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»

«Бінарні файли»

Варіант 5

Виконав студент ІП-15, Буяло Дмитро Олександрович
(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 й обробки бінарних файлів даних.

Індивідуальне завдання

Варіант 5

Завдання

Створити файл з інформацією про телефонні переговори: номер телефону, початок та кінець переговорів (за шаблоном - ГГ:ММ). З вихідного файлу створити два нових, в одному з яких мають бути зафіксовані денні переговори (з 6:00 до 20:00), а в іншому – нічні.

1. **Код на С++**

Lab_2_c++.cpp | Functions.cpp* | Functions.h

Lab_2_c++ | (Глобальная область)

```
1   #pragma once
2   #include <iostream>
3   #include <iomanip>
4   #include <string>
5   #include <fstream>
6   using namespace std;
7
8   struct TTalk {
9       char number[35];
10      char start[6], end[6];
11      int hs, ms;
12      int he, me;
13  };
14
15  bool validLetters(string);
16  void validSpaces(string&);
17  void validDuplicates(string&);
18  bool find(string);
19  string validNumbers(string);
20  void validTime(string, int&, int&);
21  string timeToString(int, int);
22  void setTime(int, int, int, int, TTalk&, int&);
23  bool tooLongCondition(int, int, int, int, int, int, int, int);
24  void writeText(int&, string, string);
25  TTalk* readFile(int, string);
26  void output(int, TTalk*);
27  void writeDayNightFile(string, string, TTalk*, int, int&, int&);
```

Lab_2_c++.cpp | Functions.cpp | Functions.h

Lab_2_c++ | (Глобальная область) | main()

```cpp
#include "Functions.h"

int main()
{
    setlocale(LC_ALL, "russian");
    /* ... */
    cout << "Hello World!\n";
    string fileName = "MainFile.dat";
    string filenameDay = "Day.dat";
    string filenameNight = "Night.dat";
    int counter = 0;
    int dayCounter, nightCounter;
    string again;
    do {
        writeText(counter, fileName, again);

        TTalk* docText = readFile(counter, fileName);
        cout << "\nFrom Main binary file:" << endl;
        output(counter, docText);

        dayCounter = 0, nightCounter = 0;
        writeDayNightFile(filenameDay, filenameNight, docText, counter, dayCounter, nightCounter);

        TTalk* docDay = readFile(dayCounter, filenameDay);
        cout << "\nFrom Day binary file:" << endl;
        output(dayCounter, docDay);

        TTalk* docNight = readFile(nightCounter, filenameNight);
        cout << "\nFrom Night binary file:" << endl;
        output(nightCounter, docNight);

        cout << "\n\nDo you want to add something? [Y/N]: "; cin >> again; cin.ignore();
    } while (again[0] == 'y' || again[0] == 'Y' || again[0] == '1' || again[0] == '+');
}
```

