НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №1.2

з дисципліни

“Програмування мобільних систем”

Виконав:

студент групи ІО-82

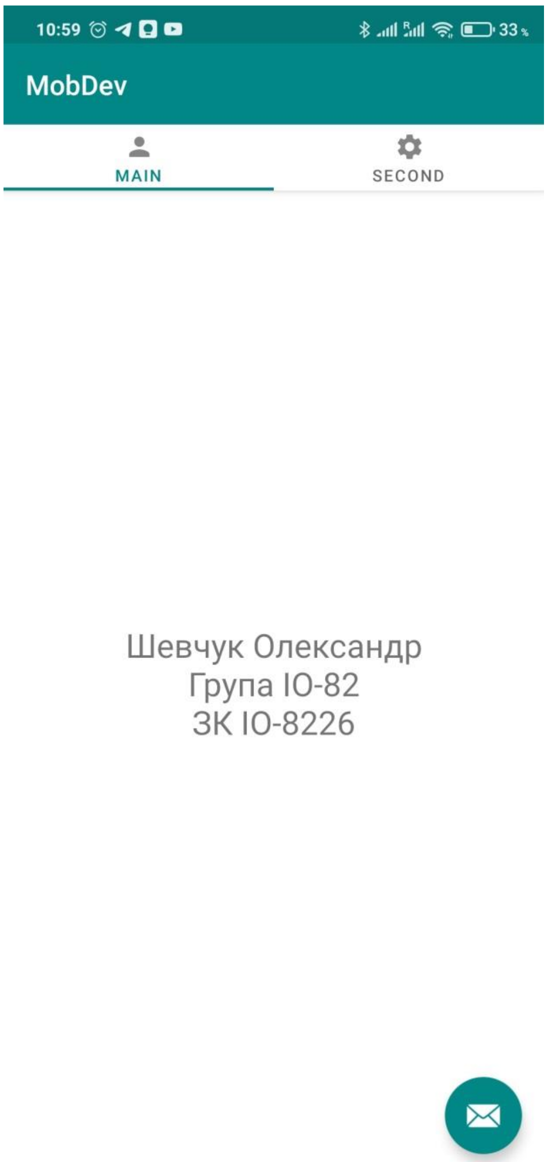
ЗК ІО-8226

Шевчук Олександр

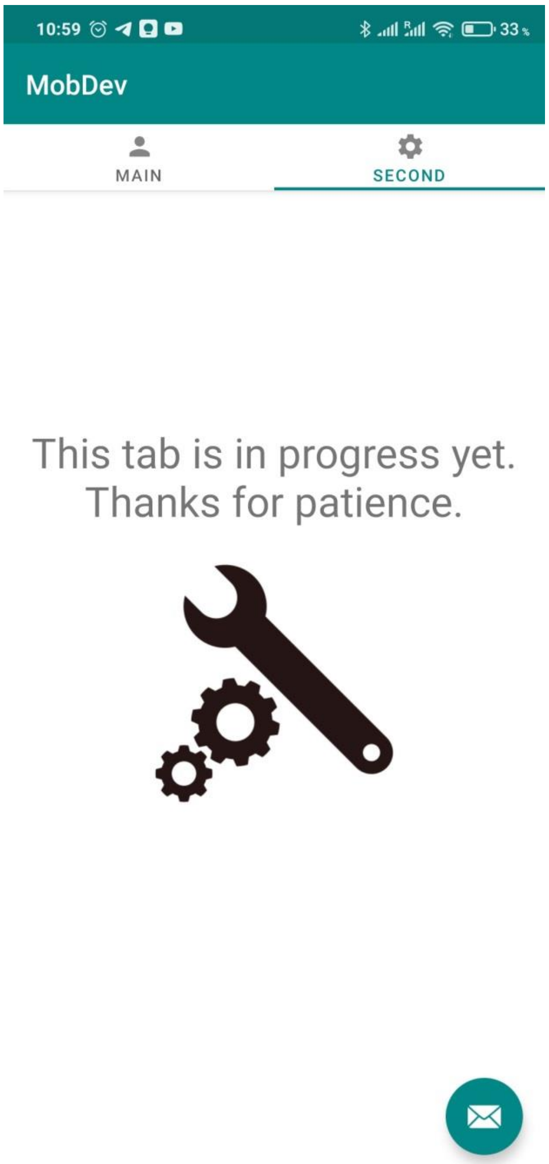
Київ 2021

Варіант №1

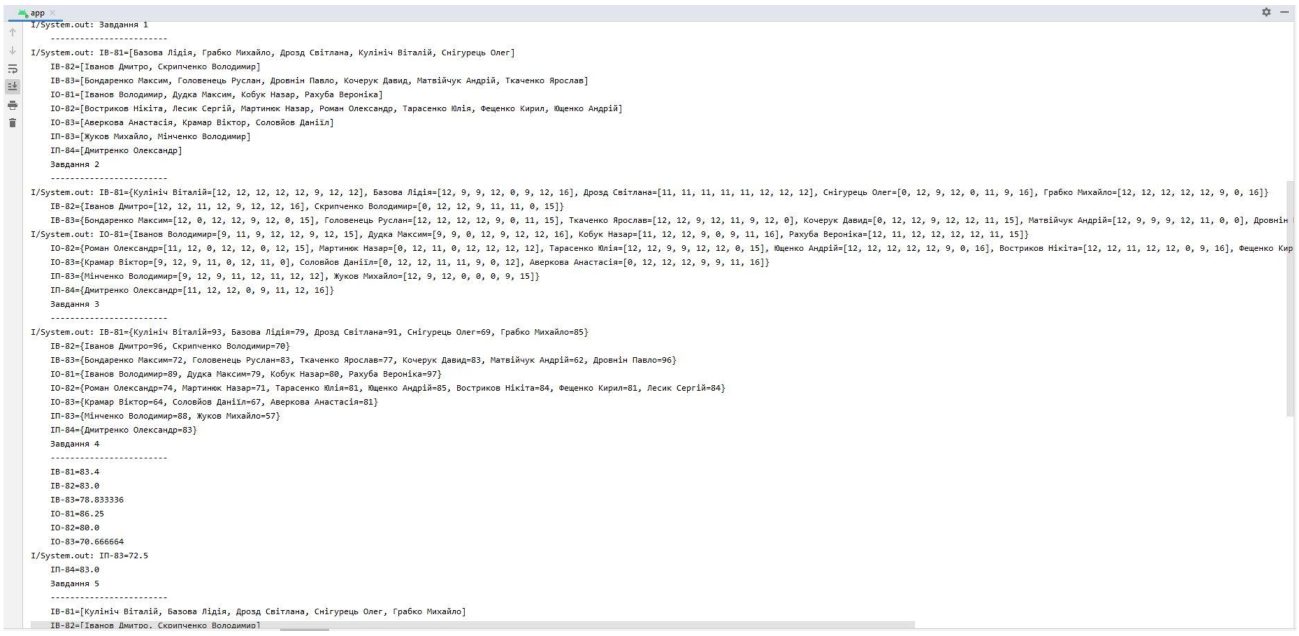
Скріншоти роботи додатку



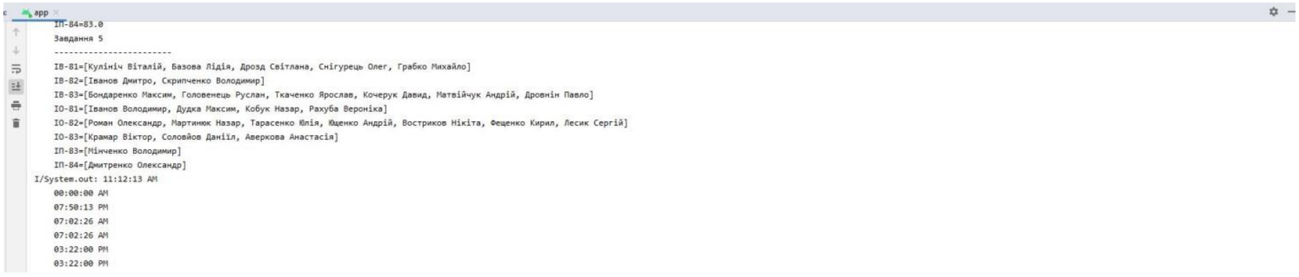
Шевчук Олександр
Група IO-82
ЗК IO-8226



This tab is in progress yet.
Thanks for patience.



```
I/System.out: Завдання 1
    ------------------------
I/System.out: IB-81=[Базова Лідія, Грабко Михайло, Дрозд Світлана, Кулініч Віталій, Снігурець Олег]
    IB-82=[Іванов Дмитро, Скрипченко Володимир]
    IB-83=[Бондаренко Максим, Головенець Руслан, Дровнін Павло, Кочерук Давид, Матвійчук Андрій, Ткаченко Ярослав]
    IO-81=[Іванов Володимир, Дудка Максим, Кобук Назар, Рахуба Вероніка]
    IO-82=[Востриков Нікіта, Лесик Сергій, Мартинюк Назар, Роман Олександр, Тарасенко Юлія, Фещенко Кирил, Ющенко Андрій]
    IO-83=[Аверкова Анастасія, Крамар Віктор, Соловйов Даніїл]
    IП-83=[Жуков Михайло, Мінченко Володимир]
    IП-84=[Дмитренко Олександр]
    Завдання 2
    ------------------------
I/System.out: IB-81={Кулініч Віталій=[12, 12, 12, 12, 12, 9, 12, 12], Базова Лідія=[12, 9, 9, 12, 0, 9, 12, 16], Дрозд Світлана=[11, 11, 11, 11, 11, 12, 12, 12], Снігурець Олег=[0, 12, 9, 12, 0, 11, 9, 16], Грабко Михайло=[12, 12, 12, 12, 12, 9, 0, 16]}
