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 № 1.2 з дисципліни

«Основи програмування - 2.

Модульне програмування.»

“Бінарні файли”

Варіант: 30

Виконав студент ІП-11 Тихонов Федір Сергійович

(шифр, прізвище, ім'я, по батькові)

Перевірив

(прізвище, ім'я, по батькові)

**Лабораторна робота № 1.2**

**“Бінарні файли”**

**Мета -** Визначити особливості створення і обробки бінарних файлів.

**Постановка задачі -** Створити файл із розкладом занять у навчальному закладі протягом дня: назва предмета, час початку пари (ГГ:ХХ), час закінчення пари. При введені даних виконувати перевірку накладання кожної пари. Перерва має бути не більше 45 хв та не менше 5 хв.

**Код на С++ -**

**header**

**#pragma once**

**#include <iostream>**

**#include <string>**

**#include <fstream>**

**#include <iomanip>**

**#include <vector>**

**using namespace std;**

**struct timeFormat{**

**int hours, minutes;**

**};**

**struct lessonInfo{**

**char subject[100];**

**timeFormat start, finish;**

**};**

**lessonInfo inputLesson();**

**void outputLesson(lessonInfo lesson);**

**void inputLessons(const string& fileName);**

**void displayTimetable(const string& fileName);**

**void askToDelete(const string& fileName);**

**void insertSorted(lessonInfo lesson, vector<lessonInfo>& lessonList);**

**vector <lessonInfo> getVector(string fileName);**

**void CheckIntervals(vector<lessonInfo> lessonList);**

**source**

**#include "library.h"**

**using namespace std;**

**lessonInfo inputLesson() {**

**lessonInfo lesson{};**

**setlocale(LC\_ALL, "");**

**cout << "Введіть назву предмета" << endl;**

**cin >> lesson.subject;**

**cout << "Введіть початок пари в форматі ГГ:ХХ" << endl;**

**scanf("%d:%d", &lesson.start.hours, &lesson.start.minutes);**

**lesson.finish.hours = (lesson.start.hours\*60 + lesson.start.minutes + 105) / 60;**

**lesson.finish.minutes = (lesson.start.hours\*60 + lesson.start.minutes + 105) % 60;**

**return lesson;**

**}**

**void outputLesson(lessonInfo lesson){**

**cout << "Дисципліна - " << lesson.subject << endl;**

**printf("Час проведення: %d:%02d - %d:%02d\n", lesson.start.hours, lesson.start.minutes, lesson.finish.hours, lesson.finish.minutes);**

