

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 15

Виконав студент ІП-13 Карамян Вартан Суренович  
(шифр, прізвище, ім'я, по батькові)

Перевірів Вечерковська Анастасія Сергіївна  
(прізвище, ім'я, по батькові)

# Лабораторна робота 1.2

Бінарні файли

## Варіант 15

Створити файл із списком працівників підприємства: ПІБ, дата народження, дата прийому на роботу. Вивести список співробітників, старших за 40 років, та співробітників, які працюють на підприємстві не менше 20 років. Видалити з файлу інформацію про співробітників, які працюють менше року.

**Код програми на мові C++:**

### lab1 C++.cpp

```
#include "Module.h"

int main()
{
    string fileName = initFileName();
    createList(fileName);

    vector<Employee> emps = readFile(fileName);
    outputList(emps);

    Date todayDate = inputDate();
    outputOver40yo(emps, todayDate);

    outputOver20exp(emps, todayDate);

    deleteLess1exp(emps, todayDate, fileName);

    vector<Employee> newEmps = readFile(fileName);
    outputList(newEmps);
}
```

## module.h

```
#pragma once
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
using namespace std;

struct Date
{
    int day;
    int month;
    int year;
    Date();
    void setDate();
};

struct Employee
{
    char fullName[40] = "";
    Date birthday;
    Date recrDate;
};

string initFileName();
void createList(string);
vector<Employee> readFile(string);
Employee addEmployee();
void outputEmps(Employee);
void outputList(vector<Employee>);
Date inputDate();
void outputOver40yo(vector<Employee>, Date);
void outputOver20exp(vector<Employee>, Date);
void deleteLess1exp(vector<Employee>, Date, string);
bool calculateYears(Employee, Date, string, int);
```

## module.cpp

```
#include "Module.h"

string initFileName()
{
    string name;
    cout << "Enter file name:";
    cin >> name;
    cout << "=====\\n";
    return name;
}

Date::Date()
{
    day = 0; month = 0; year = 0;
}

void Date::setDate()
{
    do
    {
        cout << "day:";
        cin >> day;
        cin.ignore();
    }
```

```

    } while (day<1 || day>31);

do
{
    cout << "month:";
    cin >> month;
    cin.ignore();
} while (month<1 || month>12);

do
{
    cout << "year:";
    cin >> year;
    cin.ignore();
} while (year < 1900 || year>2022);

}

void createList(string fileName)
{
    vector<Employee> employees;
    ofstream fout;
    char writeMode;
    cout << "Please, select the write mode.\nTo create a new file, press 1.\nTo append to an already existing file, press 2.\n>>>";
    while (true)
    {
        cin >> writeMode;
        cout << "=====\\n";
        if (writeMode == '1')
        {
            fout.open(fileName, ios::binary);
            break;
        }
        else if (writeMode == '2')
        {
            employees = readFile(fileName);
            outputList(employees);
            fout.open(fileName, ios::app | ios::binary);
            break;
        }
        else
        {
            cout << "Error. Please press 1 or 2.\\n";
        }
    }
}

Employee empl;
char mode;

if (!fout.is_open())
{
    cout << "Error of opening file!";
}
else
{
    while (true)
    {
        cout << "Do you want to add new employee?(Y/N)\\n>>>";
        cin >> mode;
        cin.ignore();
        cout << "=====\\n";
    }
}

```

```

        if (mode == 'Y')
        {
            empl = addEmployee();
            employees.push_back(empl);
            fout.write((char*)&empl, sizeof(Employee));
            cout << endl;
        }
        else if (mode == 'N')
        {
            break;
        }
    }

    fout.close();
}

}

vector<Employee> readFile(string fileName)
{
    Employee emp;
    vector<Employee> emps;
    ifstream fin;

    fin.open(fileName, ios::binary);

    if (!fin.is_open())
    {
        cout << "Could not read a file\n";
    }
    else
    {
        while (fin.read((char*)&emp, sizeof(Employee)))
        {
            emps.push_back(emp);
        }
        fin.close();
    }

    return emps;
}

Employee addEmployee()
{
    Employee empl;
    cout << "Full name:";
    cin.get(empl.fullName, 40);

    cout << "Birthday\n";
    Date birthDate;
    birthDate.setDate();
    empl.birthday = birthDate;

    cout << "Date of recruitment\n";
    Date recruitmentDate;
    recruitmentDate.setDate();
    empl.recrDate = recruitmentDate;
    cout << "===== ";
    return empl;
}

void outputList(vector<Employee> emps)
{
    cout << "List of employees:" << endl;
    for (const auto &empl: emps)