    IB-82={Іванов Дмитро=[12, 12, 11, 12, 9, 12, 12, 16], Скрипченко Володимир=[0, 12, 12, 9, 11, 11, 0, 15]}
    IB-83={Бондаренко Максим=[12, 0, 12, 12, 9, 12, 0, 15], Головенець Руслан=[12, 12, 12, 12, 9, 0, 11, 15], Ткаченко Ярослав=[12, 12, 9, 12, 11, 9, 12, 0], Кочерук Давид=[0, 12, 12, 9, 12, 12, 11, 15], Матвійчук Андрій=[12, 9, 9, 9, 12, 11, 0, 0], Дровнін
I/System.out: IO-81={Іванов Володимир=[9, 11, 9, 12, 12, 9, 12, 15], Дудка Максим=[9, 9, 0, 12, 9, 12, 12, 16], Кобук Назар=[11, 12, 12, 9, 0, 9, 11, 16], Рахуба Вероніка=[12, 11, 12, 12, 12, 12, 11, 15]}
    IO-82={Роман Олександр=[11, 12, 0, 12, 12, 0, 12, 15], Мартинюк Назар=[0, 12, 11, 0, 12, 12, 12, 12], Тарасенко Юлія=[12, 12, 9, 9, 12, 12, 0, 15], Ющенко Андрій=[12, 12, 12, 12, 12, 9, 0, 16], Востриков Нікіта=[12, 12, 11, 12, 12, 0, 9, 16], Фещенко Кир
    IO-83={Крамар Віктор=[9, 12, 9, 11, 0, 12, 11, 0], Соловйов Даніїл=[0, 12, 12, 11, 11, 9, 0, 12], Аверкова Анастасія=[0, 12, 12, 12, 9, 9, 11, 16]}
    IП-83={Мінченко Володимир=[9, 12, 9, 11, 12, 11, 12, 12], Жуков Михайло=[12, 9, 12, 0, 0, 0, 9, 15]}
    IП-84={Дмитренко Олександр=[11, 12, 12, 0, 9, 11, 12, 16]}
    Завдання 3
    ------------------------
I/System.out: IB-81={Кулініч Віталій=93, Базова Лідія=79, Дрозд Світлана=91, Снігурець Олег=69, Грабко Михайло=85}
    IB-82={Іванов Дмитро=96, Скрипченко Володимир=70}
    IB-83={Бондаренко Максим=72, Головенець Руслан=83, Ткаченко Ярослав=77, Кочерук Давид=83, Матвійчук Андрій=62, Дровнін Павло=96}