**}**

**void inputLessons(const string& fileName){**

**setlocale(LC\_ALL, "");**

**vector <lessonInfo> lessons = getVector(fileName);**

**string move\_on;**

**while(move\_on != "т"){**

**lessonInfo lesson = inputLesson();**

**insertSorted(lesson, lessons);**

**CheckIntervals(lessons);**

**cout << "Розклад заповнений? Введіть 'т', якщо розклад заповнений." << endl;**

**cin >> move\_on;**

**}**

**ofstream file;**

**file.open(fileName, ios::out | ios::binary);**

**for(lessonInfo lesson : lessons){**

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

**}**

**file.close();**

**}**

**void displayTimetable(const string& fileName){**

**ifstream file;**

**file.open(fileName, ios::in | ios::binary);**

**lessonInfo lesson{};**

**cout << "Читаємо розклад з бінарного файлу " << fileName << endl;**

**while(file.read((char\*)&lesson, sizeof(lesson))){**

**cout << endl;**

**outputLesson(lesson);**

**}**

**file.close();**

**}**

**void askToDelete(const string& fileName){**

**cout << "Хочете зберегти файл?" << endl;**

**cout << "Напишіть 'т', якщо хочете." << endl;**

**string answer;**

**cin.ignore();**

**getline(cin, answer);**

**if(answer != "т"){**

**ofstream file(fileName, ios::binary | ios::out);**

**file.close();**

**}**

**}**

**void insertSorted(lessonInfo lesson, vector<lessonInfo>& lessonList){**

**bool inserted = false;**

**for(int i = 0; i < lessonList.size() && !inserted; i++){**

**if(lesson.start.hours \* 60 + lesson.start.minutes -**

**lessonList[i].finish.hours \* 60 - lessonList[i].finish.minutes < 0){**

**lessonList.insert(lessonList.begin() + i, lesson);**

**inserted = true;**

**}**

**}**

**if(!inserted){**

**lessonList.push\_back(lesson);**

**}**

**}**

**vector <lessonInfo> getVector(string fileName){**

**ifstream file;**

**vector <lessonInfo> lessonVector = {};**

**lessonInfo lesson{};**

**file.open(fileName, ios::in | ios::binary);**

**while (file.read((char\*)&lesson, sizeof(lesson))){**

**lessonVector.push\_back(lesson);**

**}**

**file.close();**

**return lessonVector;**

**}**

**void CheckIntervals(vector<lessonInfo> lessonList){**

**for(int i = 0; i < lessonList.size() - 1; i++){**

**int difference = lessonList[i+1].start.hours\* 60 + lessonList[i+1].start.minutes -**

**lessonList[i].finish.hours \* 60 - lessonList[i].finish.minutes;**

**if(difference > 45){**

**cout << "Дуже велика перерва між парами:" << endl;**

**outputLesson(lessonList[i]);**

**outputLesson(lessonList[i+1]);**

**}**

**else if(difference > 0 && difference < 5){**

**cout << "Дуже мала перерва між парами:" << endl;**

**outputLesson(lessonList[i]);**

**outputLesson(lessonList[i+1]);**

**}**

**else if(difference < 0) {**

**cout << "Пари накладаються одна на іншу:" << endl;**

**outputLesson(lessonList[i]);**

**outputLesson(lessonList[i + 1]);**

**}**

**}**

**}**

**main**

**#include "library.h"**

**using namespace std;**

**int main() {**

**setlocale(LC\_ALL, "");**

**string fileName = "lessons\_data.dat";**

**inputLessons(fileName);**

**displayTimetable(fileName);**

**askToDelete(fileName);**

**return 0;**

**}**

**Код на Python -**

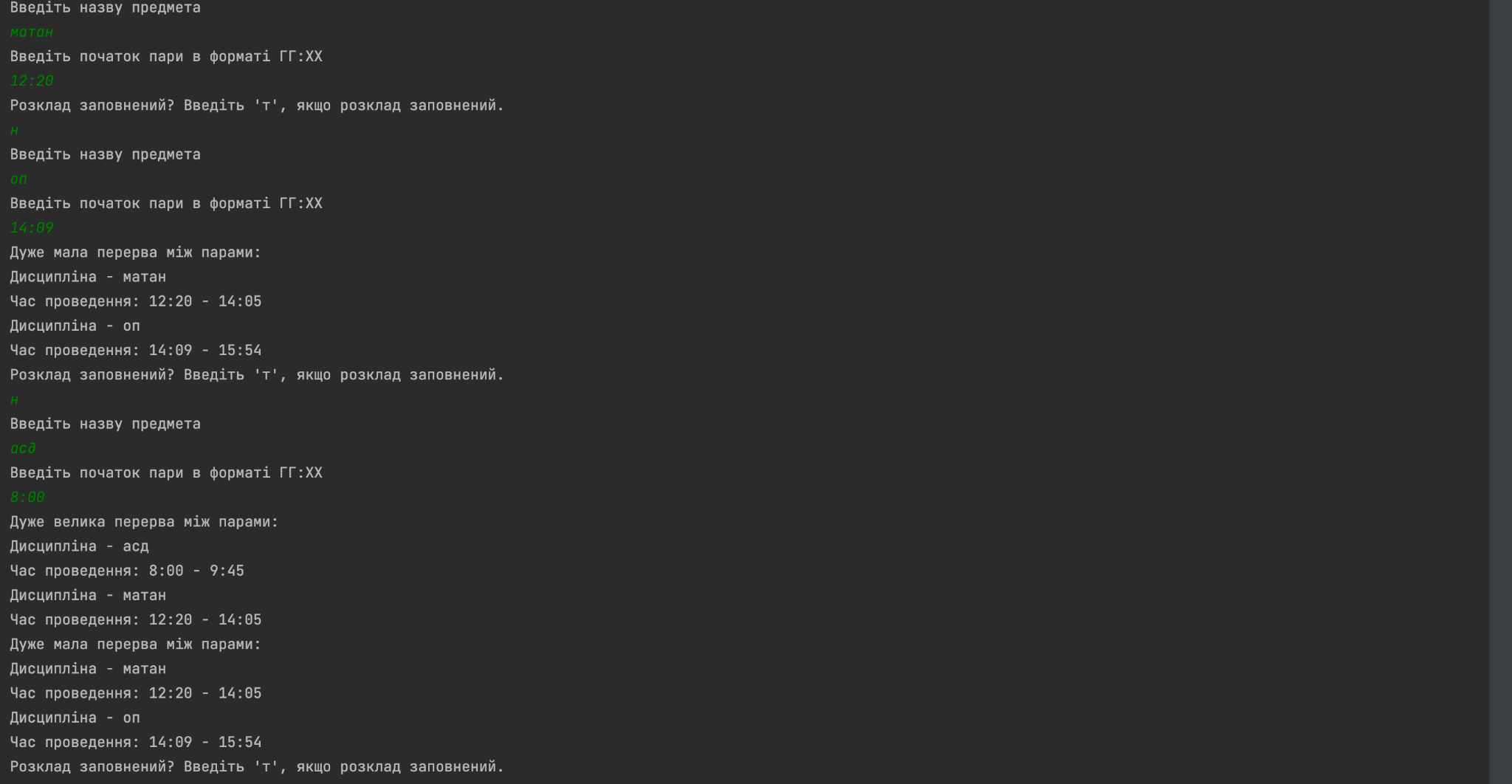
**module**

| **import \_pickle** |  |
| --- | --- |
|  |  |
|  |  |
|  | **class timeFormat:** |
|  | **def \_\_init\_\_(self, hours, minutes):** |
|  | **self.hours = hours** |
|  | **self.minutes = minutes** |
|  |  |
|  |  |
|  | **class lessonFormat:** |
|  | **def \_\_init\_\_(self, startHours, startMinutes, finishHours, finishMinutes, lessonName):** |
|  | **self.start = timeFormat(startHours, startMinutes)** |
|  | **self.finish = timeFormat(finishHours, finishMinutes)** |
|  | **self.lessonName = lessonName** |
|  |  |
|  |  |
|  | **def inputLesson():** |
|  | **lessonName = input("Введіть назву предмета:\n")** |
|  | **startHours, startMinutes = input("Введіть час початку пари у форматі ГГ:ХХ\n").split(":")** |
|  | **startHours, startMinutes = int(startHours), int(startMinutes)** |
|  | **finishHours = (startHours \* 60 + startMinutes + 105) // 60** |
|  | **finishMinutes = (startHours \* 60 + startMinutes + 105) % 60** |
|  | **newLesson = lessonFormat(startHours, startMinutes, finishHours, finishMinutes, lessonName)** |
|  | **return newLesson** |
|  |  |
|  |  |
|  | **def printLesson(lesson):** |
|  | **print(f"\nНазва дисципліни - {lesson.lessonName}.")** |
|  | **print(** |
|  | **f"Час проведення - {lesson.start.hours}:{str(lesson.start.minutes).zfill(2)} - "** |
|  | **f"{lesson.finish.hours}:{str(lesson.finish.minutes).zfill(2)}")** |
|  | **return None** |
|  |  |
|  |  |
|  | **def inputLessons(fileName):** |
|  | **answer = ""** |
|  | **lessonList = getList(fileName)** |
|  | **while answer != "т":** |
|  | **lesson = inputLesson()** |
|  | **lessonList = insertSorted(lesson, lessonList)** |
|  | **answer = input("Введіть 'т', якщо розклад заповнений.\n")** |
|  | **CheckInterval(lessonList)** |
|  | **with open(fileName, 'wb') as file:** |
|  | **for lesson in lessonList:** |
|  | **\_pickle.dump(lesson, file)** |
|  | **return None** |
|  |  |
|  |  |
|  | **def outputLessons(fileName):** |
|  | **print("Виводимо розклад з файлу:")** |
|  | **lesson = 0** |
|  | **with open(fileName, 'rb') as file:** |
|  | **try:** |
|  | **lesson = \_pickle.load(file)** |
|  | **printLesson(lesson)** |
|  | **except EOFError:** |
|  | **print("Розклад пустий.")** |
|  | **while lesson:** |
|  | **try:** |
