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. Модульне програмування»

«Файли даних. Бінарні файли»

Варіант 24

Виконав студент ІП-11 Печковський Олександр Костянтинович

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна

(прізвище, ім'я, по батькові)

Київ 2022

**Умова задачі:**

Створити файл із списком працівників підприємства: прізвище, дата народження, дата прийому на роботу. Вивести список працівників, у яких день народження у поточному місяці та які пропрацювали на підприємстві не менше 5-ти років. Створити новий файл з інформацією про співробітників, які оформилися на роботу на дане підприємство у віці не старше 25-ти років та пропрацювали на ньому не менше 10-ти років.

**Код на С++:**

**Lab2C++.cpp**

#include "Header.h"

int main()

{

string first\_file\_path = "first file.txt";

string second\_file\_path = "second file.txt";

input\_file(first\_file\_path);

cout << "\nFirst file:\n";

output\_file(first\_file\_path);

cout << "\nBirthday in this month and work experience equals or more than 5 years: " << endl;

birthday\_in\_this\_month(first\_file\_path);

cout << endl;

create\_second\_file(first\_file\_path, second\_file\_path);

cout << "\nStarted career in 25 y.o. or less, and work experience equals or more than 10 years (second file):" << endl;

output\_file(second\_file\_path);

return 0;

}

**Header.h**

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma once

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <ctime>

using namespace std;

struct date;

struct employee;

void input\_file(string);

void output\_file(string);

void birthday\_in\_this\_month(string);

void create\_second\_file(string, string);

template <typename T1, typename T2>

int get\_years\_between\_dates(T1, T2);

date get\_system\_date();

vector<string> split(string, char);

**functions.cpp**

#include "Header.h"

template <typename T1, typename T2>

int get\_years\_between\_dates(T1 start\_date, T2 end\_date)

{

int years = end\_date.year - start\_date.year;

if (start\_date.month > end\_date.month || (start\_date.month == end\_date.month && start\_date.day > end\_date.day))

{

years -= 1;

}

return years;

}

vector<string> split(string str, char separator)

{

vector<string> res;

string slice = "";

str += " ";

for (int i = 0; i < str.length(); i++) {

if (str[i] == separator)

{

if (slice.length() > 0) {

res.push\_back(slice);

slice = "";

}

}

else {

slice += str[i];

}

}

res.push\_back(slice);

return res;

}

struct date {

int day;

int month;

int year;

string get\_format() {

string str\_day = to\_string(day);

string str\_month = to\_string(month);

if (day < 10) {

str\_day = '0' + to\_string(day);

}

if (month < 10) {

str\_month = '0' + to\_string(month);

}

return str\_day + '.' + str\_month + '.' + to\_string(year);

}

};

date get\_system\_date()

{

date res;

time\_t theTime = time(NULL);

struct tm\* aTime = localtime(&theTime);

res.day = aTime->tm\_mday;

res.month = aTime->tm\_mon + 1;

res.year = aTime->tm\_year + 1900;

return res;

}

struct employee {

struct date {

int day;

int month;

int year;

bool check(string line) {

vector<string> dmy = split(line, '.');

day = stoi(dmy[0]);

month = stoi(dmy[1]);

year = stoi(dmy[2]);

if (day > 31 || month > 12) {

cout << "Entered invalid date" << endl;

return 0;

}

return 1;

}

string get\_format() {

string str\_day = to\_string(day);

string str\_month = to\_string(month);

if (day < 10) {

str\_day = '0' + to\_string(day);

}

if (month < 10) {

str\_month = '0' + to\_string(month);

}

return str\_day + '.' + str\_month + '.' + to\_string(year);

}

};

char surname[50];

date birthday;

date start\_career;

void print() {

cout << "Surname: " << surname << "\t\t";

cout << "Birthday: " << birthday.get\_format() << "\t\t";

cout << "Started career: " << start\_career.get\_format() << endl;

}

};

void input\_file(string file\_path)

{

cout << "Choose input mode:" << endl << "1) create new file" << endl << "2) append info to the file" << endl << "3) open existing file" << endl;

int input\_mode;

cin >> input\_mode;

ofstream fileout;

while (input\_mode != 1 && input\_mode != 2 && input\_mode != 3) {

cout << "Invalid input mode. Try again.\n";

cin >> input\_mode;