    IO-81={Іванов Володимир=89, Дудка Максим=79, Кобук Назар=80, Рахуба Вероніка=97}
    IO-82={Роман Олександр=74, Мартинюк Назар=71, Тарасенко Юлія=81, Ющенко Андрій=85, Востриков Нікіта=84, Фещенко Кирил=81, Лесик Сергій=84}
    IO-83={Крамар Віктор=64, Соловйов Даніїл=67, Аверкова Анастасія=81}
    IП-83={Мінченко Володимир=88, Жуков Михайло=57}
    IП-84={Дмитренко Олександр=83}
    Завдання 4
    ------------------------
    IB-81=83.4
    IB-82=83.0
    IB-83=78.833336
    IO-81=86.25
    IO-82=80.0
    IO-83=70.666664
I/System.out: IП-83=72.5
    IП-84=83.0
    Завдання 5
    ------------------------
    IB-81=[Кулініч Віталій, Базова Лідія, Дрозд Світлана, Снігурець Олег, Грабко Михайло]
    IB-82=[Іванов Дмитро, Скрипченко Володимир]
```



```
    IП-84=83.0
    Завдання 5
    ------------------------
    IB-81=[Кулініч Віталій, Базова Лідія, Дрозд Світлана, Снігурець Олег, Грабко Михайло]
    IB-82=[Іванов Дмитро, Скрипченко Володимир]
    IB-83=[Бондаренко Максим, Головенець Руслан, Ткаченко Ярослав, Кочерук Давид, Матвійчук Андрій, Дровнін Павло]
    IO-81=[Іванов Володимир, Дудка Максим, Кобук Назар, Рахуба Вероніка]
    IO-82=[Роман Олександр, Мартинюк Назар, Тарасенко Юлія, Ющенко Андрій, Востриков Нікіта, Фещенко Кирил, Лесик Сергій]
    IO-83=[Крамар Віктор, Соловйов Даніїл, Аверкова Анастасія]
    IП-83=[Мінченко Володимир]
    IП-84=[Дмитренко Олександр]
I/System.out: 11:12:13 AM
    00:00:00 AM
    07:50:13 PM
    07:02:26 AM
    07:02:26 AM
    03:22:00 PM
    03:22:00 PM
```

