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:

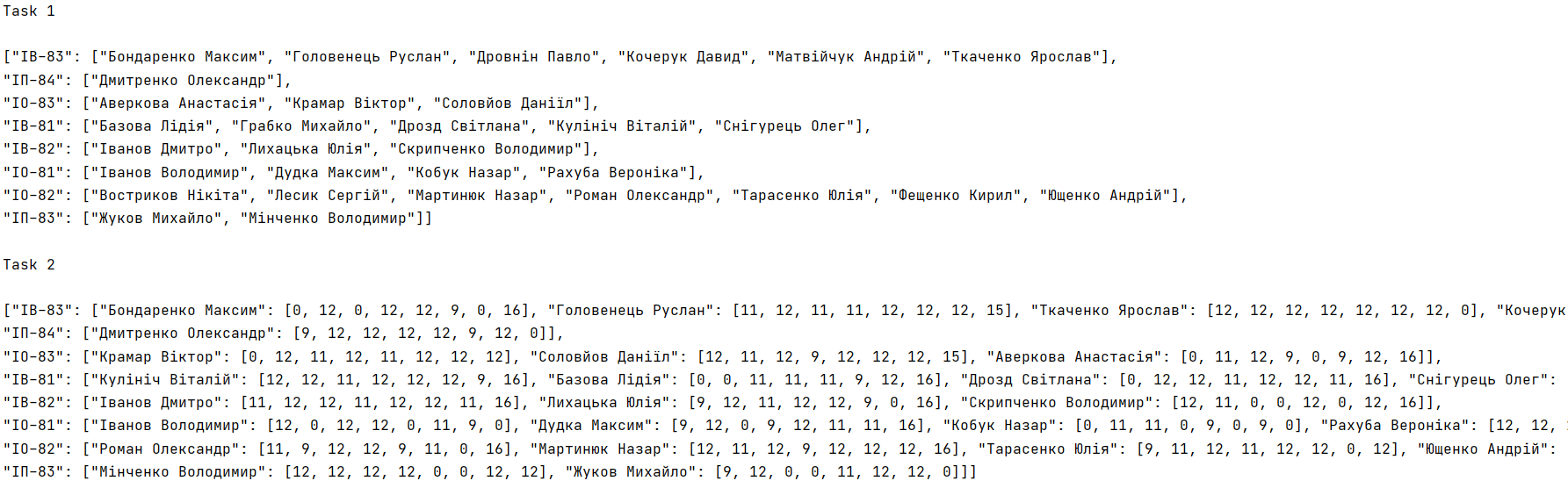
студент групи ІВ-82

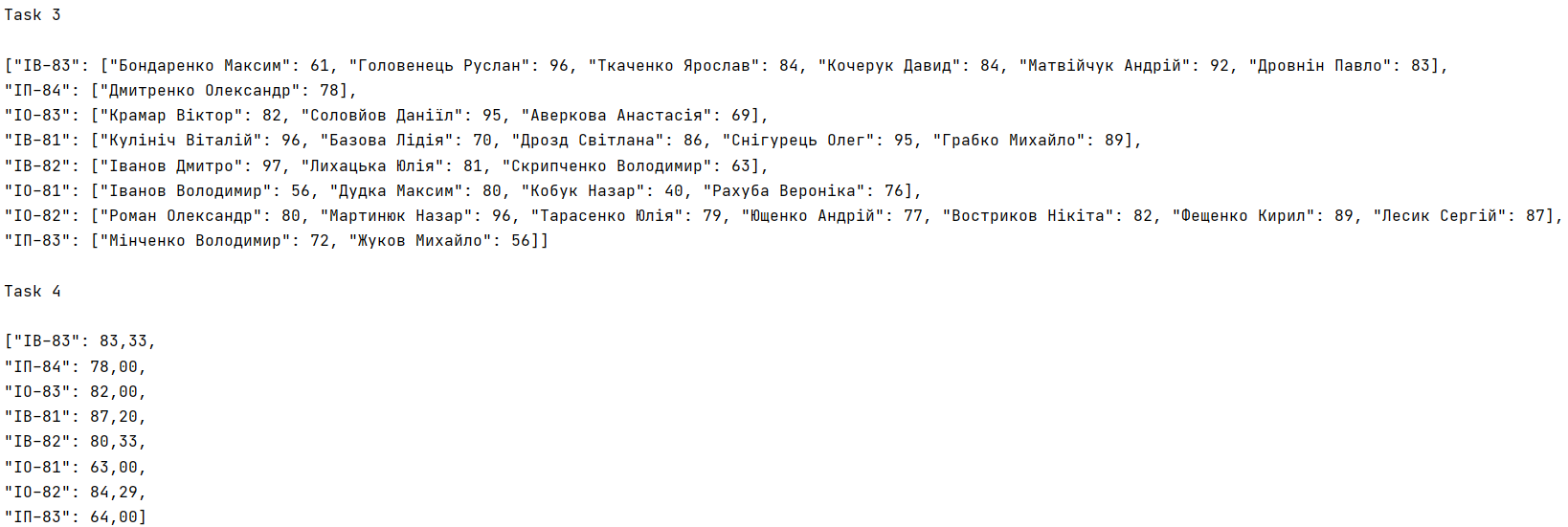
ЗК ІВ-8218

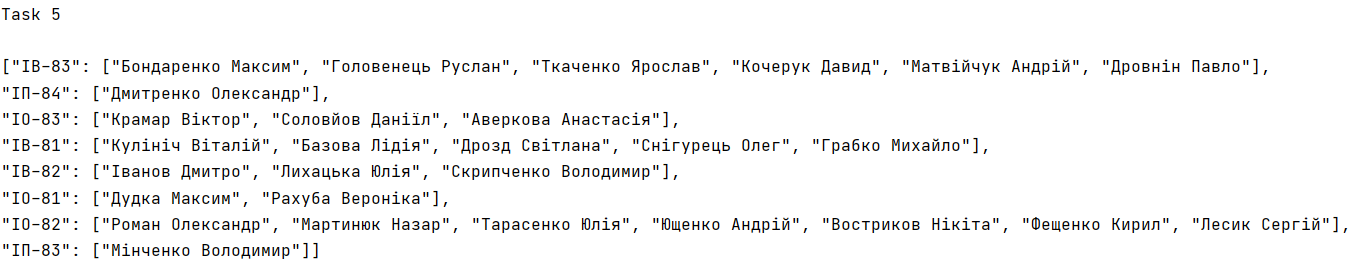
Пінчук Микита

Київ 2021

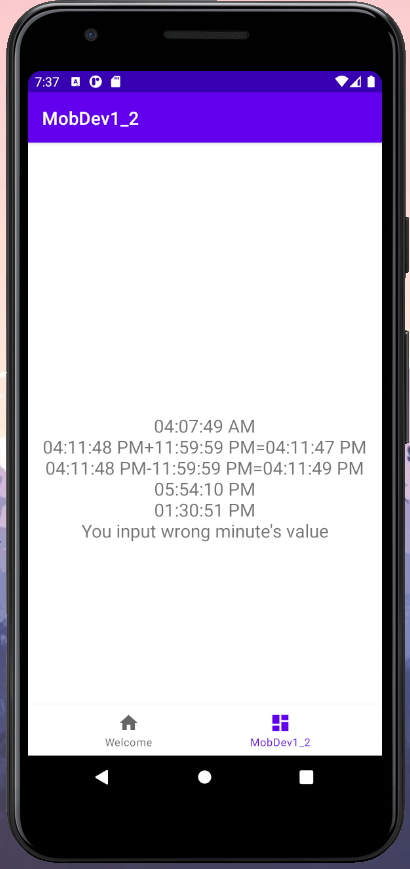
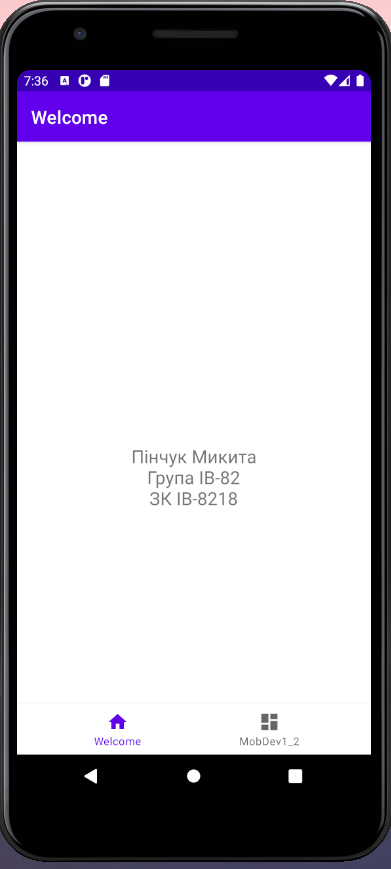
Скріншоти роботи програми частини 1







Скріншоти роботи додатка частини 2 (Варіант 8218 % 2 + 1 = 1)



Лістинг першого завдання

**Main.java**

package com.company;  
import java.util.HashMap;  
import java.util.Arrays;  
import java.util.Random;  
import java.lang.Math;  
  
public class Main {  
   
 public static HashMap<String, String[]> task1(String[] arr) {  
 HashMap<String, String[]> dict = new HashMap<>();  
  
 for (String string: arr) {  
 String[] student = string.split(" - ");  
 if (!dict.containsKey(student[1])){  
 dict.put(student[1], new String[]{student[0]});  
 }  
 else {  
 String[] old\_arr = dict.get(student[1]);  
 String[] new\_arr = new String[old\_arr.length + 1];  
 System.*arraycopy*(old\_arr, 0, new\_arr, 0, old\_arr.length);  
 new\_arr[new\_arr.length - 1] = student[0];  
 dict.replace(student[1], new\_arr);  
 }  
 }  
  
 for (String key: dict.keySet()) {  
 String[] value = dict.get(key);  
 Arrays.*sort*(value);  
 dict.put(key, value);  
 }  
  
 return dict;  
 }  
  
 public static String getTask1(HashMap<String, String[]> dict) {  
  
 System.*out*.println("Task 1\n");  
  
