":

file.memory = int(input("Введите размер буфера в байтах>>> "))

elif command=="getram":

print("Размер буфера в байтах "+str(file.memory))

elif command=="commands":

Все основные принципы работы программы оптимизации данных SmartFileLinker были перечислены! Основные части программы не являются основными, но без них программа не была бы полной. Программа была написана на языке программирования Python версии 3.9 для 64-разрядной системы.