Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Катедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1.2 з дисципліни

«Основи програмування 2.

Модульне програмування»

«Файли даних. Бінарні файли»

Варіант 18

Виконав студент ІП-11 Лесів Владислав Ігорович

Перевірив Вітковська Ірина Іванівна

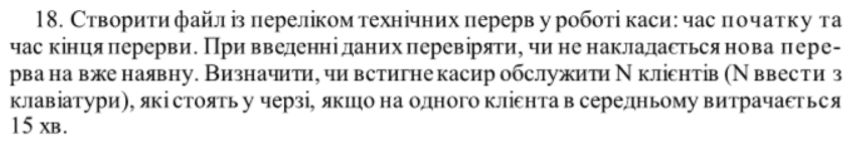
Київ 2022

**Лабораторна робота 1.2**

**Файли даних. Бінарні файли.**

**Мета** – вивчити особливості створення і обробки бінарних файлів.

**Варіант №18.**

****

**Постановка задачі.** Результатом розв’язку є ствердна або заперечна відповідь на питання, чи встигне касир обслужити клієнтів, так, як це вказано в умові задачі відповідно до заданого файлу. Для визначення результату повинен бути заданий файл з переліком технічних перерв у форматі «час\_початку час\_кінця…», а також кількість клієнтів N. Інших початкових даних для розв’язку не потрібно.

Математичне формулювання задачі зводиться до знаходження загального часу роботи каси. Від усього робочого дня (припустімо, каса цілодобова) віднімаємо час кожної технічної перерви. Тоді від часу в хвилинах віднімаємо N\*15 – час, який йде на обслуговування N клієнтів. Якщо отриманий результат від’ємний – часу не вистачило, інакше касир встигне обслужити клієнтів.

**Виконання мовою C++.**

**Код програми:**

**binFiles.h**

#pragma once

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

void inputData(int);

vector<int> readAndTranslate(string);

void verify(int, string);

**Lab2\_cpp.cpp**

#include "binFiles.h"

int main()

{

setlocale(LC\_ALL, "ukr");

int act;

cout << "Очистити данi файлу (0) чи додати (1) до iснуючих? : "; //Додавання до файлу чи очищення

cin >> act;

inputData(act);

int n;

cout << "Уведiть кiлькiсть клiєнтiв: ";

cin >> n;

verify(n, "input.bin");

}

**binFiles.cpp**

#include "binFiles.h"

void inputData(int act) {

fstream inFile;

if (act == 0) { //Якщо обрано очищення файлу

inFile.open("input.bin", ios::out | ios::binary);

inFile.close();

}

vector<int> aMins = readAndTranslate("input.bin"); //Отримуємо масив введених

int tMins[2] = { 0,0 };

int pos = 0;

cin.ignore();

string time;

cout << "\nпочаток (stop для завершення роботи): ";

getline(cin,time);

inFile.open("input.bin", ios::app | ios::binary);

string temp;

while (time != "stop") {

int h = -1, m = -1;

if (isdigit(time[0]) && isdigit(time[1]) && isdigit(time[3]) && isdigit(time[4])) { //Перевірка на коректність

temp = ""; temp += time[0]; temp += time[1];

h = stoi(temp);

temp = ""; temp += time[3]; temp += time[4];

m = stoi(temp);

}

if (time[2] != ':' || h < 0 || h>23 || m < 0 || m>59) {

cout << "Некоректний ввiд. Неправильний формат даних." << endl;

if (pos == 1) {

cout << "Початок перерви не дiйсний" << endl;

pos = 0;

}

}

else { //Якщо коректно, додаємо у проміжковий масив

if (pos == 0) {

tMins[0] = h \* 60 + m;

pos = 1;

}

else {

if (h \* 60 + m > tMins[0])

tMins[1] = h \* 60 + m;

else

cout << "Некоректний ввiд. Початок перерви не дiйсний" << endl;

pos = 0;

}

}

int inc = 0;

if (tMins[0] == 0 || tMins[1] == 0)

inc = 1;

for (int i = 0;i < aMins.size(); i += 2)

if (inc == 0 && (tMins[0] >= aMins[i] && tMins[0]<aMins[i + 1] || tMins[1]>aMins[i] && tMins[1] <= aMins[i + 1] || tMins[0] <= aMins[i] && tMins[1] >= aMins[i + 1])) {

cout << "Перерва накладається на iснуючу" << endl;

inc = 1;

}

if (inc == 0) { //Не накладається - додаємо у сталий масив

aMins.push\_back(tMins[0]);

aMins.push\_back(tMins[1]);

inFile.write((char\*)&tMins[0], sizeof(int));

inFile.write((char\*)&tMins[1], sizeof(int));

