Додаток 1

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 20

Виконала студентка ІП-13, Лисенко Анастасія Олегівна
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 1
Бінарні файли

Мета – вивчити особливості створення і обробки бінарних файлів даних.

Варіант 20

Постановка задачі.

Створити файл зі списком телепередач: назва, час початку та час закінчення передачі. Визначити тривалість кожної передачі. Створити новий файл, що містить передачі, які відображаються в денний час(09:00 до 18:00)

Код на мові C++:

myLib.cpp:

```
#include "myLib.h"

struct TeleProg {
	string nameOfProgram, text;
	int	startTimeHours, startTimeMinutes, endTimeHours, endTimeMinutes, lengthOfProgram;
	TeleProg(string line) {
		text = line + '\n';
		vector <string> parts = split(line, ' ');
		nameOfProgram = parts[0];
		startTimeHours = stoi(split(parts[1], ':')[0]);
		startTimeMinutes = stoi(split(parts[1], ':')[1]);
		endTimeHours = stoi(split(parts[2], ':')[0]);
		endTimeMinutes = stoi(split(parts[2], ':')[1]);
		lengthOfProgram = (endTimeHours*60 + endTimeMinutes) - (startTimeHours* 60 +
startTimeMinutes);
	}
};

vector <string> split(string line, char del)
{
	string str = "";
	vector <string> parts;
	line += del;
	for (int i = 0; i < line.length(); i++) {
		if (line[i] == del) {
			if (str.length() > 0) {
				parts.push_back(str);
			}
			str = "";
		}
		else {
			str += line[i];
		}
	}
	return parts;
}
string nameOfFile()
{
```

```
	string fileName;
	cout << "enter the name of the file:";
	cin >> fileName;
	return fileName;
}

bool isInBase(TeleProg prog, vector <TeleProg> base) {
	for (int i = 0; i < base.size(); i++) {
		if (base[i].nameOfProgram == prog.nameOfProgram) return true;
		else if ((base[i].endTimeHours*60 + base[i].endTimeMinutes > prog.endTimeHours*60 +
prog.endTimeMinutes) && (prog.endTimeHours * 60 + prog.endTimeMinutes > base[i].startTimeHours*60 +
base[i].startTimeMinutes ))
			return true;
		else if ((prog.startTimeHours*60 + prog.startTimeMinutes < base[i].startTimeHours *
60 + base[i].startTimeMinutes) && (prog.endTimeHours * 60 + prog.endTimeMinutes >
base[i].endTimeHours * 60 + base[i].endTimeMinutes))
			return true;
		else if ((base[i].startTimeHours * 60 + base[i].startTimeMinutes <
prog.startTimeHours * 60 + prog.startTimeMinutes) && (prog.startTimeHours * 60 +
prog.startTimeMinutes < base[i].endTimeHours * 60 + base[i].endTimeMinutes))
			return true;
	}
	return false;
}

void create(string name)
{
	char mode;
	ifstream fileif;
	vector <TeleProg> base;
	string text = "";
	cout << "Would you like to append your input? If so, enter a. Otherwise enter w:";
	cin >> mode;
	while (true) {
		if (mode == 'a'){
			fileif.open(name, ios::binary);
			char letter;
			while (fileif.read((char*)&letter, sizeof(char)))
			{
				if (letter == '\n') {
					base.push_back(TeleProg(text));
					text = "";
				}
				else text += letter;
			}
			fileif.close();
			break;
		}

		else if (mode == 'w')
			break;
		while (mode != 'w' && mode != 'a')
		{
			cout << "Enter correct letter:";
			cin >> mode;
		}
	}
	ofstream fileout(name, ios::binary);
	string line = "";
	cout << "Enter the input data as [name h:m h:m]\n To end the line | press ---> ENTER\n To
end the input | press ---> Ctrl + X\n";
	cin.ignore();
	getline(cin, line);
	while (line[0] != 24)
	{
		TeleProg program(line);
		if (base.size() == 0)
		{
```

```
                base.push_back(program);
                getline(cin, line);
                continue;
        }
        if (!isInBase(program, base))
        {
                base.push_back(program);
        }
        else
                cout << "Enter not repeating name or not overlapping time" << endl;
        getline(cin, line);
    }

    for (int i = 0; i < base.size(); i++) {
        fileout.write(base[i].text.c_str(), base[i].text.length());
    }

    fileout.close();
}

void output(string name)
{

    ifstream filein(name, ios::binary);
    char letter;
    while (filein.read((char*)&letter, sizeof(char)))
    {
        cout << letter;
    }

    cout << "_____" << endl;
    filein.close();
}

void new_list(string name)
{

    ifstream filein(name, ios::binary);
    char letter;
    string line = "";
    while (filein.read((char*)&letter, sizeof(char)))
    {
        if (letter == '\n') {
                TeleProg program(line);
                cout << "the name: " << program.nameOfProgram << "| " << "the length: " <<
program.lengthOfProgram/60 << " hours and " << program.lengthOfProgram%60 << " minutes" << endl;
                line = "";
        }
        else line += letter;

    }
    cout << "_____" << endl;
    filein.close();

}
```

```
void newfile(string name)
{
	ifstream filein(name, ios::binary);
	ofstream fileout("output.txt", ios::binary);
	vector <TeleProg> base;
	string line = "";
	char letter;
	while (filein.read((char*)&letter, sizeof(char)))
	{
		if (letter == '\n') {
			TeleProg program(line);
			if ((program.startTimeHours >= 9) && ((program.endTimeHours < 18) ||
(program.endTimeHours == 18 && program.endTimeMinutes == 0))){
				base.push_back(program);
			}
			line = "";
		}
		else line += letter;
	}
	for (int i = 0; i < base.size(); i++) {
		fileout.write(base[i].text.c_str(), base[i].text.length());
	}
	filein.close();
	fileout.close();


}
```