Лістинг коду

Contents.java

```
package ua.kpi.compsys.io8226;

import java.util.ArrayList;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;
import java.util.Random;


public class Contents {
    public static final String studentStr = "Дмитренко Олександр - ІП-84; 
Матвійчук Андрій - ІВ-83; " +
            "Лесик Сергій - ІО-82; Ткаченко Ярослав - ІВ-83; Аверкова Анастасія - 
ІО-83; " +
            "Соловйов Даніїл - ІО-83; Рахуба Вероніка - ІО-81; Кочерук Давид - ІВ-
83; " +
            "Лихацька Юлія- ІВ-82; Головенець Руслан - ІВ-83; Ющенко Андрій - ІО-
82; " +
            "Мінченко Володимир - ІП-83; Мартинюк Назар - ІО-82; Базова Лідія -
ІВ-81; " +
            "Снігурець Олег - ІВ-81; Роман Олександр - ІО-82; Дудка Максим - ІО-
81; " +
            "Кулініч Віталій - ІВ-81; Жуков Михайло - ІП-83; Грабко Михайло - ІВ-
81; " +
            "Іванов Володимир - ІО-81; Востриков Нікіта - ІО-82; Бондаренко Максим
- ІВ-83; " +
            "Скрипченко Володимир - ІВ-82; Кобук Назар - ІО-81; Дровнін Павло -
ІВ-83; " +
            "Тарасенко Юлія - ІО-82; Дрозд Світлана - ІВ-81; Фещенко Кирил - ІО-
82; " +
            "Крамар Віктор - ІО-83; Іванов Дмитро - ІВ-82";


    // TASK 1

    /**
     * Splitting String on 'Student - Group' pairs, then split them again and add
to
     * 'student', 'group' ArrayList.
     * Then passing through this List to detect student and
     * this group, and add them to final Set;
     */

    public HashMap<String, ArrayList<String>> groupStudents(String studentStr) {
        HashMap<String, ArrayList<String>> studentsGroups = new HashMap<>();
        String[] splitted1 = studentStr.split("; ");
        ArrayList<String> splitted2 = new ArrayList<>();
        String elem;

        for (String student : splitted1) {
            String[] studGroupPair = student.split(" - ");
            if (studGroupPair.length == 2) {
                splitted2.add(studGroupPair[0]);
                splitted2.add(studGroupPair[1]);
            }
        }

        for (int i = 0; i < splitted2.size(); i++) {
            elem = splitted2.get(i);
            if (i % 2 == 1) {
                if (!studentsGroups.containsKey(elem))
                    studentsGroups.put(elem, new ArrayList<>());

                studentsGroups.get(elem).add(splitted2.get(i - 1));
            }
        }

        for (Map.Entry<String, ArrayList<String>> group:
studentsGroups.entrySet()) {
            Collections.sort(group.getValue());
        }

        studentsGroups.entrySet()
                .stream()
                .sorted(Map.Entry.<String, ArrayList<String>>comparingByKey())
                .forEach(System.out::println);
```

```java
        return studentsGroups;

    }

    // TASK 2 -- Helper Method

    /**
     * Returns random grade for student.
     */
    private int randomValue(int maxValue) {
        Random random = new Random();
        switch (random.nextInt(6)) {
            case 1:
                return (int) Math.ceil((float) maxValue * 0.7);
            case 2:
                return (int) Math.ceil((float) maxValue * 0.9);
            case 3:
            case 4:
            case 5:
                return maxValue;
            default:
                return 0;
        }
    }

    // TASK 2.

    /**
     * Creates and fills HashMap with Group and Students (including his/her
grades) pairs.
     */
    public HashMap<String, HashMap<String, ArrayList<Integer>>>
fillGrades(HashMap<String,
            ArrayList<String>> studentsGroups, int[] points) {

        HashMap<String, HashMap<String, ArrayList<Integer>>> grades = new
HashMap<>();

        for (Map.Entry<String, ArrayList<String>> group:
studentsGroups.entrySet()) {

            HashMap<String, ArrayList<Integer>> studentsOfTheGroup = new
HashMap<>();

            for (String student : group.getValue()) {

                ArrayList<Integer> randGrades = new ArrayList<>();

                for (int point : points) {
                    randGrades.add(randomValue(point));
                }

                studentsOfTheGroup.put(student, randGrades);
            }
            grades.put(group.getKey(), studentsOfTheGroup);
        }

        grades.entrySet()
```

```
                .stream()
                .sorted(Map.Entry.<String, HashMap<String,
ArrayList<Integer>>>comparingByKey())
                .forEach(System.out::println);

        return grades;
    }

    // TASK 3

    /**
     * Returns HashMap with Group - Average grade pairs.
     */
    public HashMap<String, HashMap<String, Integer>> showGradesSum(HashMap<String,
            HashMap<String, ArrayList<Integer>>> grades) {

        HashMap<String, HashMap<String, Integer>> sumGrades = new HashMap<>();

        for (Map.Entry<String, HashMap<String, ArrayList<Integer>>> group :
grades.entrySet()) {

            HashMap<String, Integer> studGrade = new HashMap<>();

            for (Map.Entry<String, ArrayList<Integer>> student :
group.getValue().entrySet()) {

                int sum = 0;
                for (int i : student.getValue()) {
                    sum += i;
                }

                studGrade.put(student.getKey(), sum);
            }
            sumGrades.put(group.getKey(), studGrade);
        }

        sumGrades.entrySet()
                .stream()
                .sorted(Map.Entry.<String, HashMap<String,
Integer>>comparingByKey())
                .forEach(System.out::println);

        return sumGrades;
    }

    // TASK 4

    /**
     * Returns HashMap with Group - Average grade pairs.
     */
    public HashMap<String, Float> showAvgGradesInGroups(HashMap<String,
            HashMap<String, Integer>> sumGrades) {

        HashMap<String, Float> averages = new HashMap<>();

        for (Map.Entry<String, HashMap<String, Integer>> group:
sumGrades.entrySet()) {

            float sumInGroup = 0;
            for (Map.Entry<String, Integer> student: group.getValue().entrySet())
{
                sumInGroup += student.getValue();
            }

            float avgGrade = (float) sumInGroup / group.getValue().size();
            averages.put(group.getKey(), avgGrade);
        }

        averages.entrySet()
                .stream()
                .sorted(Map.Entry.<String, Float>comparingByKey())
                .forEach(System.out::println);

        return averages;
    }

    // TASK 5

    /**
     * Returns HashMap with Group - The most successful students (> 60 points)
     */
    public HashMap<String, ArrayList<String>> showBestInGroups(HashMap<String,
            HashMap<String, Integer>> sumGrades) {

        HashMap<String, ArrayList<String>> bests = new HashMap<>();

        for (Map.Entry<String, HashMap<String, Integer>> group:
sumGrades.entrySet()) {

            ArrayList<String> bestStudents = new ArrayList<>();

            for (Map.Entry<String, Integer> student: group.getValue().entrySet())
{
                if (student.getValue() >= 60)
                    bestStudents.add(student.getKey());

            }

            bests.put(group.getKey(), bestStudents);
        }

        bests.entrySet()
                .stream()
                .sorted(Map.Entry.<String, ArrayList<String>>comparingByKey())
                .forEach(System.out::println);

        return bests;
    }
}
```