Lab_2_c++.cpp | Functions.cpp | Functions.h

Lab_2_c++ | (Глобальная область)

```cpp
#include "Functions.h"

void writeText(int& counter, string fileName, string again) {
    TTalk b;
    string yn = "y";
    string number, start, end;
    ofstream outFile;
    if (again == "") outFile.open(fileName, ios::binary); // случай, если нужно дозаписать файл
    else outFile.open(fileName, ios::binary | ios::app);
    while (yn[0] == 'y' || yn[0] == 'Y' || yn[0] == '1' || yn[0] == '+') {
        cout << "Number: "; getline(cin, number);
        cout << "Start: "; getline(cin, start);
        cout << "End: "; getline(cin, end);
        if (validLetters(number) && validLetters(start) && validLetters(end)) {

            validSpaces(number); validSpaces(start); validSpaces(end);
            validDuplicates(number); validDuplicates(start); validDuplicates(end);

            if (find(start) && find(end)) {
                validTime(start, b.hs, b.ms);
                validTime(end, b.he, b.me);
                if (b.hs > 23 || b.ms > 59 || b.he > 23 || b.me > 59) {
                    cout << "Какое-то странное у вас время!)" << endl;
                }
                if (number.length() > 33) {
                    cout << "Я проверил все номера мира, при любом вводе. Таких длинных не существует!" << endl;
                }
                else {
                    strcpy_s(b.number, number.c_str());
                    int tmpH, tmpM; //temporary hours/minutes
                    int dBreakH = 6, dBreakM = 0;  // Day break hours/minutes
                    int nBreakH = 20, nBreakM = 0; // Night break hours/minutes

                    // рассматриваем на брейках, чтобы не мешали последующим условиям
                    if (b.hs < dBreakH && b.he == nBreakH && b.me == nBreakM) {
                        setTime(b.hs, b.ms, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(dBreakH, dBreakM, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    else if (b.hs < dBreakH && b.he == dBreakH && b.me == dBreakM) {
                        setTime(b.hs, b.ms, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    else if (b.he == b.hs && b.me == b.ms) {
                        setTime(b.hs, b.ms, b.he, b.me, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    else if (b.hs < nBreakH && b.he == dBreakH && b.me == dBreakM) {
                        setTime(b.hs, b.ms, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(nBreakH, nBreakM, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    else if (b.hs < nBreakH && b.he == nBreakH && b.me == nBreakM) {
                        setTime(b.hs, b.ms, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    else if (b.he < dBreakH && b.hs == dBreakH && b.ms == dBreakM) {
                        tmpH = b.he; tmpM = b.me;
                        setTime(b.hs, b.ms, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(nBreakH, nBreakM, tmpH, tmpM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }

                    // проход через два брейка
                    else if (tooLongCondition(b.hs, b.he, b.ms, b.me, dBreakH, dBreakM, nBreakH, nBreakM)) {
                        tmpH = b.he; tmpM = b.me;
                        setTime(b.hs, b.ms, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(dBreakH, dBreakM, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(nBreakH, nBreakM, tmpH, tmpM, b, counter);
```

Lab_2_c++.cpp | Functions.cpp* | Functions.h

Lab_2_c++ | (Глобальная область) | writeText(int & counter, string fileName, string again)

```
                        setTime(nBreakH, nBreakM, tmpH, tmpM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    // полный оборот часов между 6:00 и 20:00
                    else if (b.he >= dBreakH && b.hs < nBreakH && (b.hs > b.he || (b.hs == b.he && b.ms > b.me))) {
                        tmpH = b.he; tmpM = b.me;
                        setTime(b.hs, b.ms, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(nBreakH, nBreakM, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(dBreakH, dBreakM, tmpH, tmpM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }

                    // проход через один брейк
                        // дневной брейк [0:00; 6:00) и [6:00; 20:00)
                    else if (b.hs < dBreakH && b.he >= dBreakH) {
                        tmpH = b.he; tmpM = b.me;
                        setTime(b.hs, b.ms, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(dBreakH, dBreakM, tmpH, tmpM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    // ночной брейк [0:00; 20:00) и [20:00; 24:00);(или) полный оборот часов между 6:00 и 20:00
                    else if (b.hs < nBreakH && b.he >= nBreakH || b.hs >= dBreakH && b.hs < nBreakH && b.he < dBreakH)
                        tmpH = b.he; tmpM = b.me;
                        setTime(b.hs, b.ms, nBreakH, nBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(nBreakH, nBreakM, tmpH, tmpM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                    // полный оборот с 20:00 до 6:00-20:00
                    else if (b.hs > nBreakH && b.he > dBreakH && b.he < nBreakH) {
                        tmpH = b.he; tmpM = b.me;
                        setTime(b.hs, b.ms, dBreakH, dBreakM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));

                        setTime(dBreakH, dBreakM, tmpH, tmpM, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }

                    else {
                        setTime(b.hs, b.ms, b.he, b.me, b, counter);
                        outFile.write((char*)&b, sizeof(TTalk));
                    }
                }
            }
            else cout << "Время точно введено по шаблону?" << endl;
        }
                }
            }
            else cout << "Время точно введено по шаблону?" << endl;
        }
        else cout << "Время/номер телефона не должны содержать букв)" << endl;

        cout << "\nContinue? [Y/N]: "; cin >> yn; cin.ignore();
    }
    outFile.close();
}
```