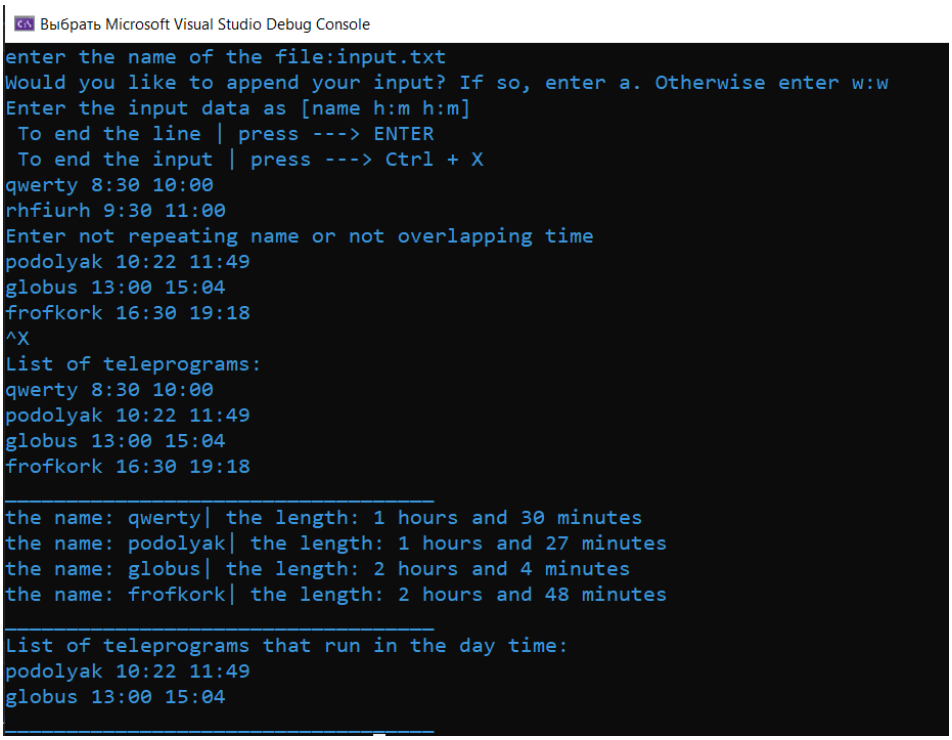
myLib.h:

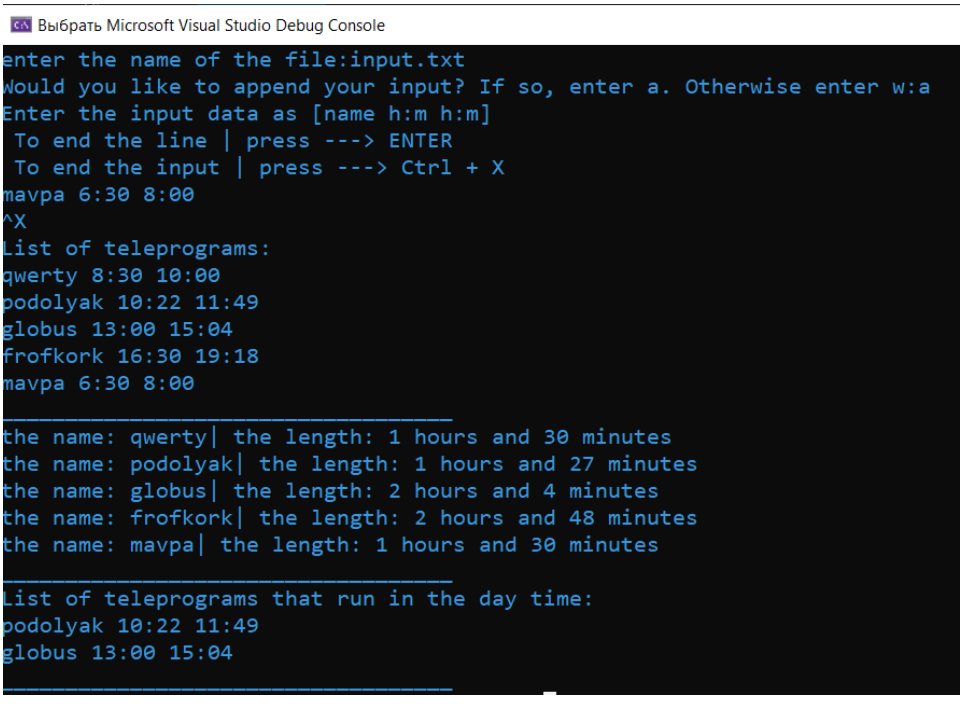
```
#pragma once
#include <iostream>
#include <fstream>
#include <string>
#include <iomanip>
#include <vector>;
using namespace std;

string nameOfFile();
vector <string> split(string, char);
void create(string );
void output(string );
void new_list(string );
void newfile(string );
```

lab1.cpp:

```
#include "myLib.h"

int main()
{
	string name = nameOfFile();
	create(name);
	cout << "List of teleprograms:\n";
	output(name);
	new_list(name);
	newfile(name);
	cout << "List of teleprograms that run in the day time:\n";
	output("output.txt");
	return 0;
}
```

Консоль:


Выбрать Microsoft Visual Studio Debug Console
enter the name of the file:input.txt
Would you like to append your input? If so, enter a. Otherwise enter w:w
Enter the input data as [name h:m h:m]
To end the line | press ---> ENTER
To end the input | press ---> Ctrl + X
qwerty 8:30 10:00
rhfiurh 9:30 11:00
Enter not repeating name or not overlapping time
podolyak 10:22 11:49
globus 13:00 15:04
frofkork 16:30 19:18
^X
List of teleprograms:
qwerty 8:30 10:00
podolyak 10:22 11:49
globus 13:00 15:04
frofkork 16:30 19:18
the name: qwerty| the length: 1 hours and 30 minutes
the name: podolyak| the length: 1 hours and 27 minutes
the name: globus| the length: 2 hours and 4 minutes
the name: frofkork| the length: 2 hours and 48 minutes
List of teleprograms that run in the day time:
podolyak 10:22 11:49
globus 13:00 15:04


Выбрать Microsoft Visual Studio Debug Console
enter the name of the file:input.txt
Would you like to append your input? If so, enter a. Otherwise enter w:a
Enter the input data as [name h:m h:m]
To end the line | press ---> ENTER
To end the input | press ---> Ctrl + X
mavpa 6:30 8:00
^X
List of teleprograms:
qwerty 8:30 10:00
podolyak 10:22 11:49
globus 13:00 15:04
frofkork 16:30 19:18
mavpa 6:30 8:00
the name: qwerty| the length: 1 hours and 30 minutes
the name: podolyak| the length: 1 hours and 27 minutes
the name: globus| the length: 2 hours and 4 minutes
the name: frofkork| the length: 2 hours and 48 minutes
the name: mavpa| the length: 1 hours and 30 minutes
List of teleprograms that run in the day time:
podolyak 10:22 11:49
globus 13:00 15:04

