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»
«Бінарні файли»

Варіант 34

Виконав студент ІП-15, Чінь Хоанг Вьет
Перевірила Вечерковська Анастасія Сергіївна

Київ 2022

Лабораторна робота №2

Варіант 34

Задача

34. Створити файл з інформацією про студентів: ПІБ, дата народження, форма навчання, група, середній рейтинг успішності за останню сесію. На кожному потоці (групи потоку мають однакову абревіатуру) визначити студентів з мінімальним середнім балом успішності і групи, в яких вони навчаються. В новому файлі сформувати відсортований за прізвищами список студентів-четверокурсників денної форми навчання, середній бал успішності яких не менше за вказаний.

Код:

1. C#

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab_2_cs
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            //Ім'я бінарних файлів
            string BinaryFileName = "BinaryFile.txt";
            string newBinaryFile = "newBinaryFile.txt";

            //Список студентів
            create_info create_Info = new create_info();
            List<info> list_of_students = create_Info.information();
            Console.WriteLine("\nAll of the students:");
            create_Info.output_list(list_of_students);

            //Передача інформації в бінарний файл
            int AmountOfStudents = create_Info.CreateBinaryFile(list_of_students, BinaryFileName);
```

```
            //Студенти із найнижчими балами
            List<info> Lowest = create_Info.LowLevelStudents(BinaryFileName, AmountOfStudents);
            Console.WriteLine("Lowest mark students:");
            create_Info.output_list(Lowest);

            //Відсортований список студентів 4-го курсу
            List<info> newList = create_Info.Identify(Lowest);
            Console.WriteLine("4-th grade students:");
            create_Info.output_list(newList);
            int Stud = create_Info.CreateBinaryFile(newList, newBinaryFile);
            Console.ReadLine();
        }
    }
}
```

Class1.cs

```cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;

namespace Lab_2_cs
{
    18 references
    public struct info
    {
        public string name,
                      FormOfStudying,
                      GroupName,
                      DateOfBirth;
        public int GroupNumber;
        public double AverageMark;
    };
    2 references
    public class create_info
    {
        1 reference
        public List<info> information()
        {
            char stop_button = (char)19;
            List<info> list = new List<info>();
            info student_info;
            Console.Write("Enter the name of the group: ");
            string groupName = Console.ReadLine();
            string additional;
            while (true)
            {
                Console.Write("Student's name: ");
                additional = Console.ReadLine();
                if (additional[0] != stop_button)
                {
                    student_info.name = additional;

                    Console.Write("Date of Birth(DD.MM.YEAR): ");
                    student_info.DateOfBirth = Console.ReadLine();

                    Console.Write("Form of studying(Daily or Correspondence): ");
                    student_info.FormOfStudying = Console.ReadLine();

                    Console.Write("Group number: ");
                    student_info.GroupNumber = Convert.ToInt32(Console.ReadLine());

                    Console.Write("Average mark(0-100): ");
                    student_info.AverageMark = Convert.ToDouble(Console.ReadLine());

                    Console.WriteLine();
                    student_info.GroupName = groupName;
                    list.Add(student_info);
                }
                else
                {
                    break;
                }
            }
            return list;
        }
        3 references
        public void output_list(List<info> list)
        {
            Console.WriteLine("-----------------------------------------------------------------------------------------------");
            Console.WriteLine("          Student's name           | Date of Birth | Form of studying | Group | Average Mark ");
            Console.WriteLine("-----------------------------------------------------------------------------------------------");
            for (int i = 0; i < list.Count; i++)
            {
                Console.WriteLine(String.Format(" {0,-33} | {1,-13} | {2,-16} | {3}-{4,-5} | {5,-12} ",
                        list[i].name, list[i].DateOfBirth, list[i].FormOfStudying,
                        list[i].GroupName, list[i].GroupNumber, list[i].AverageMark));
            }
            Console.WriteLine("-----------------------------------------------------------------------------------------------");
        }
        1 reference
        public int CreateBinaryFile(List<info> list, string fileName)
        {
            BinaryWriter BinaryFile;
            BinaryFile = new BinaryWriter(new FileStream(fileName, FileMode.OpenOrCreate, FileAccess.Write));
            for (int i = 0; i < list.Count; i++)
            {
```