|  | **lesson = \_pickle.load(file)** |
|  | **printLesson(lesson)** |
|  | **except EOFError:** |
|  | **break** |
|  | **return None** |
|  |  |
|  |  |
|  | **def askToDelete(fileName):** |
|  | **answer = input("Бажаєте стерти файл? Напишіть 'т' якщо ви хочете стерти файл.")** |
|  | **if answer == 'т':** |
|  | **f = open(fileName, 'wb')** |
|  | **f.close()** |
|  | **return None** |
|  |  |
|  |  |
|  | **def insertSorted(lesson, lessonList):** |
|  | **if lessonList is None:** |
|  | **lessonList = []** |
|  | **lessonList.append(lesson)** |
|  | **else:** |
|  | **inserted = False** |
|  | **for i in range(len(lessonList)):** |
|  | **if lesson.start.hours \* 60 + lesson.start.minutes < \** |
|  | **lessonList[i].start.hours \* 60 + lessonList[i].start.minutes and not inserted:** |
|  | **lessonList.insert(i, lesson)** |
|  | **inserted = True** |
|  | **if not inserted:** |
|  | **lessonList.append(lesson)** |
|  | **return lessonList** |
|  |  |
|  |  |
|  | **def CheckInterval(lessonList):** |
|  | **if len(lessonList) > 1:** |
|  | **for i in range(len(lessonList) - 1):** |
|  | **difference = lessonList[i+1].start.hours \* 60 + lessonList[i+1].start.minutes - lessonList[i].finish.hours\*60 - lessonList[i].finish.minutes** |
|  | **if difference > 45:** |
|  | **print("Між даними дисциплінами дуже велика перерва:")** |
|  | **printLesson(lessonList[i])** |
|  | **printLesson(lessonList[i+1])** |
|  | **if 5 > difference > 0:** |
|  | **print("Між даними дисциплінами дуже мала перерва:")** |
|  | **printLesson(lessonList[i])** |
|  | **printLesson(lessonList[i + 1])** |
|  | **if difference < 0:** |
|  | **print("Одна дисципліна накладається на іншу:")** |
|  | **printLesson(lessonList[i])** |
|  | **printLesson(lessonList[i + 1])** |
|  | **return None** |
|  |  |
|  |  |
|  | **def getList(fileName):** |
|  | **lessonList = []** |
|  | **with open(fileName, 'rb') as file:** |
|  | **try:** |
|  | **lesson = \_pickle.load(file)** |
|  | **lessonList.append(lesson)** |
|  | **except EOFError:** |
|  | **return lessonList** |
|  | **while lesson:** |
|  | **try:** |
|  | **lesson = \_pickle.load(file)** |
|  | **lessonList.append(lesson)** |
|  | **except EOFError:** |
|  | **break** |
|  | **return** |

**main**

|  | **import Lab\_2\_module** |
| --- | --- |
|  |  |
|  | **fileName = "lessonData.dat"** |
|  |  |
|  | **Lab\_2\_module.inputLessons(fileName)** |
|  |  |
|  | **Lab\_2\_module.outputLessons(fileName)** |
|  |  |
|  | **Lab\_2\_module.askToDelete(fileName)** |
|  |  |

**Результати на С++ -**

****

**Результати на Python -**

****

**Висновок -** Отже, ми навчилися створювати програмним шляхом бінарні файли та організовувати роботу з ними мовами С++ та Python.