Код на мові Python:

methods.py:

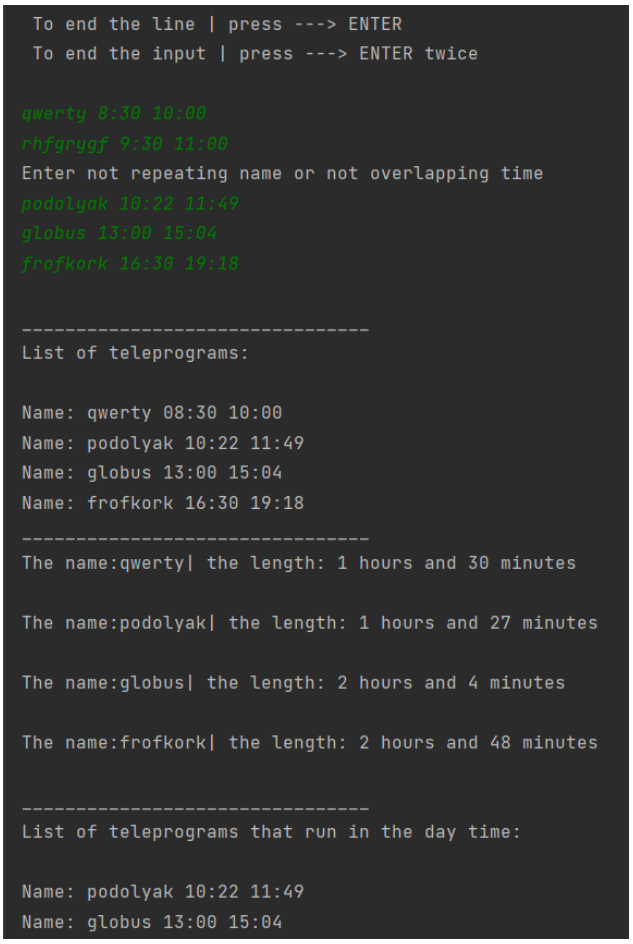
```
import pickle


def nameoffile():
    name = input("Enter the name of the file: ")
    return name


def isInBase(TeleProg: dict, base: list):
    for prog in base:
        startTime = TeleProg["startTimeHours"]*60 + TeleProg["startTimeMinutes"]
        startTime1 = prog["startTimeHours"]*60 + prog["startTimeMinutes"]
        endTime = TeleProg["endTimeHours"]*60 + TeleProg["endTimeMinutes"]
        endTime1 = prog["endTimeHours"]*60 + prog["endTimeMinutes"]
        if TeleProg["nameOfProgram"] == prog["nameOfProgram"]:
            return True
        elif startTime < startTime1 and endTime > endTime1:
            return True
        elif endTime1 > endTime > startTime1:
            return True
        elif startTime1 < startTime < endTime1:
            return True
        else:
            return False
    return False


def get_input(name):
    list = []
    mode = input("Would you like to append your input? If so, enter a. Otherwise enter
w:")
    while True:
        if mode == 'a':
            with open(name, "rb") as file:
                list = pickle.load(file)
            break
        if mode == 'w':
            with open(name, "wb") as file:
                file.truncate()
            break
        while mode != 'a' and mode != 'w':
            mode = input('Enter correct letter:')
    with open(name, "wb") as file:
        print("Enter the input data as [name h:m h:m]\n To end the line | press --->
ENTER\n To end the input | press ---> ENTER twice\n")
        line = input().split()
        while line:
            TeleProg = {
                "nameOfProgram": str(line[0]),
                "startTimeHours": int((line[1].split(':'))[0]),
                "startTimeMinutes": int((line[1].split(':'))[1]),
                "endTimeHours": int((line[2].split(':'))[0]),
                "endTimeMinutes": int((line[2].split(':'))[1])}
            if not isInBase(TeleProg, list):
                list.append(TeleProg)
            else:
                print("Enter not repeating name or not overlapping time")
            line = input().split()
        pickle.dump(list, file)
```