Lab_2_c++.cpp | Functions.cpp* | Functions.h

Lab_2_c++ | (Глобальная область)

```cpp
bool validLetters(string data) { // если есть буквы, то не подходит
    for (int i = 0; i < data.length(); i++) {
        if (isalpha(data[i])) return false;
    }
    return true;
}

void validSpaces(string& data) { // убираем все пробелы
    for (int i = 0; i < data.length(); i++) {
        if (isspace(data[i])) {
            data.erase(i, 1);
            i--;
        }
    }
}

void validDuplicates(string& data) { // убираем лишние повторяющиеся символы
    for (int i = 1; i < data.length(); i++) {
        if (!isdigit(data[i - 1]) && !isdigit(data[i])) {
            data.erase(i, 1);
            i--;
        }
    }
}
```

```cpp
bool find(string line) { // проверяем, есть ли ':' во времени
    if (line.length() < 3) return false;
    for (int i = 1; i < line.length() - 1; i++) {// не первый и не последний символ
        if (line[i] == ':') return true;
    }
    return false;
}

void validTime(string time, int& hours, int& minutes) { // разделяем на часы и минуты
    string h, m;
    int ind = time.find(':');
    h = validNumbers(time.substr(0, ind));
    m = validNumbers(time.substr(ind + 1));
    hours = stoi(h);
    minutes = stoi(m);
}

string validNumbers(string time) { // убираем все лишнее
    for (int i = 0; i < time.length(); i++) {
        if (!isdigit(time[i])) {
            time.erase(i, 1);
            i--;
        }
    }
    return time;
}
```

Lab_2_c++.cpp | Functions.cpp* | Functions.h

Lab_2_c++ | (Глобальная область) | writeText(int & counter, string fileName, string again)

```cpp
void setTime(int hs, int ms, int he, int me, TTalk& b, int& counter) { // сохраняем отвалидированные данные
    string start, end;
    start = timeToString(hs, ms);
    strcpy_s(b.start, start.c_str());
    end = timeToString(he, me);
    strcpy_s(b.end, end.c_str());
    b.hs = hs; b.ms = ms; b.he = he; b.me = me;
    cout << b.number << ' ' << b.start << ' ' << b.end << endl;
    counter++;
}

string timeToString(int hours, int minutes) { // превращаем в правильную строку
    string time;
    if (minutes < 10) time = to_string(hours) + ":0" + to_string(minutes);
    else time = to_string(hours) + ":" + to_string(minutes);
    return time;
}

// сборник условий с одинаковым действием
bool tooLongCondition(int hs, int he, int ms, int me, int dBreakH, int dBreakM, int nBreakH, int nBreakM) {
    if (hs < dBreakH && he >= nBreakH) return true; // до 6:00 и после 20:00
    else if (hs < dBreakH && (hs > he || hs == he && ms > me)) return true; // полный оборот часов до 6:00
    else if (hs == dBreakH && ms == dBreakM && hs > he) return true;
    else if (he >= nBreakH && (hs > he || hs == he && ms > me)) return true; // полный оборот часов после 20:00
    else return false;
}

TTalk* readFile(int counter, string fileName) { // читаем файл в массив
    TTalk b;
    int i = 0;
    cout << "counter = " << counter << endl;//!ToDo прибрать
    ifstream inFile(fileName, ios::binary);
    TTalk* docText = new TTalk[counter];

    while (inFile.read((char*)&b, sizeof(TTalk))) {
        docText[i] = b;
        i++;
    }

    inFile.close();
    return docText;
}


void output(int counter, TTalk* arr) { // выводим содержание файла, который записали в массив
    for (int i = 0; i < counter; i++) {
        cout << arr[i].number << " " << arr[i].start << " " << arr[i].end << endl;
    }
}
```

```
Lab_2_c++.cpp | Functions.cpp* | Functions.h
Lab_2_c++ | (Глобальная область) | writeDayNightFile(string fileD, string fileN, TTalk * arr, int cc
243  // по условию создаем два файла с дневными/ночными разговорами
244  void writeDayNightFile(string fileD, string fileN, TTalk* arr, int counter, int& countD, int& countN) {
245      int dBreakH = 6, dBreakM = 0;
246      int nBreakH = 20, nBreakM = 0;
247      ofstream outFileDay(fileD, ios::binary);
248      ofstream outFileNight(fileN, ios::binary);
249      for (int i = 0; i < counter; i++) {
250          // проверка на дневной файл
251          if (arr[i].hs >= dBreakH && (arr[i].he < nBreakH || arr[i].he == nBreakH && arr[i].me == nBreakM)) {
252              // пришлось разделить условие, чтобы оно не было слишком длинным
253              if (arr[i].he > arr[i].hs || arr[i].he == arr[i].hs && arr[i].me >= arr[i].ms) {
254                  outFileDay.write((char*)&arr[i], sizeof(TTalk));
255                  countD++;
256              }
257              else {
258                  outFileNight.write((char*)&arr[i], sizeof(TTalk));
259                  countN++;
260              }
261          }
262          else {
263              outFileNight.write((char*)&arr[i], sizeof(TTalk));
264              countN++;
265          }
266      }
267      outFileDay.close();
268      outFileNight.close();
269  }
```

