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.
Методології програмування»

«Бінарні файли»

Варіант 10

Виконав студент ІП-11, Друзенко Олександра Юріївна
(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна
(прізвище, ім'я, по батькові)

Київ 2022

Мета: вивчити особливості створення і обробки бінарних файлів.

Постановка задачі: Створити бінарний файл зі списком пацієнтів, де вказані наступні данні: прізвище, дата попереднього відвідування, дата наступного відвідування. Якщо час, на який пацієнт записаний, минув, то потрібно видалити цей запис з файлу. Потім потрібно створити два файли. Перший із записами про вторинних пацієнтів (якщо від часу попереднього відвідування минуло менше 10-ти днів). Другий файл з рештою пацієнтів.

Виконання мовою С++:

1)код:

lib.h:

```
#pragma once
#include <iostream>
#include <iomanip>
#include <stdlib.h>
#include <ctime>
#include <Windows.h>
#include <fstream>
#include <string>
using namespace std;

struct TimeVisit {
    int day, month, year;
};
struct Patient {
    char pib[100];
    TimeVisit pVisit, fVisit;

};

void createKatalog(string name); //створення бінарного файлу
void outputKatalog(string name); //виведення файлу в консоль
void copyFile(string name); //копіювання бінарного файлу
void createNewKatalogs(string name); //сортування пацієнтів по файлах
void cleanKatalog(string name); //очищаємо записи, які вже відбулися
void deleteFile(string name); //очищення файлу
```

source.cpp:

```
#include "lib.h"

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);

    createKatalog("Patient");
    outputKatalog("Patient");
    cleanKatalog("Patient");
```

```
    createNewKatalogs("Patient");
    cout << "\n=====Файл вторинних=====";
    outputKatalog("secondary");
    cout << "\n\n=====Файл інших=====";
    outputKatalog("other");
    deleteFile("Patient");
}
```

lib.cpp:

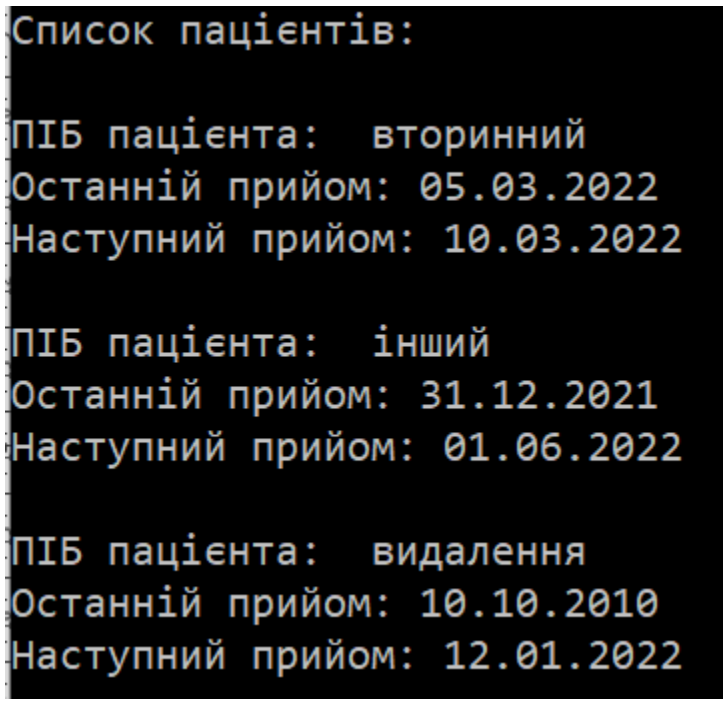
```
#include "lib.h"

//============створення бінарного файлу===========
void createKatalog(string name) {
    char c;
    Patient human = {};
    ofstream fOut(name, ios::app | ios::out | ios::binary);

    do {
        cout << "введіть прізвище пацієнта: ";
        cin.getline(human.pib, 100);
        cout << "введіть дату попереднього прийому (dd.mm.year): ";
        scanf_s("%d.%d.%d", &human.pVisit.day, &human.pVisit.month,
&human.pVisit.year);
        cout << "введіть дату наступного прийому (dd.mm.year): ";
        scanf_s("%d.%d.%d", &human.fVisit.day, &human.fVisit.month,
&human.fVisit.year);
        fOut.write((char*)&human, sizeof(Patient));
        cout << "Продовжити? (т/н)"; cin >> c;
        cout << endl;
        cin.ignore();
    } while ((c != 'н') && (c != 'Н'));
    fOut.close();
}
//============виведення файлу в консоль===========
void outputKatalog(string name) {
    Patient human;
    ifstream fIn(name, ios::binary);
    cout << "\nСписок пацієнтів:\n";
    while (fIn.read((char*)&human, sizeof(human))) {
        //cout << "\nПІБ пацієнта: " << human.pib;
        printf("\nПІБ пацієнта:  %s", human.pib);
        printf("\nОстанній прийом: %02d.%02d.%04d", human.pVisit.day,
human.pVisit.month, human.pVisit.year);
        printf("\nНаступний прийом: %02d.%02d.%04d  \n", human.fVisit.day,
human.fVisit.month, human.fVisit.year);
    }
    fIn.close();
    return;
}

//============копіювання бінарного файлу===========
void copyFile(string name) {
    Patient human;
    ofstream curr("curr.dat", ios::binary);
    ifstream in(name, ios::binary);
    int i;
    while (in.read((char*)&human, sizeof(human))) {
        curr.write((char*)&human, sizeof(Patient));
```