}

if (input\_mode != 3)

{

if (input\_mode == 1)

{

fileout.open(file\_path);

}

if (input\_mode == 2)

{

fileout.open(file\_path, ios::app);

}

if (!fileout.is\_open()) {

cout << "ERROR: Could not open";

}

else {

string line;

employee person;

cin.ignore();

cout << "Enter information about the employees in format [surname dd.mm.yyyy dd.mm.yyyy] (send empty line to finish):\n";

getline(cin, line);

while (line!="") {

vector<string> words = split(line, ' ');

strcpy\_s(person.surname, words[0].c\_str());

if (person.birthday.check(words[1]) && (person.start\_career.check(words[2]))) {

fileout.write((char\*)&person, sizeof(employee));

}

getline(cin, line);

}

}

}

fileout.close();

}

void output\_file(string file\_path)

{

ifstream filein;

filein.open(file\_path);

if (!filein.is\_open()) {

cout << "ERROR: Could not open";

}

else {

employee person;

while (filein.read((char\*)&person, sizeof(employee)))

{

person.print();

}

}

filein.close();

}

void birthday\_in\_this\_month(string file\_path)

{

ifstream filein;

filein.open(file\_path);

if (!filein.is\_open()) {

cout << "ERROR: Could not open";

}

else {

employee person;

date sys\_date = get\_system\_date();

while (filein.read((char\*)&person, sizeof(employee)))

{

int work\_experience = get\_years\_between\_dates(person.start\_career, sys\_date);

if (person.birthday.month == sys\_date.month && work\_experience >= 5)

{

person.print();

}

}

}

filein.close();

}

void create\_second\_file(string filein\_name, string fileout\_name)

{

ifstream filein;

ofstream fileout;

filein.open(filein\_name);

fileout.open(fileout\_name);

if (!filein.is\_open() || !fileout.is\_open()) {

cout << "ERROR: Could not open";

}

else {

employee person;

date sys\_date = get\_system\_date();

while (filein.read((char\*)&person, sizeof(employee)))

{

int start\_career\_age = get\_years\_between\_dates(person.birthday, person.start\_career);

int work\_experience = get\_years\_between\_dates(person.start\_career, sys\_date);

if (start\_career\_age <= 25 && work\_experience >= 10)

{

fileout.write((char\*)&person, sizeof(employee));