TimeOS.java

```
package ua.kpi.compsys.io8226;

import androidx.annotation.NonNull;
import java.util.Date;
```

```java
public class TimeOS {
    int hours;
    int minutes;
    int seconds;

    /**
     * Default constructor, that sets time to 00:00:00 AM
     */
    public TimeOS() {
        setTime(0, 0, 0);
    }

    /**
     * The constructor sets time that has been got as parameters
     */
    public TimeOS(int hours, int minutes, int seconds) {
        if ((hours >= 0 && hours <=23) &&
                (minutes >= 0 && minutes <= 59) &&
                (seconds >= 0 && seconds <= 59)) {

            setTime(hours, minutes, seconds);
        } else {
            System.out.println("Object TimeOS was not created! Time must be 
between 00:00:00 " +
                    "and 23:59:59.");
        }
    }

    /**
     * Constructors gets time from Date object transferred as parameter.
     */
    public TimeOS(Date date) {
        this.hours = date.getHours();
        this.minutes = date.getMinutes();
        this.seconds = date.getSeconds();
    }

    /**
     * Static method for adding 2 TimeOS object transferred as parameters.
     */
    public static TimeOS add2Times(TimeOS time1, TimeOS time2) {
        TimeOS resTime = new TimeOS(time1.hours, time1.minutes, time1.seconds);

        if (time1.hours + time2.hours >= 24)
            resTime.hours = (time1.hours + time2.hours) % 24;
        else
            resTime.hours = time1.hours + time2.hours;

        if (time1.minutes + time2.minutes >= 60) {
            resTime.minutes = (time1.minutes + time2.minutes) % 60;
            if (resTime.hours != 23)
                resTime.hours ++;
            else
                resTime.hours = 0;
        } else resTime.minutes = time1.minutes + time2.minutes;

        if (time1.seconds + time2.seconds >= 60) {
            resTime.seconds = (time1.seconds + time2.seconds) % 60;
            if (resTime.minutes != 59)
                resTime.minutes ++;
            else
                resTime.minutes = 0;
                if (resTime.hours != 23)
                    resTime.hours ++;
                else
                    resTime.hours = 0;
        } else resTime.seconds = time1.seconds + time2.seconds;

        return resTime;
    }


    /**
     * Static method for subtracting 2 TimeOS object transferred as parameters.
     */
    public static TimeOS subtract2Times(TimeOS time1, TimeOS time2) {
        TimeOS resTime = new TimeOS(time1.hours, time1.minutes, time1.seconds);

        if (time1.hours >= time2.hours)
            resTime.hours = time1.hours - time2.hours;
        else {
            if (time2.hours < 24)
                resTime.hours = 24 - (time2.hours - time1.hours);
            else if (time2.hours > 24)
                resTime.hours = 24 - ((time2.hours - time1.hours) % 24);
        }

        if (time1.minutes >= time2.minutes) {
            resTime.minutes = time1.minutes - time2.minutes;
        } else {
            if (time2.minutes < 60)
                resTime.minutes = 60 - (time2.minutes - time1.minutes);
            else if (time2.minutes > 60)
                resTime.minutes = 60 - ((time2.minutes - time1.minutes) % 60);
            if (resTime.hours != 0)
                resTime.hours --;
            else
                resTime.hours = 23;
        }

        if (time1.seconds >= time2.seconds) {
            resTime.seconds = time1.seconds - time2.seconds;
        } else {
            if (time2.seconds < 60)
                resTime.seconds = 60 - (time2.seconds - time1.seconds);
            else if (time2.seconds > 60)
                resTime.seconds = 60 - ((time2.seconds - time1.seconds) % 60);
            if (resTime.minutes != 0)
                resTime.minutes --;
            else {
                resTime.minutes = 59;
                if (resTime.hours != 0)
                    resTime.hours--;
                else
                    resTime.hours = 23;
            }
        }

        return resTime;
    }

    /**
     * Method for adding TimeOS object to the current instance.
     */
    public TimeOS addTime(TimeOS time) {
        TimeOS resTime = new TimeOS(hours, minutes, seconds);

        if (resTime.hours + time.hours >= 24)
            resTime.hours = (resTime.hours + time.hours) % 24;
        else
            resTime.hours += time.hours;

        if (resTime.minutes + time.minutes >= 60) {
            resTime.minutes = (resTime.minutes + time.minutes) % 60;
            if (resTime.hours != 23)
                resTime.hours ++;
            else
                resTime.hours = 0;
        } else resTime.minutes += time.minutes;

        if (resTime.seconds + time.seconds >= 60) {
            resTime.seconds = (resTime.seconds + time.seconds) % 60;
            if (resTime.minutes != 59)
                resTime.minutes ++;
            else
                resTime.minutes = 0;
            if (resTime.hours != 23)
                resTime.hours ++;
            else
                resTime.hours = 0;
        } else resTime.seconds += time.seconds;

        return resTime;
    }

    /**
     * Method for subtracting TimeOS object from the current instance.
     */
    public TimeOS subtractTime(TimeOS time) {
        TimeOS resTime = new TimeOS(hours, minutes, seconds);

        if (resTime.hours >= time.hours)
            resTime.hours -= time.hours;
        else {
            if (time.hours < 24)
                resTime.hours = 24 - (time.hours - resTime.hours);
            else if (time.hours > 24)
                resTime.hours = 24 - ((time.hours - resTime.hours) % 24);
        }

        if (resTime.minutes >= time.minutes) {
            resTime.minutes -= time.minutes;
        } else {
            if (time.minutes < 60)
                resTime.minutes = 60 - (time.minutes - resTime.minutes);
            else if (time.minutes > 60)
                resTime.minutes = 60 - ((time.minutes - resTime.minutes) % 60);
            if (resTime.hours != 0)
                resTime.hours --;
            else
                resTime.hours = 23;
        }

        if (resTime.seconds >= time.seconds) {
            resTime.seconds -= time.seconds;
        } else {
            if (time.seconds < 60)
                resTime.seconds = 60 - (time.seconds - resTime.seconds);
            else if (time.seconds > 60)
                resTime.seconds = 60 - ((time.seconds - resTime.seconds) % 60);
            if (resTime.minutes != 0)
                resTime.minutes --;
            else {
                resTime.minutes = 59;
                if (resTime.hours != 0)
                    resTime.hours--;
                else
                    resTime.hours = 23;
            }
        }

        return resTime;
    }


    @NonNull
    @Override
    public String toString() {
        String hours = "";
        String minutes = "";
        String seconds = "";
        String partOfTheDay;

        if (this.hours >= 12) {
            if (this.hours - 12 < 10)
                hours += "0";
            hours += this.hours - 12;
            partOfTheDay = "PM";
        } else  {
            if (this.hours < 10)
                hours += "0";
            hours += this.hours;
            partOfTheDay = "AM";
        }

        minutes = this.minutes > 9 ? String.valueOf(this.minutes) : ("0" +
this.minutes);
        seconds = this.seconds > 9 ? String.valueOf(this.seconds) : ("0" +
this.seconds);

        return hours + ":" + minutes + ":" + seconds + " " + partOfTheDay;
    }
```

```java
    private void setTime(int hours, int minutes, int seconds) {
        this.hours = hours;
        this.minutes = minutes;
        this.seconds = seconds;
    }
}
```