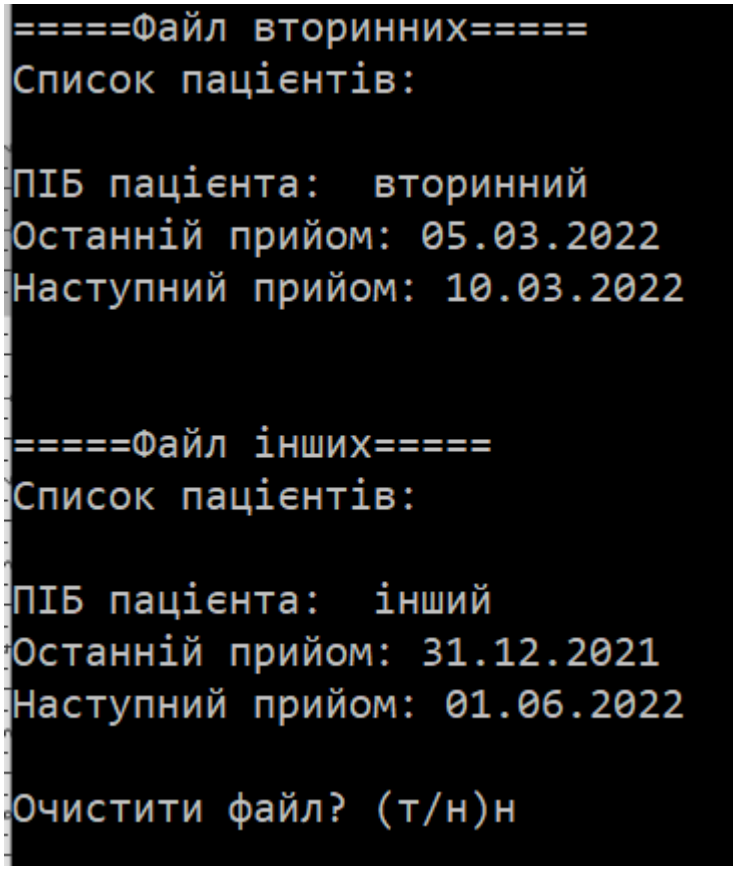
```
    }
    in.close();
    curr.close();
}
//============очищення старих записів============
void cleanKatalog(string name) {
    copyFile(name);
    Patient human;
    SYSTEMTIME tm;
    GetLocalTime(&tm);
    ifstream curr("curr.dat", ios::binary);
    ofstream in(name, ios::binary);
    while (curr.read((char*)&human, sizeof(human))) {
        if (human.fVisit.year >= tm.wYear) {
            if (human.fVisit.year == tm.wYear && human.fVisit.month >=
tm.wMonth) {
                if (human.fVisit.month == tm.wMonth && human.fVisit.day >=
tm.wDay) {
                    in.write((char*)&human, sizeof(Patient));
                }
                else if (human.fVisit.month > tm.wMonth) {
                    in.write((char*)&human, sizeof(Patient));
                }
            }
            else if (human.fVisit.year > tm.wYear) {
                in.write((char*)&human, sizeof(Patient));
            }
        }
    }
    curr.close();
    in.close();
}

//============сортування пацієнтів по файлах============
void createNewKatalogs(string name) {
    ifstream oldFile(name, ios::binary);
    ofstream FFile("secondary", ios::binary);
    ofstream SFile("other", ios::binary);
    Patient human;
    while (oldFile.read((char*)&human, sizeof(human))) {
        if (human.fVisit.year == human.pVisit.year) {
            if (human.fVisit.month == human.pVisit.month) {
                if ((human.fVisit.day - human.pVisit.day) <= 10) {
                    FFile.write((char*)&human, sizeof(Patient));
                }
            }
            else if (abs(human.fVisit.month - human.pVisit.month) == 1) {
                if (abs(human.fVisit.day - human.pVisit.day) >= 23) {
                    FFile.write((char*)&human, sizeof(Patient));
                }
                else {
                    SFile.write((char*)&human, sizeof(Patient));
                }
            }
            else {
                SFile.write((char*)&human, sizeof(Patient));
            }
        }
        else if (abs(human.fVisit.year - human.pVisit.year) == 1) {
```

```
                if (abs(human.fVisit.month - human.pVisit.month) == 11) {
                    if (abs(human.fVisit.day - human.pVisit.day) >= 23) {
                        FFile.write((char*)&human, sizeof(Patient));
                    }
                }
                else {
                    SFile.write((char*)&human, sizeof(Patient));
                }
            }
            else {
                SFile.write((char*)&human, sizeof(Patient));
            }
        }
        oldFile.close();
        FFile.close();
        SFile.close();
    }

    //============очищення файлу============
    void deleteFile(string name) {
        char c;
        cout << "\nОчистити файл? (т/н)"; cin >> c;
        if (c == 'т' || c == 'Т') {
            ofstream file(name, ios::trunc);
            file.close();
        }
    }
```