}

}

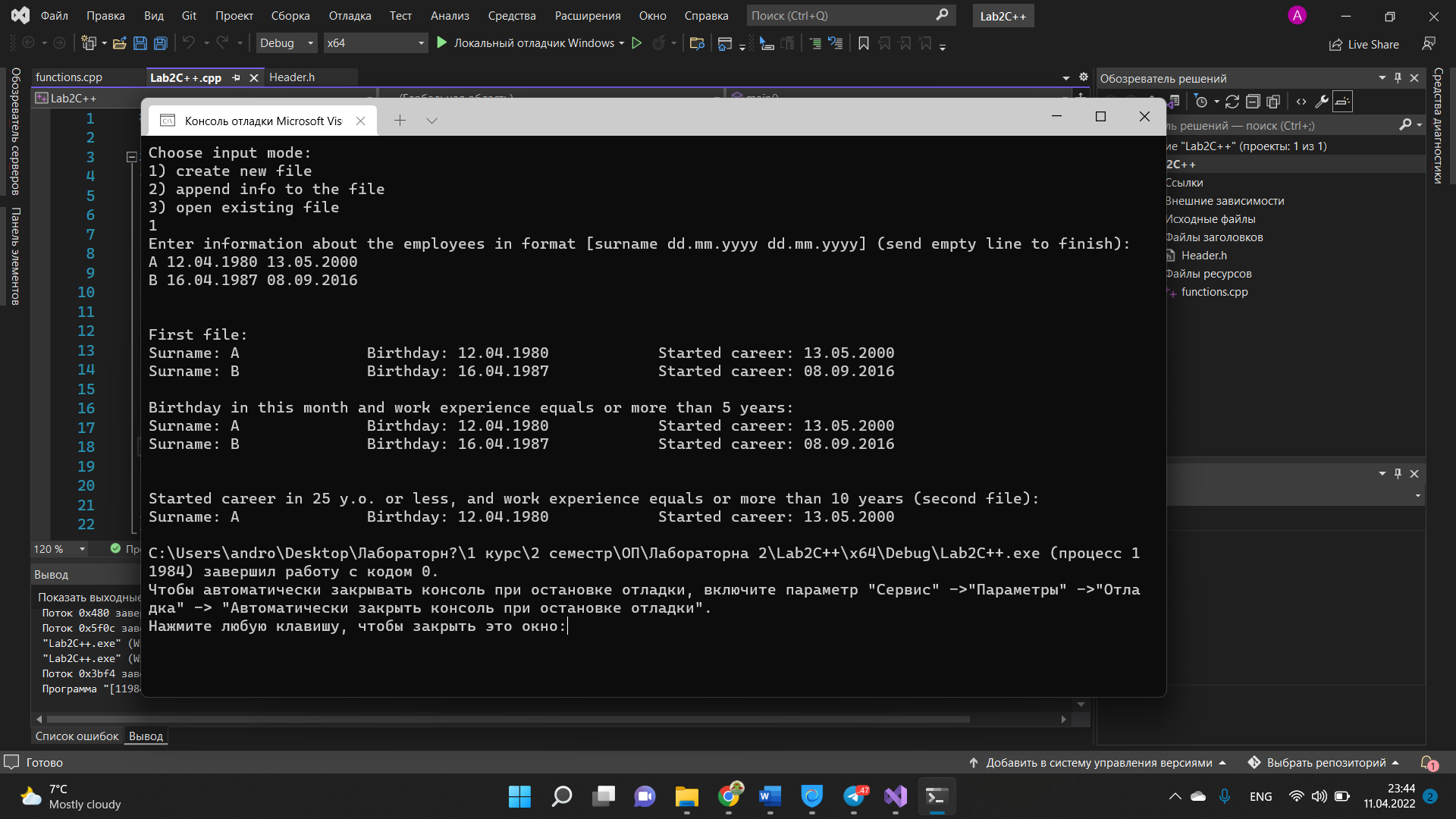
}

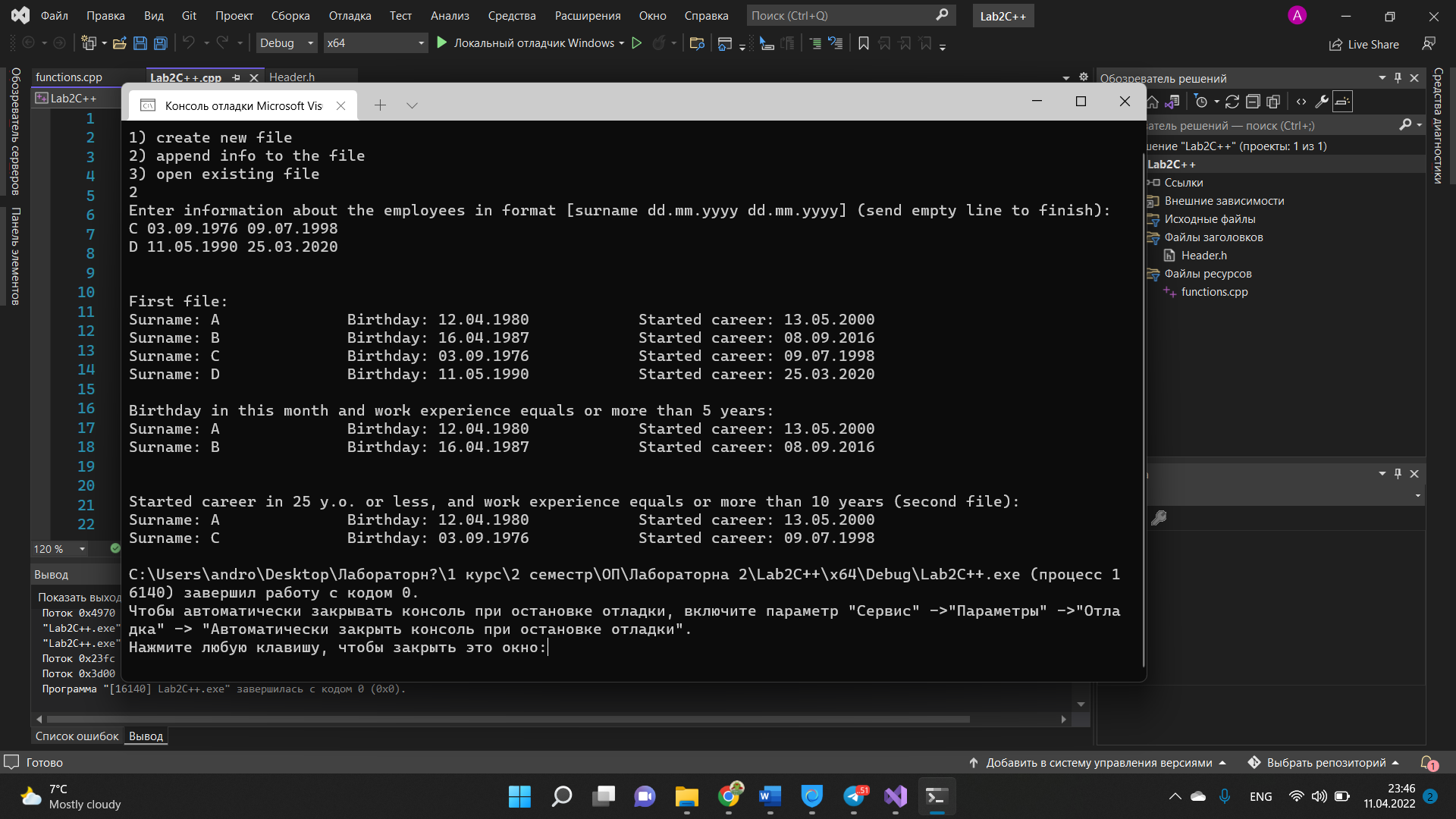
filein.close();

