# Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 2 з дисципліни

«Основи програмування-2.

Базові конструкції»

«Бінарні Файли»

Варіант 20

Виконав студент Лошак Віктор Іванович

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна

( прізвище, ім'я, по батькові)

Київ 2022

**Лабораторна робота 2**

**Мета:**

Вивчити особливості створення і обробки бінарних файлів даних.

**Тема:**

Створити файл із списком телепередач: назва час початку, час закінчення передачі. Визначити тривалість кожної передачі. Створити новий файл , що містить передачі які відображаються в денний час з 9:00 до 18:00.

**Математична модель:**

Для знаходження вмісту результуючого файлу використаємо підпрограми що реалізуватимуть відповідно: читання вмісту вже записаного файлу , зчитування користувацького багаторядкового вводу з наступним записом у файл. Для реалізації заданих функцій використаємо стандартні доступні бібліотеки та модулі. Для реалізації фільтрації передач за часом їх показу використаємо проходження по об’єктах файлу. Для читання і запису в файл використаємо окремі підпрограми що реалізуватимуть блоковий запис в бінарний файл.

**2)Приклад коду на Рython:**

**Lab2\_py\_:**

import pickle

from module import \*

from ProgramTV import \*

def main():

filePath1="firstFile"

filePath2="secondFile"

WriteToFile(filePath1, GetInputPrograms())

fileContent:list[ProgramTV]=GetFilePrograms(filePath1)

DisplayFilePrograms(fileContent)

chosenContent:list[ProgramTV]=[item for item in fileContent if item.startTime.hour>=9 and item.endTime.hour<=18 ]

WriteToFile(filePath2, chosenContent)

DisplayFilePrograms(GetFilePrograms(filePath2))

if \_\_name\_\_=="\_\_main\_\_":

main()

**module:**

import datetime

import keyboard

import pickle

from ProgramTV import \*

import msvcrt

import sys

def DisplayFilePrograms(programTVs: list[ProgramTV]):

for program in programTVs:

program.Print()

def GetFilePrograms(filePath: str):

fileContent=list()

with open(filePath, "rb") as reader:

print(f"Reading content of file \"{filePath}\".")

fileContent=pickle.load(reader)

return fileContent

def WriteToFile(file: str, someList: list[ProgramTV] , mode: str="wb"):

with open(file, mode) as writer:

pickle.dump(someList, writer)

def GetInputPrograms():

myProgrames=list()

while True:

answ=input("Do you want to add another program?(y/n)")

if(answ=="y"):

myProgrames.append(GetProgramTVFromConsole())

else:

break

print()

return myProgrames

def GetProgramTVFromConsole():

inp = ProgramTV()

inp.title = input("Enter program name: ")

inp.startTime = datetime.datetime.strptime(input("Enter program start time: "), "%H:%M")

inp.endTime = datetime.datetime.strptime(input("Enter program end time: ") , "%H:%M")

keyboard.clear\_all\_hotkeys

sys.stdout.flush()

while msvcrt.kbhit():

msvcrt.getch()

inp.timeSpan = inp.endTime-inp.startTime

return inp

**ProgramTV:**

import datetime

class ProgramTV(object):

def \_\_init\_\_(self):

self.title=''

self.startTime=datetime.datetime.now()

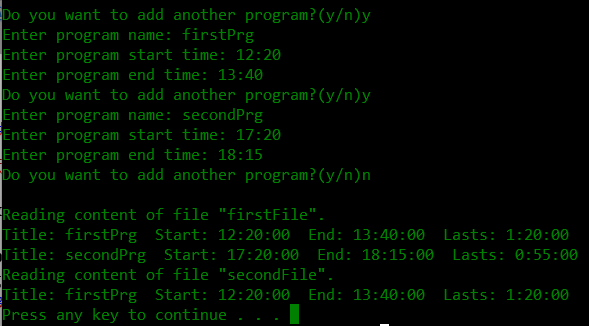
self.endTime=datetime.datetime.now()

self.timeSpan=self.endTime-self.startTime

def Print(self):

print(f"Title: {self.title} Start: {self.startTime.time()} End: {self.endTime.time()} Lasts: {self.timeSpan}")

**Виконання коду на python:**

****

**3)Приклад коду на C#**

**Program.cs**

using System;

using System.Collections.Generic;

using System.IO;

using System.Threading;

using static Lab2.ProgramMethods;

namespace Lab2

{

class Program

{

static void Main(string[] args)

{

string filePath1 = "firstFile";

string filePath2 = "secondFile";

//Getting user input and writing it into the file

WriteIEnumerableToFile<ProgramTV>(filePath1, GetInputPrograms());

//displaying the content of the first file

List<ProgramTV> fileContent = GetFilePrograms(filePath1);

displayFilePrograms(fileContent);

//selecting the programs that satisfy given condition

List<ProgramTV> condSatisf=fileContent.FindAll(x => x.StartTime.Hour >= 9 && x.EndTime.Hour <= 18 );

//writing selected programs to the second file

WriteIEnumerableToFile<ProgramTV>(filePath2, condSatisf);

//displaying the selected programs

displayFilePrograms(GetFilePrograms(filePath2));

}

}

}

**ProgramTV.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

class ProgramTV

{

public DateTime StartTime { get; set; }

public DateTime EndTime { get; set; }

public TimeSpan TimeSpan { get; set; }

public string Title { get; set; }

public ProgramTV(DateTime start, DateTime end, string title)