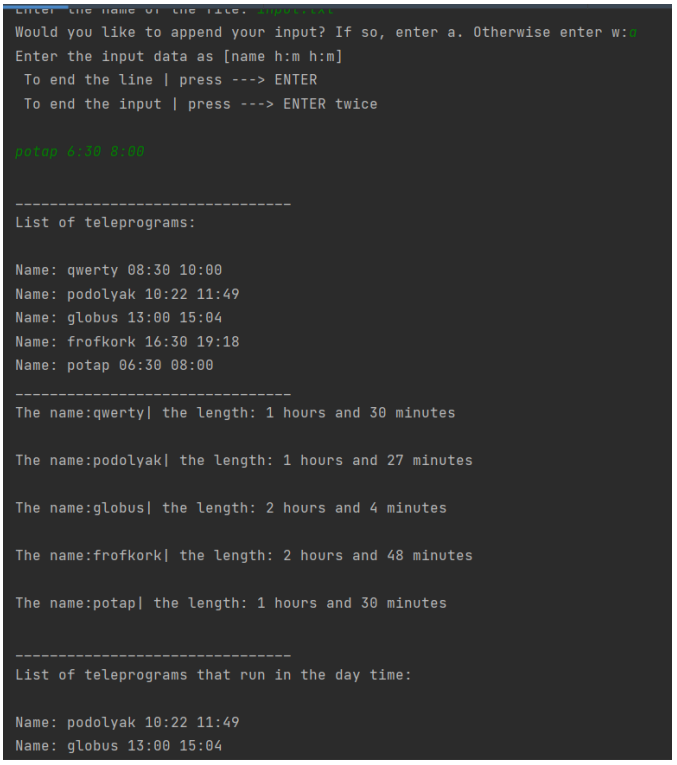
```
def lenghtOfProgram(name):
     with open(name, "rb") as fileif:
         temp = pickle.load(fileif)
         for i in range(len(temp)):
             startTime = temp[i]["startTimeHours"]*60 + temp[i]["startTimeMinutes"]
             endTime = temp[i]["endTimeHours"]*60 + temp[i]["endTimeMinutes"]
             length = endTime - startTime
             print("The name:" + temp[i]["nameOfProgram"] + "| the length: " +
str(int(length/60)) + " hours and " + str(length % 60) + " minutes\n")


def new_list(name):
    with open(name, "rb") as fileif:
        temp1 = pickle.load(fileif)
        str = []
        for i in range(len(temp1)):
            if temp1[i]["startTimeHours"] >= 9 and (temp1[i]["endTimeHours"] < 18 or
(temp1[i]["endTimeHours"] == 18 and temp1[i]["endTimeMinutes"] == 0)):
                str.append(temp1[i])
        with open("output.txt", "wb") as fileof:
            pickle.dump(str, fileof)


def get_format(TeleProg):
    str_startTimeHours = str(TeleProg["startTimeHours"])
    str_startTimeMinutes = str(TeleProg["startTimeMinutes"])
    str_endTimeHours = str(TeleProg["endTimeHours"])
    str_endTimeMinutes = str(TeleProg["endTimeMinutes"])
    if TeleProg["startTimeHours"] < 10:
        str_startTimeHours = '0' + str_startTimeHours
    if TeleProg["startTimeMinutes"] < 10:
        str_startTimeMinutes = '0' + str_startTimeMinutes
    if TeleProg["endTimeHours"] < 10:
        str_endTimeHours = '0' + str_endTimeHours
    if TeleProg["endTimeMinutes"] < 10:
        str_endTimeMinutes = '0' + str_endTimeMinutes
    return str_startTimeHours + ':' + str_startTimeMinutes + ' ' + str_endTimeHours +
':' + str_endTimeMinutes


def print_TeleProg(TeleProg):
    print("Name: " + TeleProg["nameOfProgram"] + " " + get_format(TeleProg))


def out_put(name):
    with open(name, 'rb') as file:
        temp = pickle.load(file)
        for i in range(len(temp)):
            print_TeleProg(temp[i])
```

main.py

```
import methods

name = methods.nameoffile()
methods.get_input(name)
print("_____")
print("List of teleprograms:\n")
methods.out_put(name)
print("_____")
methods.lenghtOfProgram(name)
print("_____")
print("List of teleprograms that run in the day time:\n")
methods.new_list(name)
methods.out_put("output.txt")
```

Консоль:





1. **Висновки**

Ми вивчили особливості створення і обробки бінарних файлів даних. Та навчилися створювати файли програмним шляхом, доповнювати їх, працювати з ними, читати та виводити їх.