2. Код на Python

```
main.py | functions.py
1   import functions as f
2
3   fileName = "MainFile.dat"
4   filenameDay = "Day.dat"
5   filenameNight = "Night.dat"
6   again = 'y'
7   ink = 1
8   counter = 0
9   while again[0] == 'y' or again[0] == '1' or again[0] == '+':
10      if ink == 1:
11          again = ''
12
13      counter = f.writeText(fileName, again, counter)
14      docText = f.readFile(fileName, counter)
15      print("\nFrom Main binary file:")
16      f.output(docText, counter)
17
18      dayCounter, nightCounter = f.writeDayNightFile(filenameDay, filenameNight, docText, counter)
19      docDay = f.readFile(filenameDay, dayCounter)
20      print("\nFrom Day binary file:")
21      f.output(docDay, dayCounter)
22      docNight = f.readFile(filenameNight, nightCounter)
23      print("\nFrom Night binary file:")
24      f.output(docNight, nightCounter)
25      again = input('\nDo you want to add something to the file? [Y/N]: ').lower()
26      ink = 0
```

main.py functions.py

```python
import pickle


def writeText(name, a, count):
    yn = "y"
    if a == '':  # случай, если нужно дозаписать файл
        outFile = open(name, 'wb')
    else:
        outFile = open(name, 'ab')
    while yn[0] == 'y' or yn[0] == '1' or yn[0] == '+':
        talkT = {
            "number": getNumber(),
            "start": getTime(),
            "end": getTime()
        }
        setStart(talkT, talkT["start"])
        setEnd(talkT, talkT["end"])

        # разбиение на брейках + полный оборот часов
        if talkT["hs"] < 6 and talkT["he"] == 20 and talkT["me"] == 0:
            setTime(talkT["start"], "6:00", talkT)
            pickle.dump(talkT, outFile)
            count += 1

            setTime("6:00", "20:00", talkT)
            pickle.dump(talkT, outFile)
            count += 1
        elif talkT["hs"] < 6 and talkT["he"] == 6 and talkT["me"] == 0:
            setTime(talkT["start"], talkT["end"], talkT)
            pickle.dump(talkT, outFile)
            count += 1
        elif talkT["hs"] == talkT["he"] and talkT["me"] == talkT["ms"]:
            setTime(talkT["start"], talkT["end"], talkT)
            pickle.dump(talkT, outFile)
            count += 1
        elif talkT["hs"] < 20 and talkT["he"] == 6 and talkT["me"] == 0:
            setTime(talkT["start"], "20:00", talkT)
            pickle.dump(talkT, outFile)
            count += 1

            setTime("20:00", "6:00", talkT)
            pickle.dump(talkT, outFile)
            count += 1
        elif talkT["hs"] < 20 and talkT["he"] == 20 and talkT["me"] == 0:
            setTime(talkT["start"], talkT["end"], talkT)
            pickle.dump(talkT, outFile)
            count += 1
```

```
    elif talkT["he"] < 6 and talkT["hs"] == 6 and talkT["ms"] == 0:
        tmpE = talkT["end"]
        setTime(talkT["start"], "20:00", talkT)
        pickle.dump(talkT, outFile)
        count += 1

        setTime("20:00", tmpE, talkT)
        pickle.dump(talkT, outFile)
        count += 1
    elif longCondition(talkT["hs"], talkT["he"], talkT["ms"], talkT["me"]):
        tmpE = talkT["end"]
        setTime(talkT["start"], "6:00", talkT)
        pickle.dump(talkT, outFile)
        count += 1

        setTime("6:00", "20:00", talkT)
        pickle.dump(talkT, outFile)
        count += 1

        setTime("20:00", tmpE, talkT)
        pickle.dump(talkT, outFile)
        count += 1
    elif talkT["he"] >= 6 and talkT["hs"] < 20 and (
            talkT["hs"] > talkT["he"] or talkT["hs"] == talkT["he"] and talkT["ms"] > talkT["me"]):
        tmpE = talkT["end"]
        setTime(talkT["start"], "20:00", talkT)
        pickle.dump(talkT, outFile)
        count += 1

        setTime("20:00", "6:00", talkT)
        pickle.dump(talkT, outFile)
        count += 1

        setTime("6:00", tmpE, talkT)
        pickle.dump(talkT, outFile)
        count += 1
    elif talkT["hs"] < 6 <= talkT["he"]:
        tmpE = talkT["end"]
        setTime(talkT["start"], "6:00", talkT)
        pickle.dump(talkT, outFile)
        count += 1

        setTime("6:00", tmpE, talkT)
        pickle.dump(talkT, outFile)
        count += 1
```