}

if (pos == 0) {

tMins[0] = 0;

tMins[1] = 0;

cout << "початок (stop для завершення роботи): ";

}

else

cout << "кiнець (stop для завершення роботи): ";

cin >> time;

}

inFile.close();

}

vector<int> readAndTranslate(string name) {

ifstream inFile(name, ios::binary);

vector<int>arrMins;

int aMin, m=0;

cout << "Вмiст файлу: \n";

while (inFile.read((char\*)&aMin, sizeof(int))) { //Вичитуємо перерви з файлу і виводимо їх на консоль

cout << aMin/60 << ":"<<aMin-(aMin/60)\*60<<" ";

if (m == 0) {

cout << "- ";

m = 1;

}

else

m = 0;

arrMins.push\_back(aMin);

}

inFile.close();

return arrMins;

}

void verify(int n, string name) {

int time = 24 \* 60;

vector<int> aMins = readAndTranslate(name);

for (int i = 0; i < aMins.size(); i += 2) //Віднімаємо від усього часу час кожної з перерв

time -= (aMins[i + 1] - aMins[i]);

cout << "\nВи маєте в запасi " << time << " хвилин." << endl;

cout << "На " << n << " клiєнтiв знадобиться " << n \* 15 << " хвилин." << endl;

time -= n \* 15; //Розрахунок 15 хв на клієнта

if (time < 0)

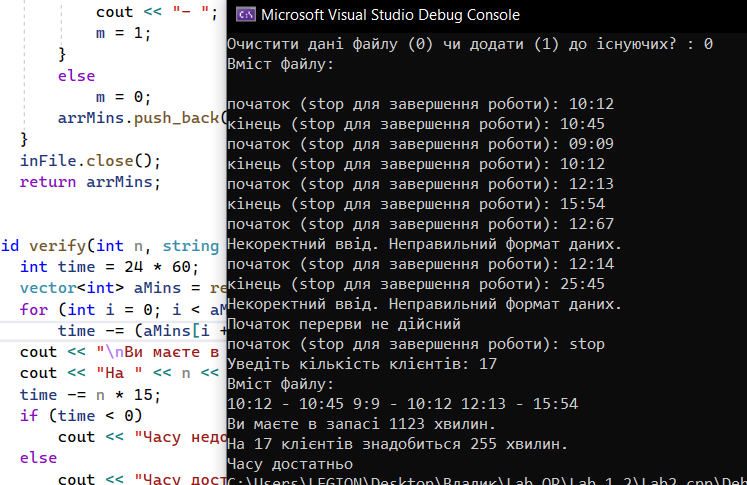
cout << "Часу недостатньо";

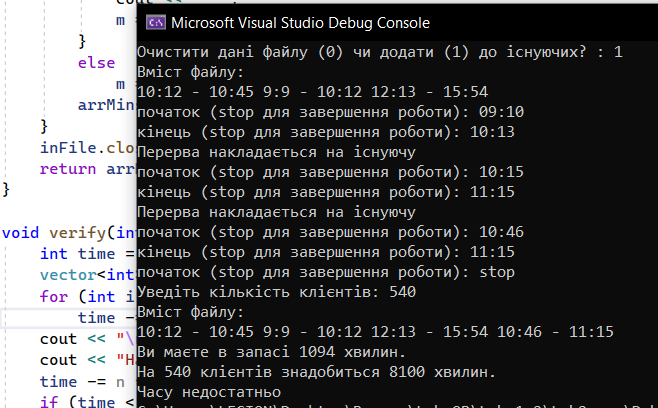
else

cout << "Часу достатньо";

}

**Випробування алгоритму.**

****

****

**Виконання мовою Python.**

**Код програми:**

**Lab2\_py.py**

from binFiles import \*

act=int(input("Очистити дані файлу (0) чи додати (1) до існуючих? : "))

inputData(act)

n=int(input("Уведіть кількість клієнтів: "))

verify(n, "input.bin")

**binFiles.py**

def inputData(act):

if act==0: #Якщо обрано очищення файлу

with open("input.bin","wb"):

pass

aMins=readAndTranslate("input.bin") #Отримуємо масив уже введених перерв

tMins=[0,0]

pos=0

print("введення відбувається у форматі ГГ:ХХ")

time=input("початок (stop для заверешення роботи): ")

with open("input.bin","ab") as inFile:

while time!="stop":

h,m=-1,-1

if time[:2].isdigit() and time[3:].isdigit(): #Перевірка на коректність

h,m=int(time[:2]),int(time[3:])

if time[2]!=":" or h<0 or h>23 or m<0 or m>59:

print("Некоректний ввід. Неправильний формат даних.")