```

```

    {
        outputEmps(empl);
    }
    cout << "=====\\n";
}

void outputEmps(Employee empl)
{
    cout << empl.fullName;
    if (empl.birthday.day < 10)
        cout << "\\t\\t0" << empl.birthday.day << '.';
    else
        cout << "\\t\\t" << empl.birthday.day << '.';

    if (empl.birthday.month < 10)
        cout << '0' << empl.birthday.month;
    else
        cout << empl.birthday.month;
    cout << '.' << empl.birthday.year;

    if (empl.recrDate.day < 10)
        cout << "\\t\\t0" << empl.recrDate.day << '.';
    else
        cout << "\\t\\t" << empl.recrDate.day << '.';

    if (empl.recrDate.month < 10)
        cout << '0' << empl.recrDate.month;
    else
        cout << empl.recrDate.month;
    cout << '.' << empl.recrDate.year << endl;
}

Date inputDate()
{
    Date todayDate;
    cout << "Enter today's date\\n";
    todayDate.setDate();
    return todayDate;
}

void outputOver40yo(vector<Employee> emps, Date todayDate)
{
    cout << "=====\\n";
    cout << "Employees over 40 years old:\\n";
    bool flag;
    for (const auto &empl: emps)
    {
        flag = calculateYears(empl, todayDate, "birthday", 40);
        if (flag)
        {
            outputEmps(empl);
        }
    }
}

void outputOver20exp(vector<Employee> emps, Date todayDate)
{
    cout << "=====\\n";
    cout << "Employees over 20 years of experience:\\n";
    bool flag;
    for (const auto& empl : emps)
    {
        flag = calculateYears(empl, todayDate, "recruitment", 20);
        if (flag)
        {
            outputEmps(empl);
        }
    }
}

```

```

    }
}

void deleteLess1exp(vector<Employee> emps, Date todayDate, string fileName)
{
    ofstream fout(fileName, ios::binary);
    if (!fout.is_open())
    {
        cout << "Error of opening file!";
    }
    else
    {
        bool flag;
        for (auto &empl : emps)
        {
            flag = calculateYears(empl, todayDate, "recruitment", 1);
            if (flag)
            {
                fout.write((char*)&empl, sizeof(Employee));
            }
        }
        fout.close();
    }
    cout << "=====\\n";
    cout << "Employees with less than one year of experience were removed.\\n";
}

bool calculateYears(Employee empl, Date todayDate, string dateMode, int years)
{
    Date date;

    if (dateMode == "birthday")
        date = empl.birthday;
    else if (dateMode == "recruitment")
        date = empl.recrDate;

    bool flag = false;
    if (date.year < (todayDate.year - years))
    {
        flag = true;
    }
    else if (date.year == (todayDate.year - years))
    {
        if (date.month < todayDate.month)
        {
            flag = true;
        }
        else if (date.month == todayDate.month)
        {
            if (date.day < todayDate.day)
            {
                flag = true;
            }
        }
    }
    return flag;
}
};

```

**Код програми на мові Python:**

## main.py

```
import module as m

name_of_file = input("Enter file name:")
m.create_file(name_of_file)
list_of_employees = m.read_file(name_of_file)
m.output_file(list_of_employees)

today_date = m.input_date()

m.output_over40_yo(list_of_employees, today_date)
m.output_over20_exp(list_of_employees, today_date)
m.delete_less1_exp(list_of_employees, today_date, name_of_file)

list_of_employees = m.read_file(name_of_file)
m.output_file(list_of_employees)
```

## module.py

```
import pickle

class Date:
    """Date"""

    def __init__(self):
        self.year = None
        self.month = None
        self.day = None

    def show_date(self):
        print(str(self.day) + '.' + str(self.month) + '.' + str(self.year),
end='')

    def set_date(self):
        while True:
            self.day = int(input("day:"))
            if(self.day>0 and self.day<32):
                break
        while True:
            self.month = int(input("month:"))
            if(self.month>0 and self.month<13):
                break
        while True:
            self.year = int(input("year:"))
            if(self.year>1900 and self.year<2023):
                break
        return self

class Employee:
    """Info about Employee"""

    def __init__(self):
        self.full_name = None
        self.birthday = Date()
        self.recruitment_date = Date()

    def show_info(self):
        print(self.full_name, end='\t')
        self.birthday.show_date()
```



```

        print(end='\t')
        self.recruitment_date.show_date()
        print()

def add_employee():
    employee = Employee()
    employee.full_name = input("Full Name:")
    print("Birthday:")
    employee.birthday.set_date()
    print("Date of recruitment")
    employee.recruitment_date.set_date()
    return employee

def create_file(file_name):
    print(
        "Please, select the write mode.\nTo create a new file, type 'w'.\nTo
        append to an already existing file, "
        "type 'a'.\n>>>")
    while True:
        write_mode = input()

print("=====")
        if write_mode == 'w':
            break
        elif write_mode == 'a':
            output_file(read_file(file_name))
            break
        else:
            print("Error. Please type 'w' or 'a'")

    try:
        with open(file_name, write_mode + 'b') as f_out:
            while True:
                mode = input("Do you want to add new employee?(Y/N)\n>>>")

print("=====")
                if mode == 'Y':
                    employee = add_employee()
                    pickle.dump(employee, f_out)
                elif mode == 'N':
                    break
            except FileNotFoundError:
                print("Error. File is not found.")

def read_file(file_name):
    try:
        with open(file_name, 'rb') as f_in:
            f_in.seek(0, 2)
            end_of_file = f_in.tell()
            f_in.seek(0, 0)
            employees = []
            while f_in.tell() != end_of_file:
                employee = pickle.load(f_in)
                employees.append(employee)
            except FileNotFoundError:
                print("Error. File is not found.")
            return employees

def output_file(employees):
    print("List of employees:")