{

StartTime = start;

EndTime = end;

Title = title;

TimeSpan = new TimeSpan(EndTime.Hour-StartTime.Hour, EndTime.Minute-StartTime.Minute, EndTime.Second - StartTime.Second);

}

public ProgramTV() { }

public void Print()

{

Console.WriteLine($"Title: {Title} Start: {StartTime.TimeOfDay} End: {EndTime.TimeOfDay} Lasts: {TimeSpan}");

}

}

}

**ProgramMethods.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Lab2

{

static class ProgramMethods

{

//method for displaying lists of programs

public static void displayFilePrograms(List<ProgramTV> programTVs)

{

foreach (var program in programTVs)

{

program.Print();

}

}

//method for getting all the programs from a file

public static List<ProgramTV> GetFilePrograms(string filePath)

{

List<ProgramTV> fileContent = new List<ProgramTV>();

BinaryReader reader = new BinaryReader(File.Open(filePath, FileMode.Open));

Console.WriteLine($"Reading content of file \"{filePath}\". ");

while (reader.PeekChar() != -1)

{

fileContent.Add(GetProgramTVFromFile(reader));

}

return fileContent;

}

//method for reading single program from file

public static ProgramTV GetProgramTVFromFile(BinaryReader reader)

{

ProgramTV fileContent = new ProgramTV();

fileContent.Title = reader.ReadString();

fileContent.StartTime = DateTime.Parse(reader.ReadString());

fileContent.EndTime = DateTime.Parse(reader.ReadString());

fileContent.TimeSpan = new TimeSpan(fileContent.EndTime.Hour - fileContent.StartTime.Hour, fileContent.EndTime.Minute - fileContent.StartTime.Minute, fileContent.EndTime.Second - fileContent.StartTime.Second);

return fileContent;

}

//method for writing a list of some objects to file

public static void WriteIEnumerableToFile<T>(string filePath, IEnumerable<T> someObject, FileMode mode = FileMode.Create)

{

BinaryWriter writer = new BinaryWriter(File.Open(filePath, mode));

foreach (var part in someObject)

{

//this switch construction is not obligatory in this case, i just wrote it for training

//we can make this class non generic easily its just more fun this way

switch (part)

{

case ProgramTV pr:

writer.Write(pr.Title);

writer.Write(pr.StartTime.ToString());

writer.Write(pr.EndTime.ToString());

break;

default:

Console.WriteLine($"The way of serialization wasn`t specified for given object." +

$" Error occured in {typeof(Program).GetMethod("WriteIEnumerableToFile").Name}");

break;

}

}

writer.Flush();

writer.Close();

}

//returns all programs from input as a List? tracks if user uses stop button

public static List<ProgramTV> GetInputPrograms()

{

List<ProgramTV> myProgrames = new List<ProgramTV>();

ConsoleKeyInfo key;

bool continueInput;

do

{

myProgrames.Add(GetProgramTVFromConsole());

Console.WriteLine("To stop prress ctrl+e now, to continue press Enter");

key = Console.ReadKey(true);

continueInput = key.Key != ConsoleKey.E && key.Modifiers != ConsoleModifiers.Control;

} while (continueInput);

Console.Write('\n');

return myProgrames;

}

//processes one program input and returns it as an object of ProgramTV

public static ProgramTV GetProgramTVFromConsole()

{

ProgramTV input = new ProgramTV();

DateTime temp;

Console.Write("Enter program name: ");

input.Title = Console.ReadLine();

Console.Write("Enter program start time: ");

DateTime.TryParseExact(Console.ReadLine(), "H:m", null, System.Globalization.DateTimeStyles.None, out temp);

input.StartTime = temp;

Console.Write("Enter program end time: ");

DateTime.TryParseExact(Console.ReadLine(), "H:m", null, System.Globalization.DateTimeStyles.None, out temp);

input.EndTime = temp;

input.TimeSpan = new TimeSpan(input.EndTime.Hour - input.StartTime.Hour, input.EndTime.Minute - input.StartTime.Minute, input.EndTime.Second - input.StartTime.Second);

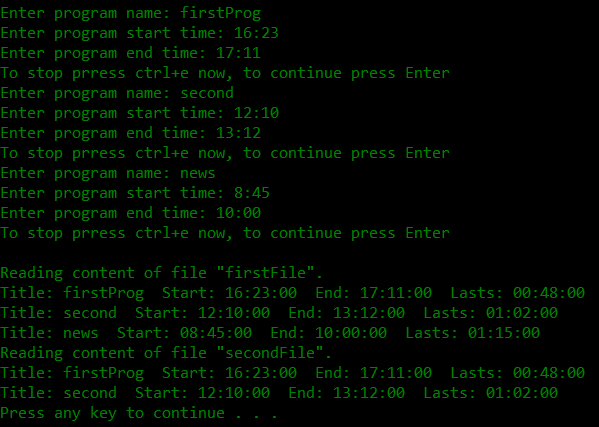
return input;

}

}

}

**Виконання коду на C#:**



**Висновок:**

Отже за допомогою даного алгоритму ми успішно створили два бінарних файли програмним шляхом та записали в нього довільні ввідні дані а також, навчилися читати та обробляти вміст бінарних файлів та вивчили роботу файлових потоків, їх типів та модифікаторів , за допомогою використання підпрограм на компільованій мові С# та python .