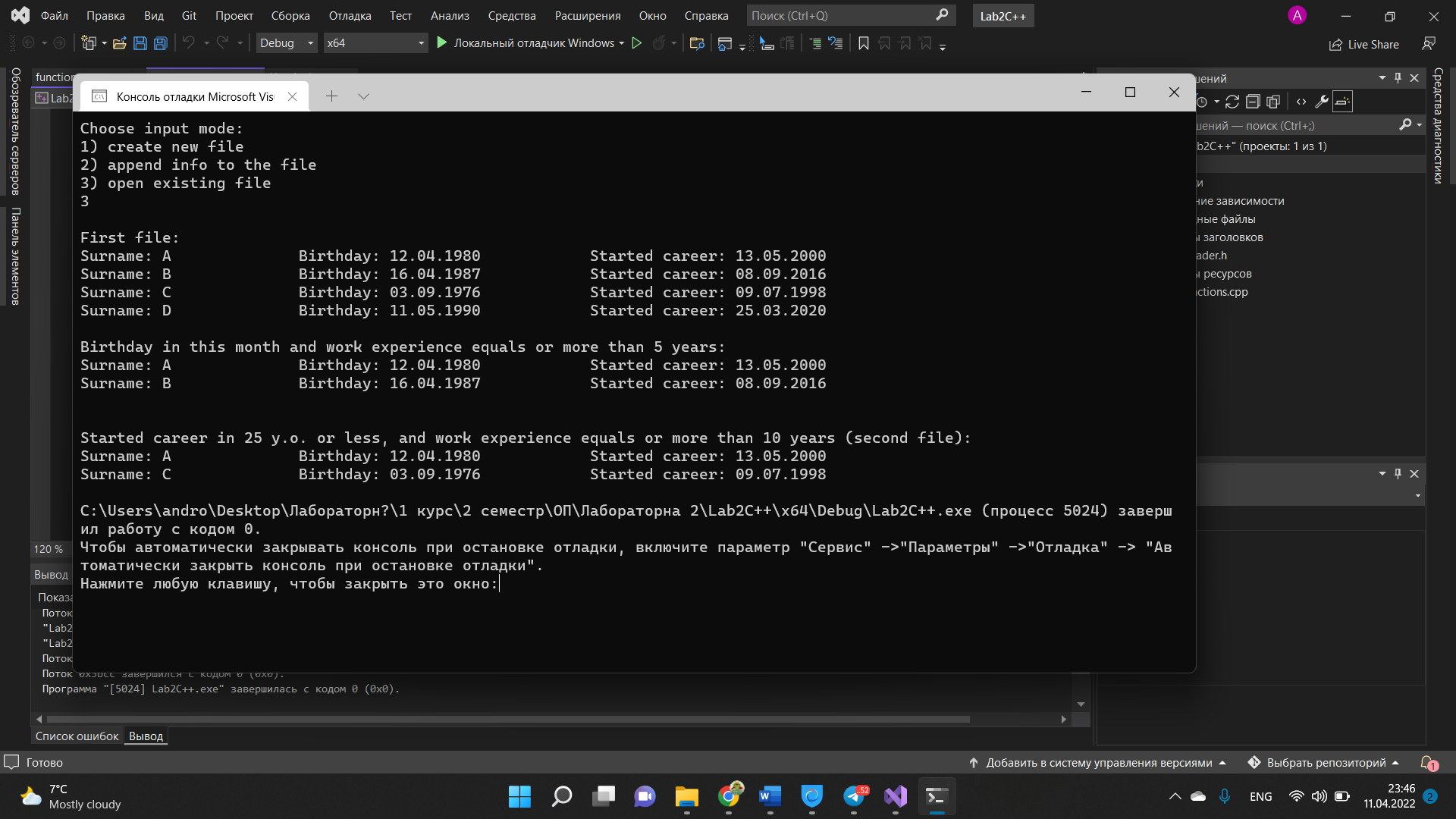
fileout.close();