PageViewModel.java

```java
package ua.kpi.compsys.io8226.ui.main;

import androidx.arch.core.util.Function;
import androidx.lifecycle.LiveData;
import androidx.lifecycle.MutableLiveData;
import androidx.lifecycle.Transformations;
import androidx.lifecycle.ViewModel;

public class PageViewModel extends ViewModel {

    private MutableLiveData<Integer> mIndex = new
MutableLiveData<>();
    private LiveData<String> mText = Transformations.map(mIndex, new
Function<Integer, String>() {
        @Override
        public String apply(Integer input) {
            return "Hello world from section: " + input;
        }
    });

    public void setIndex(int index) {
        mIndex.setValue(index);
    }

    public LiveData<String> getText() {
        return mText;
    }
}
```

PlaceholderFragment.java

```java
package ua.kpi.compsys.io8226.ui.main;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.Nullable;
import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProvider;
import ua.kpi.compsys.io8226.R;

/**
 * A placeholder fragment containing a simple view.
 */
public class PlaceholderFragment extends Fragment {

    private static final String ARG_SECTION_NUMBER =
"section_number";

    private PageViewModel pageViewModel;

    public static PlaceholderFragment newInstance(int index) {
        PlaceholderFragment fragment = new PlaceholderFragment();
        Bundle bundle = new Bundle();
        bundle.putInt(ARG_SECTION_NUMBER, index);
        fragment.setArguments(bundle);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        pageViewModel = new
ViewModelProvider(this).get(PageViewModel.class);
        int index = 1;
        if (getArguments() != null) {
            index = getArguments().getInt(ARG_SECTION_NUMBER);
        }
        pageViewModel.setIndex(index);
    }

    @Override
    public View onCreateView(
            @NonNull LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {
        View root = inflater.inflate(R.layout.fragment_main,
container, false);
        //final TextView textView =
root.findViewById(R.id.section_label);
        final TextView mainLable =
root.findViewById(R.id.mainLable_textView);
        pageViewModel.getText().observe(this, new Observer<String>()
{
            @Override
            public void onChanged(@Nullable String s) {
             //   textView.setText(s);
//                mainLable.isShown();
            }
        });
        return root;
    }
}
```

SectionsPagerAdapter.java

```
package ua.kpi.compsys.io8226.ui.main;

import android.content.Context;
import android.graphics.drawable.Drawable;
import android.text.SpannableStringBuilder;
import android.text.Spanned;
import android.text.style.DynamicDrawableSpan;
import android.text.style.ImageSpan;

import androidx.annotation.DrawableRes;
import androidx.annotation.Nullable;
import androidx.annotation.StringRes;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentPagerAdapter;

import ua.kpi.compsys.io8226.FragmentMain;
import ua.kpi.compsys.io8226.FragmentSecond;
import ua.kpi.compsys.io8226.R;

/**
 * A [FragmentPagerAdapter] that returns a fragment corresponding to
 * one of the sections/tabs/pages.
 */
public class SectionsPagerAdapter extends FragmentPagerAdapter {

    @StringRes
    private static final int[] TAB_TITLES = new
int[]{R.string.tab_text_1, R.string.tab_text_2};
    private final Context mContext;
    Drawable myDrawable;
    String title;

    public SectionsPagerAdapter(Context context, FragmentManager fm)
{
        super(fm);
        mContext = context;
    }

    @Override
    public Fragment getItem(int position) {
        // getItem is called to instantiate the fragment for the
given page.
        // Return a PlaceholderFragment (defined as a static inner
class below).
// return back if broken
        //return PlaceholderFragment.newInstance(position + 1);

        Fragment fragment = null;
        switch (position) {
            case 0:
                fragment = new FragmentMain();
                break;
            case 1:
                fragment = new FragmentSecond();
                break;
        }
        return fragment;
    }

    @Nullable
    @Override
    public CharSequence getPageTitle(int position) {
        switch (position) {
            case 0:
                myDrawable = mContext.getResources().
                        getDrawable(R.drawable.ic_tab_main);
                title =
mContext.getResources().getString(TAB_TITLES[0]);
                break;
            case 1:
                myDrawable = mContext.getResources().
                        getDrawable(R.drawable.ic_tab_second);
                title =
mContext.getResources().getString(TAB_TITLES[1]);
                break;
            default:
                //TODO: handle default selection
                break;
        }

        SpannableStringBuilder sb = new SpannableStringBuilder(" \n"
+ title); // space added before text for convenience

        myDrawable.setBounds(5, 5, myDrawable.getIntrinsicWidth(),
myDrawable.getIntrinsicHeight());
        ImageSpan span = new ImageSpan(myDrawable,
DynamicDrawableSpan.ALIGN_BASELINE);
        sb.setSpan(span, 0, 1, Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
        return sb;

       // return
mContext.getResources().getString(TAB_TITLES[position]);
    }


    @Override
    public int getCount() {
        // Show 2 total pages.
        return 2;
    }
}
```