main.py × functions.py ×

```
93          elif talkT["he"] >= 20 > talkT["hs"] or 20 > talkT["hs"] >= 6 > talkT["he"]:
94              tmpE = talkT["end"]
95              setTime(talkT["start"], "20:00", talkT)
96              pickle.dump(talkT, outFile)
97              count += 1
98
99              setTime("20:00", tmpE, talkT)
100             pickle.dump(talkT, outFile)
101             count += 1
102         elif talkT["hs"] > 20 > talkT["he"] > 6:
103             tmpE = talkT["end"]
104             setTime(talkT["start"], "6:00", talkT)
105             pickle.dump(talkT, outFile)
106             count += 1
107
108             setTime("6:00", tmpE, talkT)
109             pickle.dump(talkT, outFile)
110             count += 1
111         else:
112             setTime(talkT["start"], talkT["end"], talkT)
113             pickle.dump(talkT, outFile)
114             count += 1
115
116         yn = input('\nContinue? [Y/N]: ').lower()
117     outFile.close()
118     return count
```

```
121 def getNumber():  # ввод номера
122     number = input("Number: ")
123     number = validDuplicates(number)
124     if len(number) < 1 or len(number) > 33 or validLetters(number):  # проверка на валидность
125         print('This number does not exist, please, try again')
126         number = getNumber()
127         return number
128     return number
129
130
131 def validDuplicates(data):  # убираем дубликаты, которые не являются цифрами
132     ind = 1
133     for i in range(1, len(data)):
134         if not data[ind - 1].isdigit() and not data[ind].isdigit():
135             data = data[:ind] + data[(ind + 1):]
136             ind -= 1  # со счетчиком фор в питоне особо не поработаешь,
137         ind += 1  # поэтому выкручиваемся как можем
138     return data
139
140
141 def validLetters(data):  # проверка на наличие букв
142     for i in data:
143         if i.isalpha():
144             return True
145     return False
```

main.py functions.py

```python
def getTime():  # ввод времени начала/конца
    time = input("Start/End: ")
    time = validSpaces(time)
    time = validDuplicates(time)
    if len(time) < 3 or len(time) > 5 or validLetters(time) or time.find(':') == -1:  # проверка на правильный ввод
        print('Time entered incorrectly, please, try again')
        time = getTime()
        return time
    if time.find(':') == 0 or time.find(':') == len(time) - 1:
        print('Time entered incorrectly, please, try again')
        time = getTime()
        return time
    hours = time[:time.find(':')]
    minutes = time[time.find(':') + 1:]
    hours = validTime(hours)
    minutes = validTime(minutes)

    if int(hours) > 23 or int(minutes) > 59:
        print('Time entered incorrectly, please, try again')
        time = getTime()
        return time

    if int(minutes) < 10 and minutes[0] != '0' and len(minutes) < 2:  # делаем "красивое" время
        time = hours + ':0' + minutes
    else:
        time = hours + ':' + minutes

    return time


def validSpaces(data):  # убираем все пробелы
    ind = 1
    for i in range(1, len(data)):
        if data[ind].isspace():
            data = data[:ind] + data[(ind + 1):]
            ind -= 1
        ind += 1
    return data


def validTime(time):  # оставляем только цифры
    tmpT = "".join(c for c in time if c.isdecimal())
    return tmpT


def setStart(struct, time):  # записываем часы и минуты начала
    struct["hs"] = getHours(time)
    struct["ms"] = getMinutes(time)
```