```
79                  BinaryFile.Write(list[i].name);
80                  BinaryFile.Write(list[i].DateOfBirth);
81                  BinaryFile.Write(list[i].FormOfStudying);
82                  BinaryFile.Write(list[i].GroupName);
83                  BinaryFile.Write(list[i].GroupNumber);
84                  BinaryFile.Write(list[i].AverageMark);
85              }
86              BinaryFile.Close();
87              return list.Count;
88          }
89          public List<info> LowLevelStudents(string FileName, int AmountOfStudents)
90          {
91              List<info> MainList = new List<info>();
92              List<info> InfoList = new List<info>();
93              info StudentInfo;
94              BinaryReader BinaryFileToRead;
95              BinaryFileToRead = new BinaryReader(new FileStream(FileName, FileMode.OpenOrCreate, FileAccess.Read));
96              for (int i = 0; i < AmountOfStudents; i++)
97              {
98                  StudentInfo.name = BinaryFileToRead.ReadString();
99                  StudentInfo.DateOfBirth = BinaryFileToRead.ReadString();
100                 StudentInfo.FormOfStudying = BinaryFileToRead.ReadString();
101                 StudentInfo.GroupName = BinaryFileToRead.ReadString();
102                 StudentInfo.GroupNumber = BinaryFileToRead.ReadInt32();
103                 StudentInfo.AverageMark = BinaryFileToRead.ReadDouble();
104                 InfoList.Add(StudentInfo);
```

```
105             }
106             double LowestAverageMark;
107             int GroupNum;
108             info LowestStudent;
109             for (int i = 0; i < InfoList.Count;)
110             {
111                 LowestAverageMark = InfoList[i].AverageMark;
112                 GroupNum = InfoList[i].GroupNumber;
113                 LowestStudent = InfoList[i];
114                 for(int j = 0; j < InfoList.Count;)
115                 {
116                     if(InfoList[j].GroupNumber == GroupNum)
117                     {
118                         if(InfoList[j].AverageMark < LowestAverageMark)
119                         {
120                             LowestAverageMark = InfoList[j].AverageMark;
121                             LowestStudent = InfoList[j];
122                         }
123                         InfoList.Remove(InfoList[j]);
124                     }
125                     else j++;
126                 }
127                 MainList.Add(LowestStudent);
128             }
129             return MainList;
130         }
            1 reference
```

```
            1 reference
131         public List<info> Identify(List<info> list)
132         {
133             Console.Write("Pick the average mark: ");
134             double averageGrage = Convert.ToDouble(Console.ReadLine());
135             for(int i = 0; i < list.Count;)
136             {
137                 if(!(list[i].GroupNumber > 39 && list[i].GroupNumber < 50 && list[i].FormOfStudying == "Daily" && list[i].AverageMark
138                     list.Remove(list[i]);
139                 else i++;
140             }
141             if (list.Count() > 1)
142             {
143                 list.Sort();
144             }
145             return list;
146         }
147     }
148 }
```

```
137 )er > 39 && list[i].GroupNumber < 50 && list[i].FormOfStudying == "Daily" && list[i].AverageMark > averageGrage) && list.Count() != 0)
```

1. Python

Lab_2_py.py

```python
import functions

#Ім'я бінарних файлів
BinaryFileName = "BinaryFile.txt"
newBinaryFile = "newBinaryFile.txt"

#Список студентів
list_of_students = functions.create_info()
print("\nAll of the students:")
functions.output_students(list_of_students)

#Передача інформації в бінарний файл
AmountOfStudents = functions.CreateBinaryFile(BinaryFileName, list_of_students)

#Розшифровуємо інформацію
Readable = functions.FromByteToReadable(BinaryFileName)
print("\nAll of the students 2:")
functions.output_students(Readable)

#Студенти із найнижчими балами
Lowest = functions.LowLevelStudents(Readable)
print("Lowest mark students:")
functions.output_students(Lowest)

#Відсортований список студентів 4-го курсу
newList = functions.Identify(Lowest)
print("4-th grade students:")
functions.output_students(newList)
```