2)Випробування коду на C++:

```
введіть прізвище пацієнта: вторинний
введіть дату попереднього прийому (dd.mm.year): 05.03.2022
введіть дату наступного прийому (dd.mm.year): 10.03.2022
Продовжити? (т/н)т

введіть прізвище пацієнта: інший
введіть дату попереднього прийому (dd.mm.year): 31.12.2021
введіть дату наступного прийому (dd.mm.year): 01.06.2022
Продовжити? (т/н)т

введіть прізвище пацієнта: видалення
введіть дату попереднього прийому (dd.mm.year): 10.10.2010
введіть дату наступного прийому (dd.mm.year): 12.01.2022
Продовжити? (т/н)н
```





Виконання мовою Python

1) Код:
   module1.py:

```
import _pickle
import _datetime
```

```
#==========заповнення часу запису==========
def fillTimeDict(arr):
    D={'day':0,'month':0,'year':0}
    i=0
    for key in D:
        D[key]=int(arr[i])
        i+=1
    return D

#==========створення запису пацієнта==========
def fillDict(D):
    name=input('введіть прізвище: ')
    D['surname']= name
    a=input('введіть дату попереднього прийому:').split('.')
    D['pVisit']=fillTimeDict(a)
    a=input('введіть дату наступного прийому:').split('.')
    D['fVisit']=fillTimeDict(a)
    return D

#==========створення бінарного файлу==========
def fillFile(file_name):
    with open(file_name, "ab") as file:
        flag='т'
        while (flag=='т' or flag=='Т'):
            D_patient={}
            D_patient=fillDict(D_patient)
            _pickle.dump(D_patient, file)
            flag=input('продовжити? (т/н)')
            print()

#==========читання бінарного файлу==========
def readFile(file_name):
    patient=0
    with open(file_name, "rb") as file:
        try:
            patient = _pickle.load(file)
        except EOFError:
            print('записів немає')
        while(patient):
            print('\nпрізвище: ',patient['surname'])
            print('минулий прийом:
%02d.%02d.%04d'%(patient['pVisit']['day'],patient['pVisit']['month'],patient[
'pVisit']['year']))
            print('наступний прийом:
%02d.%02d.%04d'%(patient['fVisit']['day'],patient['fVisit']['month'],patient[
'fVisit']['year']))
            try:
                patient = _pickle.load(file)
            except EOFError:
                break

#==========копіювання бінарного файлу==========
def copyFile(file_name):
    fcopy=open('temp','wb')
    with open(file_name, "rb") as file:
        patient = _pickle.load(file)
        while(patient):
            _pickle.dump(patient, fcopy)
```

```
            try:
                patient = _pickle.load(file)
            except EOFError:
                break
    fcopy.close()


#==========очищення записів, які вже відбулися==========
def cleanKatalog(file_name):
    fcopy=open('temp','rb')
    file = open(file_name,'wb')
    file.truncate()
    now=_datetime.datetime.now()

    patient = _pickle.load(fcopy)
    while(patient):
        if (patient['fVisit']['year'] >= now.year):
            if (patient['fVisit']['year'] == now.year and
patient['fVisit']['month'] >= now.month):
                if (patient['fVisit']['month'] == now.month and
patient['fVisit']['day'] >= now.day):
                    _pickle.dump(patient, file)
                elif (patient['fVisit']['month'] > now.month):
                    _pickle.dump(patient, file)
            elif (patient['fVisit']['year'] > now.year):
                _pickle.dump(patient, file)
        try:
            patient = _pickle.load(fcopy)
        except EOFError:
            break

    fcopy.close()
    file.close()


#==========сортування пацієнтів по файлах==========
def createNewKatalogs(file_name):
    oldFile = open(file_name,'rb')
    FFile=open("secondary", 'wb')
    SFile=open("other", 'wb')
    patient = _pickle.load(oldFile)
    while (patient):
        if (patient['fVisit']['year'] == patient['pVisit']['year']):
            if (patient['fVisit']['month'] == patient['pVisit']['month']):
                if ((patient['fVisit']['day'] - patient['pVisit']['day']) <=
10):
                    _pickle.dump(patient, FFile)
            elif (abs(patient['fVisit']['month'] -
patient['pVisit']['month']) == 1):
                if (abs(patient['fVisit']['day'] - patient['pVisit']['day'])
>= 23):
                    _pickle.dump(patient, FFile)
                else:
                    _pickle.dump(patient, SFile)
            else:
                _pickle.dump(patient, SFile)
        elif (abs(patient['fVisit']['year'] - patient['pVisit']['year']) ==
1):
```

```
            if (abs(patient['fVisit']['month'] - patient['pVisit']['month'])
== 11):
                if (abs(patient['fVisit']['day'] - patient['pVisit']['day'])
>= 23):
                    _pickle.dump(patient, FFile)
            else:
                _pickle.dump(patient, SFile)
        else:
            _pickle.dump(patient, SFile)

        try:
            patient = _pickle.load(oldFile)
        except EOFError:
            break
    oldFile.close();
    FFile.close();
    SFile.close();


#==========очищення файлу==========
def cleanFile(file_name):
    c=input('Видалити записи з файлу? (т/н)')
    if (c=='т' or c=='Т'):
        file = open('bintest','wb')
        #file.truncate()
        file.close()
```

lab_2.py:

```
from module1 import *

fillFile('bintest')
readFile('bintest')
copyFile('bintest')
cleanKatalog('bintest')
print('\n======Список без старих записів======')
readFile('bintest')
createNewKatalogs('bintest')
print('\n======Вторинні пацієнти======')
readFile('secondary')
print('\n======Інші пацієнти======')
readFile('other')
cleanFile('bintest')
```

2)Випробування коду на Python:

```
введіть прізвище: вторинний
введіть дату попереднього прийому:05.03.2022
введіть дату наступного прийому:12.03.2022
продовжити? (т/н)т

введіть прізвище: інший
введіть дату попереднього прийому:12.12.2021
введіть дату наступного прийому:15.05.2022
продовжити? (т/н)т

введіть прізвище: видалення
введіть дату попереднього прийому:12.12.2019
введіть дату наступного прийому:01.10.2020
продовжити? (т/н)н


прізвище:  вторинний
минулий прийом: 05.03.2022
наступний прийом: 12.03.2022

прізвище:  інший
минулий прийом: 12.12.2021
наступний прийом: 15.05.2022

прізвище:  видалення
минулий прийом: 12.12.2019
наступний прийом: 01.10.2020
```

```
======Список без старих записів======

прізвище:  вторинний
минулий прийом: 05.03.2022
наступний прийом: 12.03.2022

прізвище:  інший
минулий прийом: 12.12.2021
наступний прийом: 15.05.2022

======Вторинні пацієнти======

прізвище:  вторинний
минулий прийом: 05.03.2022
наступний прийом: 12.03.2022

======Інші пацієнти======

прізвище:  інший
минулий прийом: 12.12.2021
наступний прийом: 15.05.2022
Видалити записи з файлу? (т/н)
```

Висновок. Отже, на цій лабораторній роботі я вивчила особливості створення і обробки бінарних файлів, надбала навички написання програм для роботи з бінарними файлами, запису та читання з них.