main.py functions.py

```
198 def setEnd(struct, time):  # записываем часы и минуты конца
199     struct["he"] = getHours(time)
200     struct["me"] = getMinutes(time)
201
202
203 def getHours(time):  # получаем часы
204     hours = int(validTime(time[:time.find(':')]))
205     return hours
206
207
208 def getMinutes(time):  # получаем минуты
209     minutes = int(validTime(time[time.find(':') + 1:]))
210     return minutes
211
212
213 def setTime(start, end, struct):  # устанавливаем время
214     struct["start"] = start
215     struct["end"] = end
216     setStart(struct, start)
217     setEnd(struct, end)
218     print(struct["number"], struct["start"], struct["end"])
219
221 def longCondition(hs, he, ms, me):  # сборник условий
222     if hs < 6 and he >= 20:
223         return True
224     elif hs < 6 and (hs > he or hs == he and ms > me):
225         return True
226     elif hs == 6 and ms == 0 and hs > he:
227         return True
228     elif he >= 20 and (hs > he or hs == he and ms > me):
229         return True
230     else:
231         return False
232
233
234 def readFile(name, count):  # читаем файл в массив
235     file = open(name, 'rb')
236     doc = []
237     for i in range(0, count):
238         doc.append(pickle.load(file))
239     return doc
240
241
242 def output(arr, count):  # выводим содержание файла, который записали в массив
243     for i in range(0, count):
244         print(arr[i]["number"] + " " + arr[i]["start"] + " " + arr[i]["end"])
245
```

main.py functions.py

```
# по условию создаем два файла с дневными/ночными разговорами
def writeDayNightFile(name_day, name_night, doc, count):
    outFileDay = open(name_day, 'wb')
    outFileNight = open(name_night, 'wb')
    countD = 0
    countN = 0
    for i in range(0, count):
        if doc[i]["he"] >= 6 and (doc[i]["he"] < 20 or doc[i]["he"] == 20 and doc[i]["me"] == 0):
            if doc[i]["he"] > doc[i]["hs"] or doc[i]["he"] == doc[i]["hs"] and doc[i]["me"] >= doc[i]["ms"]:
                pickle.dump(doc[i], outFileDay)
                countD += 1
            else:
                pickle.dump(doc[i], outFileNight)
                countN += 1
        else:
            pickle.dump(doc[i], outFileNight)
            countN += 1
    outFileDay.close()
    outFileNight.close()
    return countD, countN

```

3. Результат виконання на C++

```
Number: +38(050)48-1      5----489
Start: 11:25
End: 17:40
+38(050)48-15-489 11:25 17:40

Continue? [Y/N]: yes
Number: +38(095)1321321231231321231231231231231321321231 32
Start: 11:10
End: 15:30
Я проверил все номера мира, при любом вводе. Таких длинных не существует!

Continue? [Y/N]: Y
Number: +38(095)55-55-555
Start: 11:
End: :15
Время точно введено по шаблону?

Continue? [Y/N]: Yes
Number: +38(050)4215965
Start: 3:47
End: 7:41
+38(050)4215965 3:47 6:00
+38(050)4215965 6:00 7:41

Continue? [Y/N]: 0
counter = 3

From Main binary file:
+38(050)48-15-489 11:25 17:40
+38(050)4215965 3:47 6:00
+38(050)4215965 6:00 7:41
counter = 2

From Day binary file:
+38(050)48-15-489 11:25 17:40
+38(050)4215965 6:00 7:41
counter = 1

From Night binary file:
+38(050)4215965 3:47 6:00
```

```
Do you want to add something? [Y/N]: yes
Number: 380504125412
Start: 24:39
End: 12:68
Какое-то странное у вас время!)

Continue? [Y/N]: 1
Number: +38(050)1234567
Start: 15:47
End: 4:12
+38(050)1234567 15:47 20:00
+38(050)1234567 20:00 4:12

Continue? [Y/N]: 0
counter = 5

From Main binary file:
+38(050)48-15-489 11:25 17:40
+38(050)4215965 3:47 6:00
+38(050)4215965 6:00 7:41
+38(050)1234567 15:47 20:00
+38(050)1234567 20:00 4:12
counter = 3

From Day binary file:
+38(050)48-15-489 11:25 17:40
+38(050)4215965 6:00 7:41
+38(050)1234567 15:47 20:00
counter = 2

From Night binary file:
+38(050)4215965 3:47 6:00
+38(050)1234567 20:00 4:12


Do you want to add something? [Y/N]: 1
Number: +38(095)4714456
Start: 16:34
End: 10:55
+38(095)4714456 16:34 20:00
+38(095)4714456 20:00 6:00
+38(095)4714456 6:00 10:55
```

```
Continue? [Y/N]: 0
counter = 8

From Main binary file:
+38(050)48-15-489 11:25 17:40
+38(050)4215965 3:47 6:00
+38(050)4215965 6:00 7:41
+38(050)1234567 15:47 20:00
+38(050)1234567 20:00 4:12
+38(095)4714456 16:34 20:00
+38(095)4714456 20:00 6:00
+38(095)4714456 6:00 10:55
counter = 5

From Day binary file:
+38(050)48-15-489 11:25 17:40
+38(050)4215965 6:00 7:41
+38(050)1234567 15:47 20:00
+38(095)4714456 16:34 20:00
+38(095)4714456 6:00 10:55
counter = 3

From Night binary file:
+38(050)4215965 3:47 6:00
+38(050)1234567 20:00 4:12
+38(095)4714456 20:00 6:00


Do you want to add something? [Y/N]: 0

C:\Users\Пользователь\OneDrive\Документы\
Чтобы автоматически закрывать консоль при
```