```python
Stud = functions.CreateBinaryFile(newBinaryFile, newList)
```

functions.py

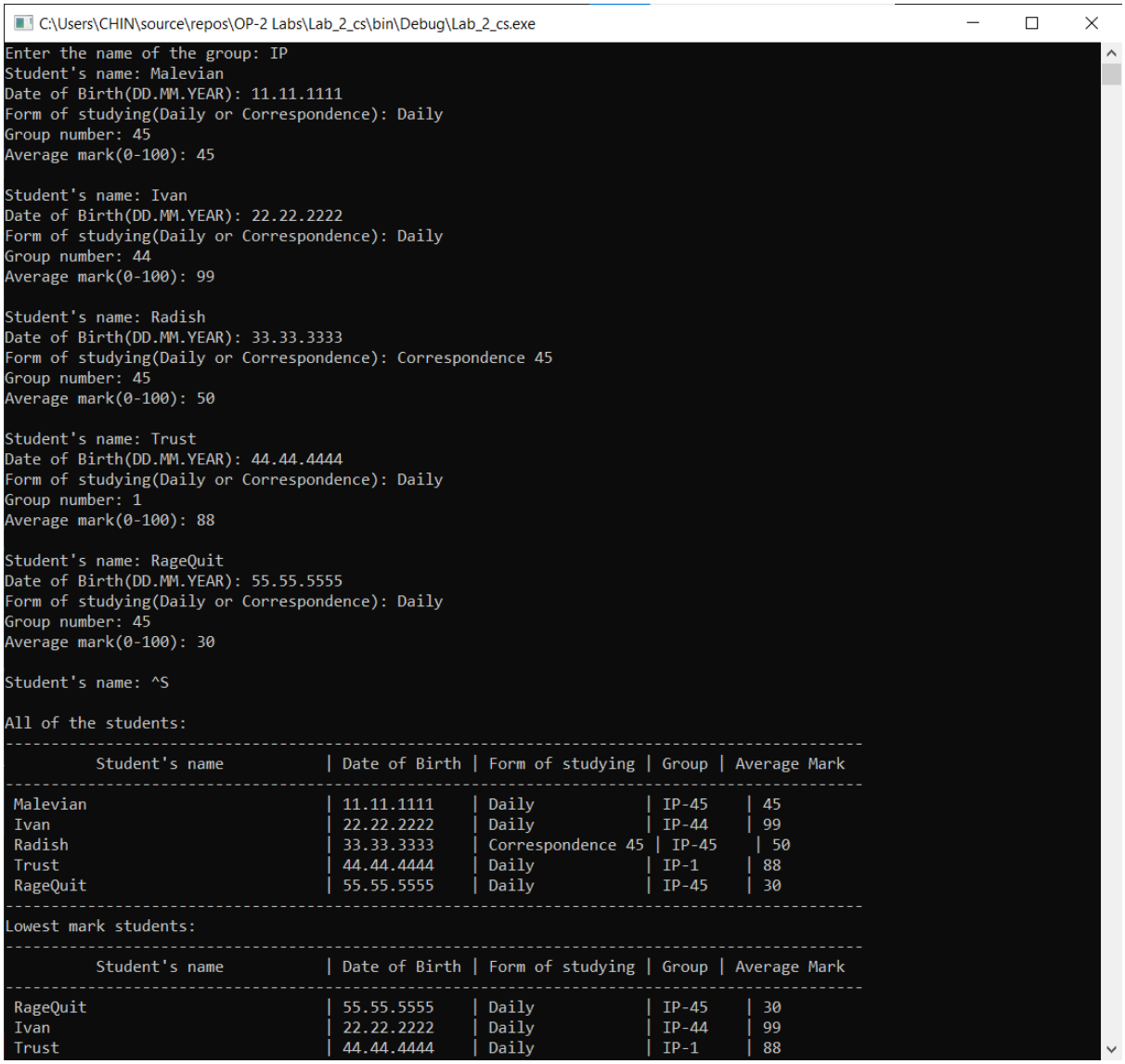
```python
import pickle

class info:
    def __init__(self, name1, DateOfBirth1, FormOfStudying1, GroupName1, GroupNumber1, AverageMark1):
        self.name = name1
        self.DateOfBirth = DateOfBirth1
        self.FormOfStudying = FormOfStudying1
        self.GroupName = GroupName1
        self.GroupNumber = GroupNumber1
        self.AverageMark = AverageMark1