}

**Скріншоти роботи програми на C++:**

****

****

****

**Код на Python:**

**Lab2Python.py**

from functions import \*

first\_file\_path = "first file.txt"

second\_file\_path = "second file.txt"

input\_file(first\_file\_path)

print("First file:")

output\_file(first\_file\_path)

print("\nBirthday in this month and work experience equals or more than 5 years: ")

birthday\_in\_this\_month(first\_file\_path)

print()

create\_second\_file(first\_file\_path, second\_file\_path)

print("Started career in 25 y.o. or less, and work experience equals or more than 10 years (second file):")

output\_file(second\_file\_path)

**functions.py**

import pickle, os.path

from datetime import datetime

def input\_file(filename):

print("Choose input mode:\n1) create new file\n2) append info to the file\n3) just open file")

input\_mode = int(input())

while input\_mode!=1 and input\_mode!=2 and input\_mode!=3:

print("Invalid input mode. Please try again.")

input\_mode = int(input())

if input\_mode!=3:

if input\_mode == 1:

file = open(filename, "wb")

elif input\_mode == 2:

file = open(filename, "ab")

line = input("Enter information about the employees in format [surname dd.mm.yyyy dd.mm.yyyy] "

"(send empty line to finish):\n")

while line != "":

words = line.split()

birthday = words[1].split('.')

start\_career = words[2].split('.')

employee = {

"surname": words[0],

"birthday": {"day": int(birthday[0]),

"month": int(birthday[1]),

"year": int(birthday[2])},

"start\_career": {"day": int(start\_career[0]),

"month": int(start\_career[1]),

"year": int(start\_career[2])}

}

pickle.dump(employee, file)

line = input()

file.close()

def get\_system\_date():

current\_datetime = datetime.now()

sys\_date = {

"day": current\_datetime.day,

"month": current\_datetime.month,

"year": current\_datetime.year

}

return sys\_date

def get\_years\_between\_dates(start\_date, end\_date):

years = end\_date["year"] - start\_date["year"]

if start\_date["month"] > end\_date["month"] or (start\_date["month"] == end\_date["month"] and

start\_date["day"] > end\_date["day"]):

years -= 1

return years

def birthday\_in\_this\_month(filename):

if os.path.isfile(filename):

with open(filename, 'rb') as file:

sys\_date = get\_system\_date()

size = file.seek(0, 2)

file.seek(0)

while file.tell() < size:

employee = pickle.load(file)

work\_experience = get\_years\_between\_dates(employee["start\_career"], sys\_date)

if employee["birthday"]["month"] == sys\_date["month"] and work\_experience >= 5:

print\_employee(employee)

else:

print("File doesn't exist")

def create\_second\_file(filein\_name, fileout\_name):

if os.path.isfile(filein\_name):

with open(filein\_name, 'rb') as filein:

with open(fileout\_name, 'wb') as fileout:

sys\_date = get\_system\_date()

size = filein.seek(0, 2)

filein.seek(0)

while filein.tell() < size:

employee = pickle.load(filein)

start\_career\_age = get\_years\_between\_dates(employee["birthday"], employee["start\_career"])

work\_experience = get\_years\_between\_dates(employee["start\_career"], sys\_date)

if start\_career\_age <= 25 and work\_experience >= 10:

pickle.dump(employee, fileout)

def get\_format(date):

str\_day = str(date["day"])

str\_month = str(date["month"])

if date["day"] < 10:

str\_day = '0' + str\_day

if date["month"] < 10:

str\_month = '0' + str\_month

return str\_day + '.' + str\_month + '.' + str(date["year"])

def print\_employee(employee):

print("Surname: " + employee["surname"] + "\t \tBirthday: " + get\_format(employee["birthday"])

+ "\t \t Started career: " + get\_format(employee["start\_career"]))

def output\_file(filename):

if os.path.isfile(filename):

with open(filename, 'rb') as file:

size = file.seek(0, 2)

file.seek(0)

while file.tell() < size:

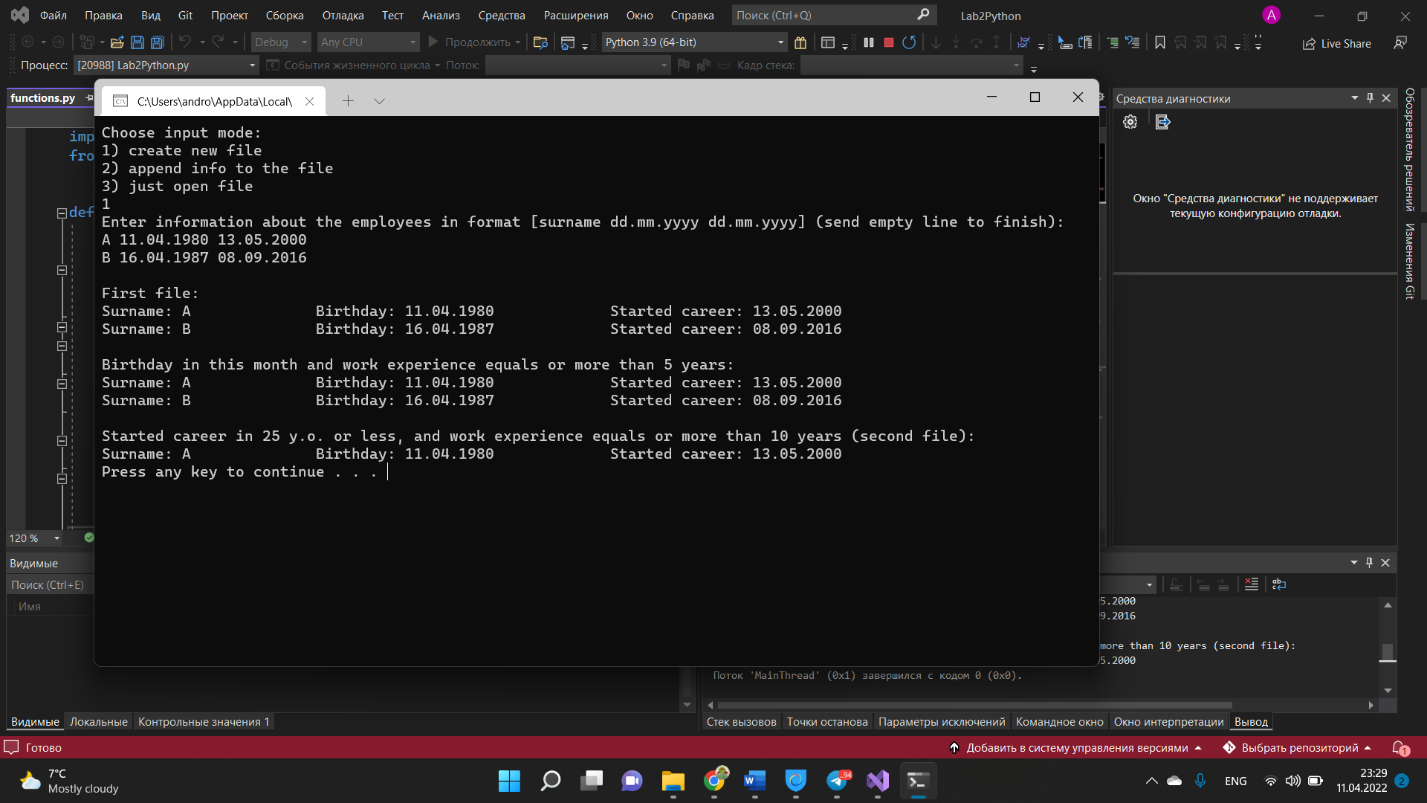
employee = pickle.load(file)