if pos==1:

print("Початок перерви не дійсний")

pos=0

elif pos==0: #Якщо коректно, додаємо у проміжковий масив перерви

tMins[0]=h\*60+m

pos=1

elif h\*60+m>tMins[0]:

tMins[1]=h\*60+m

pos=0

else:

print("Некоректний ввід. Початок перерви не дійсний.")

pos=0

inc=0

if tMins[0]==0 or tMins[1]==0:

inc=1

for i in range(0, len(aMins), 2):

if inc==0 and (tMins[0]>=aMins[i] and tMins[0]<aMins[i+1] or tMins[1]>aMins[i] and tMins[1]<=aMins[i+1] or tMins[0]<=aMins[i] and tMins[1]>=aMins[i+1]):

print("Перерва накладається на існуючу")

inc=1

if inc==0: #Не накладається - додаємо у сталий масив перерв

aMins.append(tMins[0])

aMins.append(tMins[1])

inFile.write(bytes(str(tMins[0])+" "+str(tMins[1])+" ", encoding="utf-8"))

if pos==0:

tMins=[0,0]

time=input("початок (stop для заверешення роботи): ")

else:

time=input("кінець (stop для заверешення роботи): ")

def readAndTranslate(name):

with open(name,"rb") as inFile:

arrMins=[int(i) for i in inFile.read().split()] #Отримуємо масив перерв

print("Вміст файлу:")

m=0

for i in arrMins:

print(str(i//60)+":"+str(i-(i//60)\*60), end=" ")

if m==0:

print("- ", end="")

m=1

else:

m=0

print()

return arrMins

def verify(n, name):

time=24\*60

aMins=readAndTranslate(name)

for i in range(0, len(aMins), 2): #Віднімаємо від усього часу час кожної з перерв

time-=(aMins[i+1]-aMins[i])

print("Ви маєте в запасі", time, "хвилин.")

print("На", n, "клієнтів знадобиться", n\*15, "хвилин.")

time-=n\*15 #Розрахунок 15 хв на клієнта

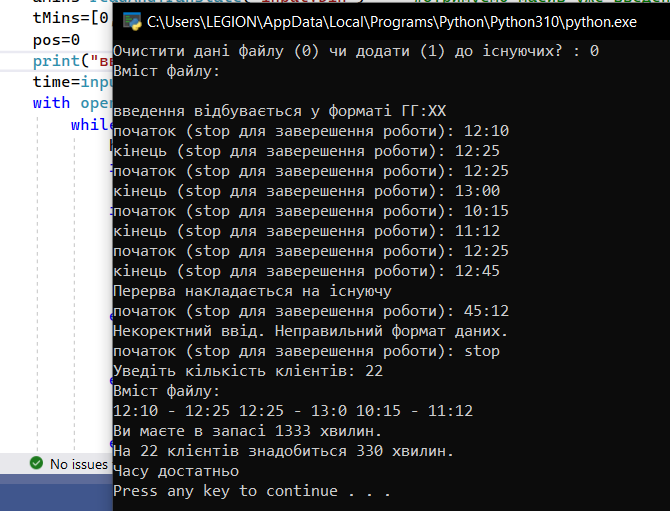
if time<0:

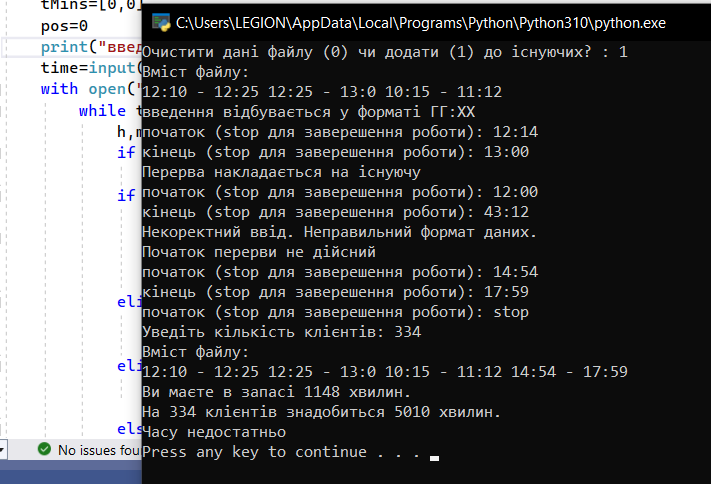
print("Часу недостатньо")

else:

print("Часу достатньо")

**Випробування алгоритму.**

****

****

**Висновок.** Отже, у цій роботі я вивчив особливості створення і обробки бінарних файлів даних. У результаті лабораторної роботи було розроблено програми, які виконують задачу відповідно до постановки. Використовуючи роботу з бінарними файлами для читання та запису необхідної інформації, функції для введення та обробки тексту, прочитаного з файлу, за умовою задачі та перевірки умов й виведення відповіді на консоль, отримуємо коректний результат.