def create_info():
    Students = []
    stop_botton = chr(19)
    print("Enter the name of the group: ")
    GroupName = input()
    while True:
        print("Student's name: ")
        additional = input()
        if additional[0] != stop_botton:
            name = additional

            print("Date of Birth(DD.MM.YEAR): ")
            DateOfBirth = input()

            print("Form of studying(Daily or Correspondence): ")
            FormOfStudying = input()
```

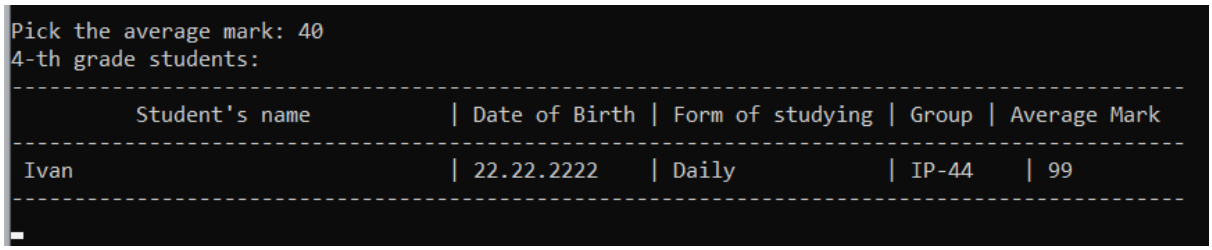
```python
            print("Group number: ")
            GroupNumber = input()

            print("Average mark(0-100): ")
            AverageMark = input()

            print()

            Student = info(name, DateOfBirth, FormOfStudying, GroupName, GroupNumber, AverageMark)
            Students.append(Student)
        else:
            break
    return Students

def output_students(Students):
    print("-----------------------------------------------------------------------------")
    print(" Student's name | Date of Birth | Form of studying | Group | Average Mark ")
    print("-----------------------------------------------------------------------------")
    for i in range(len(Students)):
        print(" {0} | {1} | {2} | {3}-{4} | {5} ".format(
                    Students[i].name, Students[i].DateOfBirth, Students[i].FormOfStudying,
                    Students[i].GroupName, Students[i].GroupNumber, Students[i].AverageMark))
    print("-----------------------------------------------------------------------------")

def CreateBinaryFile(fileName, Students):
    with open(fileName, "wb") as BinaryFile:
```

```
        pickle.dump(Students, BinaryFile)
    BinaryFile.close()
    return len(Students)

def FromByteToReadable(fileName):
    InfoList = []
    with open(fileName, "rb") as BinaryFile:
        List = pickle.load(BinaryFile)

    for what in List:
        InfoList.append(what)
    return InfoList

def LowLevelStudents(InfoList):
    MainList = []
    while len(InfoList) - 1 > 0:
        LowestAverageMark = InfoList[0].AverageMark
        GroupNum = InfoList[0].GroupNumber
        LowestStudent = InfoList[0]
        j = 0
        while j < len(InfoList) - 1:

            if InfoList[j].GroupNumber == GroupNum:

                if InfoList[j].AverageMark < LowestAverageMark:

                    LowestAverageMark = InfoList[j].AverageMark
                    LowestStudent = InfoList[j]

                InfoList.remove(InfoList[j])

            else: j += 1
        MainList.append(LowestStudent)
    return MainList

def Identify(ListOfStudents):
    print("Pick the average mark: ")
    average = float(input())
    i = 0
    while i < len(ListOfStudents):
        if int(ListOfStudents[i].GroupNumber) <= 39 or int(ListOfStudents[i].GroupNumber) >= 50 or ListOfStudents[i].FormOfStudying.u
            ListOfStudents.pop(i)
        else: i += 1

    if len(ListOfStudents) > 1:
        for i in range(len(ListOfStudents)):
            for j in range(i+1,len(ListOfStudents)):
                if ListOfStudents[i].name[0] >  ListOfStudents[j].name[0]:
                    ListOfStudents[i], ListOfStudents[j] = ListOfStudents[j], ListOfStudents[i]
    return ListOfStudents
```