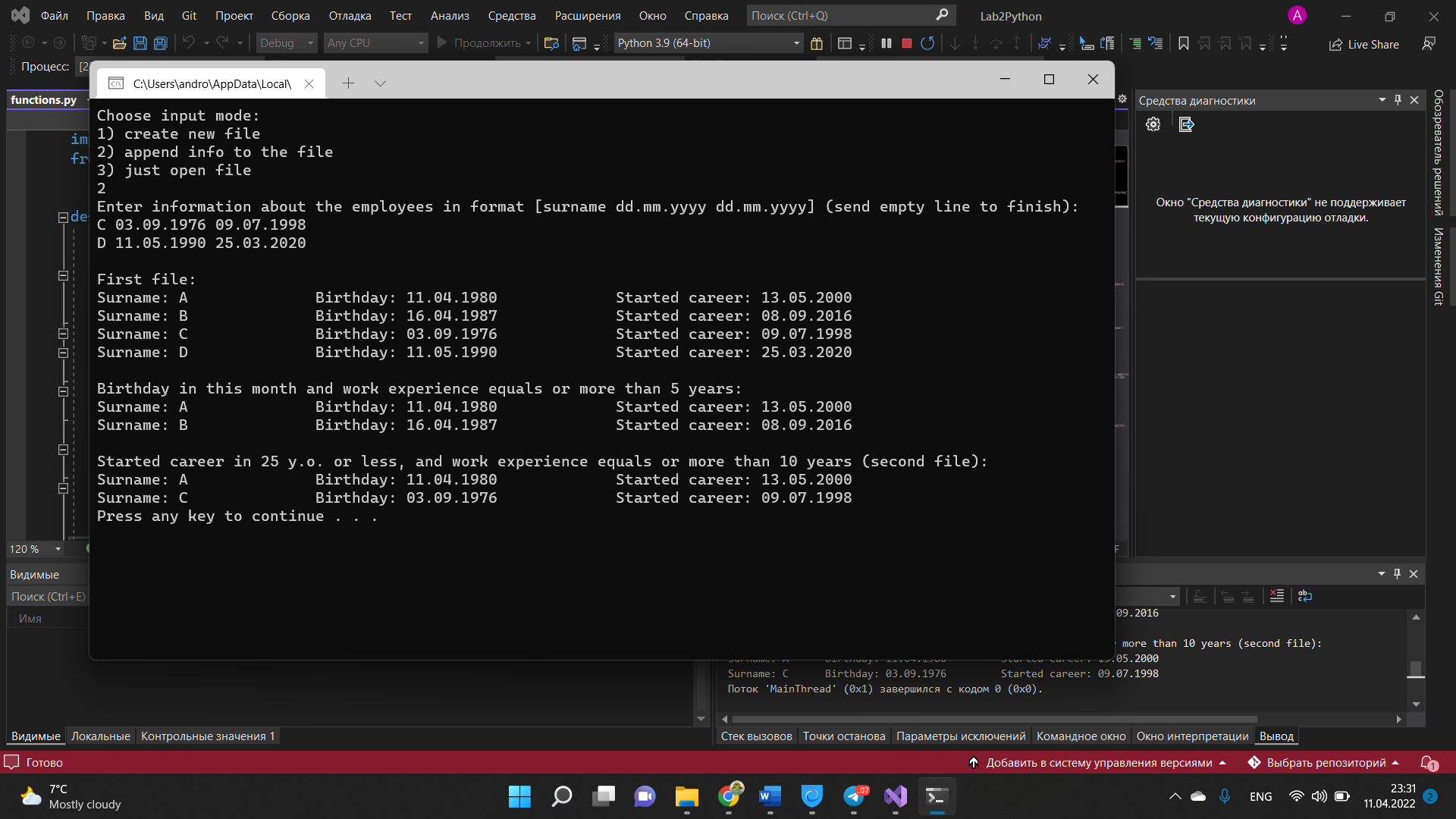
print\_employee(employee)