MainActivity.java

```
package ua.kpi.compsys.io8226;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;

import
com.google.android.material.floatingactionbutton.FloatingActionButto
n;
import com.google.android.material.snackbar.Snackbar;
import com.google.android.material.tabs.TabLayout;

import androidx.viewpager.widget.ViewPager;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Handler;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import ua.kpi.compsys.io8226.ui.main.SectionsPagerAdapter;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SectionsPagerAdapter sectionsPagerAdapter = new
SectionsPagerAdapter(this,
                getSupportFragmentManager());
        ViewPager viewPager = findViewById(R.id.view_pager);
        viewPager.setAdapter(sectionsPagerAdapter);
        TabLayout tabs = findViewById(R.id.tabs);
        tabs.setupWithViewPager(viewPager);
        FloatingActionButton fab = findViewById(R.id.fab);

        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                    sendMail();
            }
        });
    }

    private void sendMail() {

        Intent send = new Intent(Intent.ACTION_SENDTO);
        String uriText = "mailto:" +
Uri.encode("legolasokay@gmail.com") +
                "?subject=" + Uri.encode("Р’С–РґРіСѓРє РґРѕ MobDev
РґРѕРґР°С‚РєСѓ");
        Uri uri = Uri.parse(uriText);

        send.setData(uri);
        startActivity(Intent.createChooser(send, "Р—Р°Р»РёС€РёС‚Рµ
РІС–РґРіСѓРє:"));

    }
}
```

FragmentMain.java

```
package ua.kpi.compsys.io8226;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class FragmentMain extends Fragment {
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
@Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_main, container,
false);
    }
}
```

FragmentSecond.java

```
package ua.kpi.compsys.io8226;

import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;


public class FragmentSecond extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {

        // ------------------------- PART 1 ---------------------------------


        // Task 1

        Contents contents = new Contents();
        System.out.println("Завдання 1");
        System.out.println("------------------------");
        HashMap<String, ArrayList<String>> groupedStudents =
                contents.groupStudents(Contents.studentStr);
        System.out.println();

        // Task 2

        System.out.println("Завдання 2");
        System.out.println("------------------------");
        int[] points = new int[] {12, 12, 12, 12, 12, 12, 12, 16};
        HashMap<String, HashMap<String, ArrayList<Integer>>> grades =
                contents.fillGrades(groupedStudents, points);
        System.out.println();

        // Task 3

        System.out.println("Завдання 3");
        System.out.println("------------------------");
        HashMap<String, HashMap<String, Integer>> gradesSum =
contents.showGradesSum(grades);
        System.out.println();

        // Task 4

        System.out.println("Завдання 4");
        System.out.println("------------------------");
        HashMap<String, Float> averages =
contents.showAvgGradesInGroups(gradesSum);
        System.out.println();

        // Task 5

        System.out.println("Завдання 5");
        System.out.println("------------------------");
        HashMap<String, ArrayList<String>> bests =
contents.showBestInGroups(gradesSum);
        System.out.println();




        // -------------------------- PART 2 ----------------------------------


        TimeOS time1 = new TimeOS(11,12,13);
        TimeOS time2 = new TimeOS();
        TimeOS time3 = new TimeOS(new Date());
        System.out.println(time1.toString());
        System.out.println(time2.toString());
        System.out.println(time3.toString());
        System.out.println(time1.addTime(time3));
        System.out.println(TimeOS.add2Times(time1, time3));
        System.out.println(time1.subtractTime(time3));
        System.out.println(TimeOS.subtract2Times(time1, time3));


        // --------------------------------------------------------------------

        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_second, container, false);
```

```
    }
}
```

Висновок

В результаті виконання лабораторної я додав 2 нових класи до проєкту: TimeOS і Contents. Contents містить методи для роботи із заданим рядком, TimeOS описує об’єкт часу та містить статичні і нестатичні методи для роботи із такими об’єктами. Врешті, я перевірив працездатність додатку та коректність даних, що виводяться, на AVD та власному пристрої.