```
ListOfStudents[i].FormOfStudying.upper() != "DAILY" or float(ListOfStudents[i].AverageMark) < average and len(ListOfStudents) != 0:
```

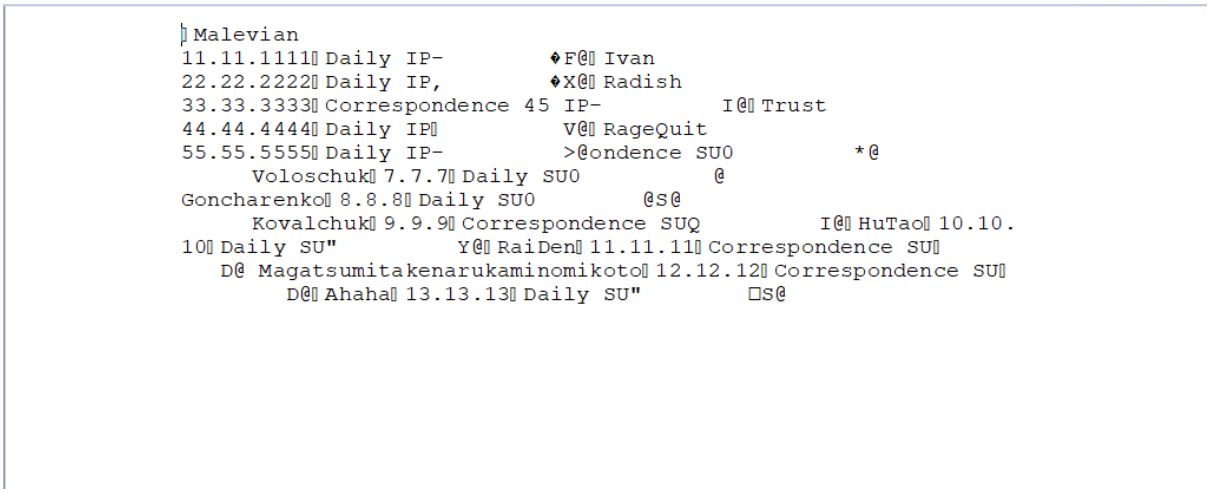
Вивід на C#





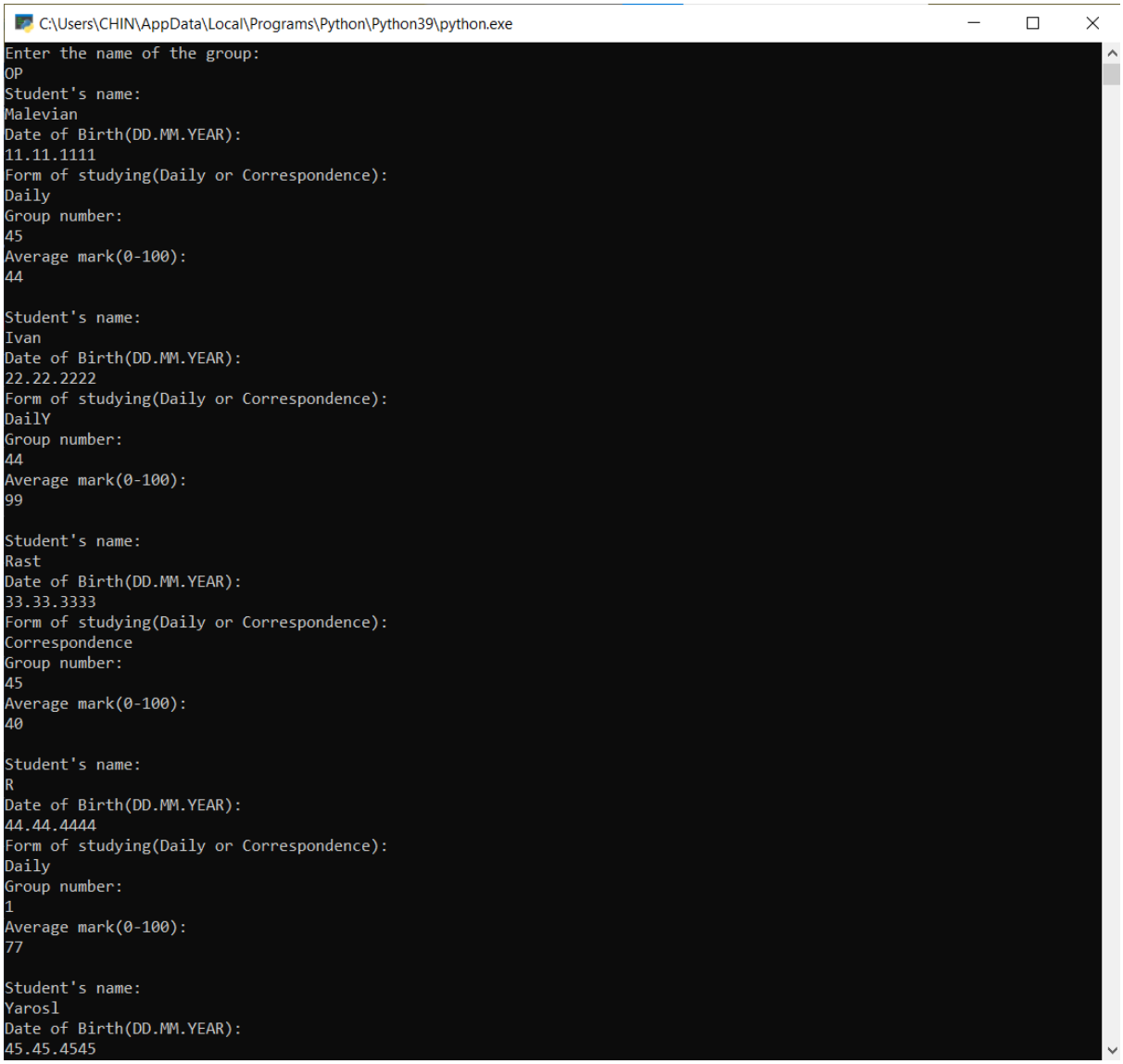
Файли C#:

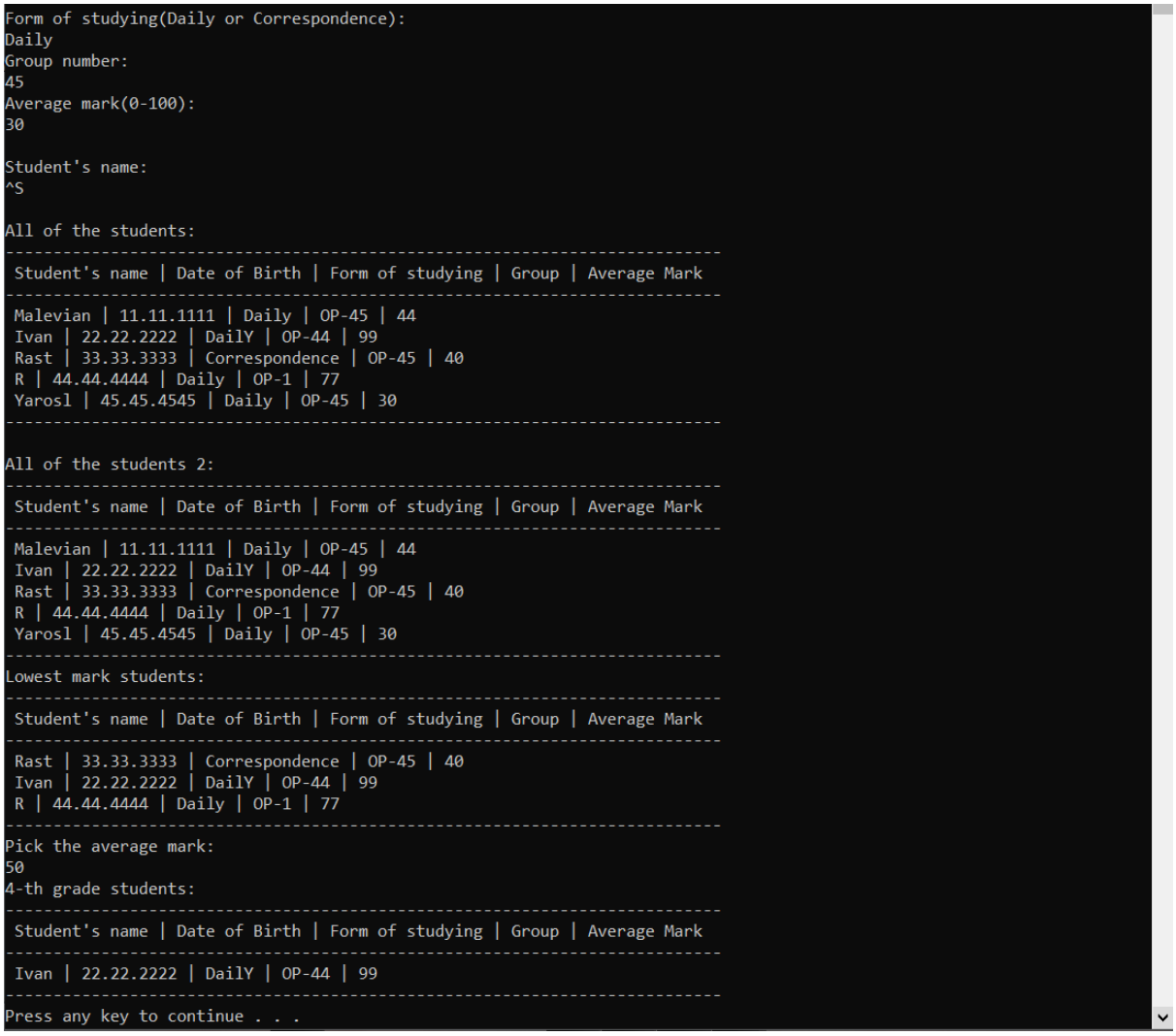
BinaryFile:



newBinaryFile:

Вивід на Python:


C:\Users\CHIN\AppData\Local\Programs\Python\Python39\python.exe
Enter the name of the group:
OP
Student's name:
Malevian
Date of Birth(DD.MM.YEAR):
11.11.1111
Form of studying(Daily or Correspondence):
Daily
Group number:
45
Average mark(0-100):
44
Student's name:
Ivan
Date of Birth(DD.MM.YEAR):
22.22.2222
Form of studying(Daily or Correspondence):
DailY
Group number:
44
Average mark(0-100):
99
Student's name:
Rast
Date of Birth(DD.MM.YEAR):
33.33.3333
Form of studying(Daily or Correspondence):
Correspondence
Group number:
45
Average mark(0-100):
40
Student's name:
R
Date of Birth(DD.MM.YEAR):
44.44.4444
Form of studying(Daily or Correspondence):
Daily
Group number:
1
Average mark(0-100):
77
Student's name:
Yarosl
Date of Birth(DD.MM.YEAR):
45.45.4545



Файли Python:

BinaryFile:

```
♦▯ ♦♦▯       ]□(□  functions♦♦▯ info♦♦♦)♦♦}♦(♦▯ name♦♦▯ Malevian♦♦
DateOfBirth□□
11.11.1111♦♦▯ FormOfStudying♦♦▯ Daily♦♦  GroupName♦♦  OP♦♦
GroupNumber♦♦  45♦♦
AverageMark♦♦  44♦ubh▯ )♦♦}♦(h▯ ♦▯ Ivan♦h▯ ♦
22.22.2222□h
♦▯ DailY♦h
h
h▯ ♦  44♦h▯ ♦  99♦ubh▯ )♦♦}♦(h▯ ♦▯ Rast♦h▯ ♦
33.33.3333□h
♦▯ Correspondence♦h
h
h▯ ♦  45♦h▯ ♦  40♦ubh▯ )♦♦}♦(h▯ ♦▯ R♦h▯ ♦
44.44.4444□h
♦▯ Daily♦h
h
h▯ ♦▯ 1♦h▯ ♦  77♦ubh▯ )♦♦}♦(h▯ ♦▯ Yarosl♦h▯ ♦
45.45.4545□h
♦▯ Daily♦h
h
h▯ ♦  45♦h▯ ♦  30♦ube.
```

newBinaryFile:

```
♦▯ ♦♦        ]□□   functions♦♦▯ info♦♦♦)♦♦}♦(♦▯ name♦♦▯ Ivan♦♦
DateOfBirth□□
22.22.2222♦♦▯ FormOfStudying♦♦▯ DailY♦♦ GroupName♦♦  OP♦♦
GroupNumber♦♦  44♦♦
AverageMark♦♦  99♦uba.
```