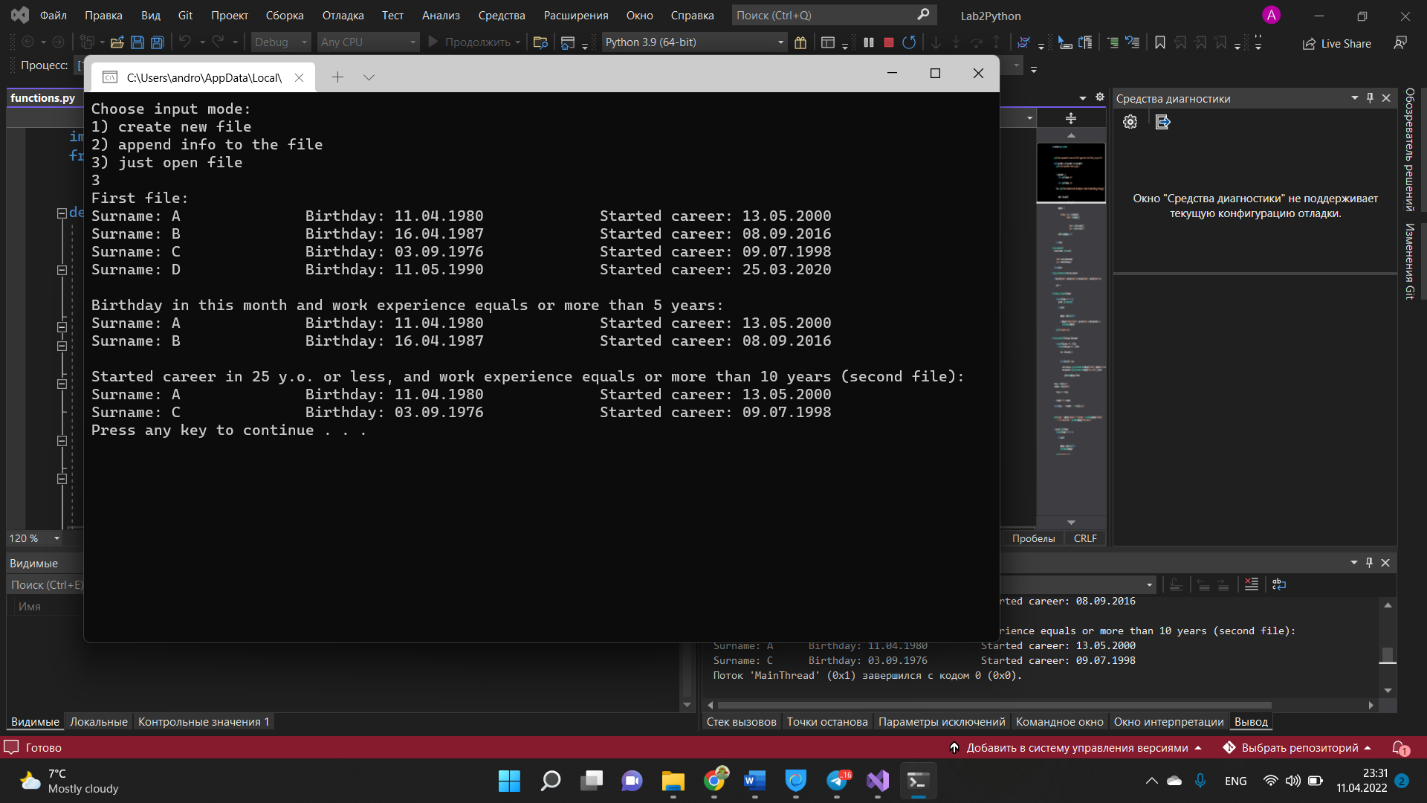
else:

print("File doesn't exist")

**Скріншоти роботи програми на Python:**

****

****

****

**Висновок:** під час виконання лабораторної роботи я вивчив особливості створення і обробки бінарних файлів даних.