4. Результат виконання на Python

```
Number: +38(050) KK 4871254
Start/End: 11:34
Start/End: 20:48
+38(050)4871254 11:34 20:00
+38(050)4871254 20:00 20:48

Continue? [Y/N]: yes
Number: +38(050)18P48-125
This number does not exist, please, try again
Number: +38(095)123123132132123123123123123123213123132132132
This number does not exist, please, try again
Number: +38(095)-----87------45        475
Start/End: 15:40
Start/End: 24:30
Time entered incorrectly, please, try again
Start/End: 20:60
Time entered incorrectly, please, try again
Start/End: 18:31
+38(095)87-45 475 15:40 18:31

Continue? [Y/N]: y
Number: +38(050)4714444
Start/End: 1:1
Start/End: 8:09
+38(050)4714444 1:01 6:00
+38(050)4714444 6:00 8:09
```

```
Continue? [Y/N]: Yeah
Number: +38(050)44-82-46
Start/End: 12:111
Time entered incorrectly, please, try again
Start/End: 150:45
Time entered incorrectly, please, try again
Start/End: 21:34
Start/End: 5:39
+38(050)44-82-46 21:34 5:39

Continue? [Y/N]: no

From Main binary file:
+38(050)4871254 11:34 20:00
+38(050)4871254 20:00 20:48
+38(095)87-45 475 15:40 18:31
+38(050)4714444 1:01 6:00
+38(050)4714444 6:00 8:09
+38(050)44-82-46 21:34 5:39

From Day binary file:
+38(050)4871254 11:34 20:00
+38(095)87-45 475 15:40 18:31
+38(050)4714444 1:01 6:00
+38(050)4714444 6:00 8:09
```

```
From Night binary file:
+38(050)4871254 20:00 20:48
+38(050)44-82-46 21:34 5:39

Do you want to add something to the file? [Y/N]: yes
Number: +38(050)9900112
Start/End: 15:48
Start/End: 12:34
+38(050)9900112 15:48 20:00
+38(050)9900112 20:00 6:00
+38(050)9900112 6:00 12:34

Continue? [Y/N]: 1
Number: 38(095)4567890
Start/End: 6:00
Start/End: 20:0
38(095)4567890 6:00 20:0

Continue? [Y/N]: 0

From Main binary file:
+38(050)4871254 11:34 20:00
+38(050)4871254 20:00 20:48
+38(095)87-45 475 15:40 18:31
+38(050)4714444 1:01 6:00
+38(050)4714444 6:00 8:09
```

```
+38(050)4714444 6:00 8:09
+38(050)44-82-46 21:34 5:39
+38(050)9900112 15:48 20:00
+38(050)9900112 20:00 6:00
+38(050)9900112 6:00 12:34
38(095)4567890 6:00 20:0

From Day binary file:
+38(050)4871254 11:34 20:00
+38(095)87-45 475 15:40 18:31
+38(050)4714444 1:01 6:00
+38(050)4714444 6:00 8:09
+38(050)9900112 15:48 20:00
+38(050)9900112 6:00 12:34
38(095)4567890 6:00 20:0

From Night binary file:
+38(050)4871254 20:00 20:48
+38(050)44-82-46 21:34 5:39
+38(050)9900112 20:00 6:00

Do you want to add something to the file? [Y/N]: 0

Process finished with exit code 0
```