```

```

for employee in employees:
    employee.show_info()
print("=====")

def input_date():
    print("Enter today's date")
    date = Date()
    date.set_date()
    return date

def output_over40_yo(employees, today):
    print("=====")
    print("Employees over 40 years old:")
    for employee in employees:
        flag = False
        if int(employee.birthday.year) < (int(today.year) - 40):
            flag = True
        elif int(employee.birthday.day) == (int(today.year) - 40):
            if int(employee.birthday.month) < int(today.month):
                flag = True
            elif int(employee.birthday.month) == int(today.month):
                if int(employee.birthday.day) < int(today.day):
                    flag = True
        if flag:
            employee.show_info()

def output_over20_exp(employees, today):
    print("=====")
    print("Employees over 20 years of experience:")
    for employee in employees:
        flag = False
        if int(employee.recruitment_date.year) < (int(today.year) - 20):
            flag = True
        elif int(employee.recruitment_date.day) == (int(today.year) - 20):
            if int(employee.recruitment_date.month) < int(today.month):
                flag = True
            elif int(employee.recruitment_date.month) == int(today.month):
                if int(employee.recruitment_date.day) < int(today.day):
                    flag = True
        if flag:
            employee.show_info()

def delete_less1_exp(employees, today, file_name):
    try:
        with open(file_name, 'wb') as f_out:
            for employee in employees:
                flag = False
                if int(employee.recruitment_date.year) < (int(today.year) -
1):
                    flag = True
                elif int(employee.recruitment_date.day) == (int(today.year) -
1):
                    if int(employee.recruitment_date.month) <
int(today.month):
                        flag = True
                    elif int(employee.recruitment_date.month) ==
int(today.month):
                        if int(employee.recruitment_date.day) <
int(today.day):
                            flag = True


```

```

        if flag:
            pickle.dump(employee, f_out)
    except FileNotFoundError:
        print("Error. File is not found.")
    print("=====")
    print("Employees with less than one year of experience were removed.")

```

## Виконання програми на C++:


Консоль отладки Microsoft Visual Studio

```

Enter file name:init.bin
=====
Please, select the write mode.
To create a new file, press 1.
To append to an already existing file, press 2.
>>>2
=====
List of employees:
Vartan Karamian      27.05.2022      12.12.2000
Suren Karamian      10.02.1972      12.12.2005
=====
Do you want to add new employee?(Y/N)
>>>Y
=====
Full name:Mariam Karamian
Birthday
day:12
month:12
year:2000
Date of recruitment
day:12
month:04
year:2022
=====
Do you want to add new employee?(Y/N)
>>>N
=====
List of employees:
Vartan Karamian      27.05.2022      12.12.2000
Suren Karamian      10.02.1972      12.12.2005
Mariam Karamian      12.12.2000      12.04.2022
=====
Enter today's date
day:21
month:04
year:2022
=====
Employees over 40 years old:
Suren Karamian      10.02.1972      12.12.2005
=====
Employees over 20 years of experience:
Vartan Karamian      27.05.2022      12.12.2000
=====
Employees with less than one year of experience were removed.
List of employees:
Vartan Karamian      27.05.2022      12.12.2000
Suren Karamian      10.02.1972      12.12.2005
=====

```

## На Python:

```
Run: main x
"D:\Projects Py\lab1.2 Py\venv\Scripts\python.exe" "D:/Projects Py/lab1.2 Py/main.py"
Enter file name:file.bin
Please, select the write mode.
To create a new file, type 'w'.
To append to an already existing file, type 'a'.
>>>a
=====
List of employees:
Vartan Karamian 27.5.2004    12.12.2019
=====
Do you want to add new employee?(Y/N)
>>>Y
=====
Full Name:Suren Karamian
Birthday:
day:10
month:10
year:1972
Date of recruitment
day:12
month:02
year:2022
Do you want to add new employee?(Y/N)
>>>N
=====
List of employees:
Vartan Karamian 27.5.2004    12.12.2019
Suren Karamian  10.10.1972   12.2.2022
=====
```

```
=====
Enter today's date
day:21
month:04
year:2022
=====
Employees over 40 years old:
Suren Karamian  10.10.1972   12.2.2022
=====
Employees over 20 years of experience:
=====
Employees with less than one year of experience were removed.
List of employees:
Vartan Karamian 27.5.2004    12.12.2019
=====

Process finished with exit code 0
```

on Control Run TODO Problems Terminal Python Packages Python Console

**Висновок:**

На цій лабораторній роботі я застосував на практиці знання щодо створення та обробки бінарних файлів даних на двох мовах програмування та побачив відмінності в їх реалізації.