 System.*out*.print("[");  
 String[] key\_strings = new String[dict.keySet().toArray().length];  
 int main\_index = 0;  
 for (String key: dict.keySet()) {  
 String group\_string = String.*format*("\"%s\": [", key);  
 String[] strings = new String[dict.get(key).length];  
 int index = 0;  
 for (String student: dict.get(key)) {  
 strings[index] = String.*format*("\"%s\"", student);  
 index += 1;  
 }  
 key\_strings[main\_index] = group\_string + String.*join*(", ", strings) + "]";  
 main\_index += 1;  
 }  
 return String.*join*(", \n", key\_strings) + "]";  
  
 }  
  
 public static HashMap<String, HashMap<String, int[]>> task2(HashMap<String, String[]> dict) {  
 HashMap<String, HashMap<String, int[]>> result\_dict = new HashMap<>();  
 int[] points = new int[]{12, 12, 12, 12, 12, 12, 12, 16};  
 for (String key: dict.keySet()) {  
 HashMap<String, int[]> group\_dict = new HashMap<>();  
 for (String student: dict.get(key)) {  
 int[] student\_points = new int[points.length];  
 for (int i = 0; i < student\_points.length ; i++) {  
 student\_points[i] = *random\_value*(points[i]);  
 }  
 group\_dict.put(student, student\_points);  
 }  
 result\_dict.put(key, group\_dict);  
 }  
  
 return result\_dict;  
 }  
  
 public static String getTask2(HashMap<String, HashMap<String, int[]>> dict) {  
  
 System.*out*.println("\nTask 2\n");  
  
 System.*out*.print("[");  
 String[] key\_strings = new String[dict.keySet().toArray().length];  
 int main\_index = 0;  
 for (String key: dict.keySet()){  
 String group\_string = String.*format*("\"%s\": [", key);  
 String[] strings = new String[dict.get(key).keySet().toArray().length];  
 int index = 0;  
 for (HashMap.Entry<String, int[]> elem: dict.get(key).entrySet()) {  
 strings[index] = String.*format*("\"%s\": %s", elem.getKey(), Arrays.*toString*(elem.getValue()));  
 index += 1;  
 }  
 key\_strings[main\_index] = group\_string + String.*join*(", ", strings) + "]";  
 main\_index += 1;  
 }  
 return String.*join*(", \n", key\_strings) + "]";  
 }  
  
  
 public static HashMap<String, HashMap<String, Integer>> task3(HashMap<String, HashMap<String, int[]>> dict) {  
 HashMap<String, HashMap<String, Integer>> result\_dict = new HashMap<>();  
 for (String key: dict.keySet()){  
 HashMap<String, Integer> group\_dict = new HashMap<>();  
 for (String student: dict.get(key).keySet()){  
 int sum = 0;  
 for (int point: dict.get(key).get(student)){  
 sum += point;  
 }  
 group\_dict.put(student, sum);  
 }  
 result\_dict.put(key, group\_dict);  
 }  
  
 return result\_dict;  
 }  
  
 public static String getTask3(HashMap<String, HashMap<String, Integer>> dict) {  
  
 System.*out*.println("\nTask 3\n");  
  
 System.*out*.print("[");  
 String[] key\_strings = new String[dict.keySet().toArray().length];  
 int main\_index = 0;  
 for (String key: dict.keySet()){  
 String group\_string = String.*format*("\"%s\": [", key);  
 String[] strings = new String[dict.get(key).keySet().toArray().length];  
 int index = 0;  
 for (HashMap.Entry<String, Integer> elem: dict.get(key).entrySet()) {  
 strings[index] = String.*format*("\"%s\": %d", elem.getKey(), elem.getValue());  
 index += 1;  
 }  
 key\_strings[main\_index] = group\_string + String.*join*(", ", strings) + "]";  
 main\_index += 1;  
 }  
 return String.*join*(", \n", key\_strings) + "]";  
 }  
  
 public static HashMap<String, Float> task4(HashMap<String, HashMap<String, Integer>> dict) {  
 HashMap<String, Float> result\_dict = new HashMap<>();  
  
 for (String key: dict.keySet()){  
 float sum = 0, amount = 0;  
 for (String student: dict.get(key).keySet()){  
 sum += dict.get(key).get(student);  
 amount += 1;  
 }  
  
 result\_dict.put(key, sum / amount);  
 }  
 return result\_dict;  
 }  
  
 public static String getTask4(HashMap<String, Float> dict) {  
  
 System.*out*.println("\nTask 4\n");  
  
 System.*out*.print("[");  
 String[] key\_strings = new String[dict.keySet().toArray().length];  
 int main\_index = 0;  
 for (String key: dict.keySet()){  
 String group\_string = String.*format*("\"%s\": %.2f", key, dict.get(key));  
 key\_strings[main\_index] = group\_string;  
 main\_index += 1;  
 }  
 return String.*join*(", \n", key\_strings) + "]";  
 }  
  
 public static HashMap<String, String[]> task5(HashMap<String, HashMap<String, Integer>> dict) {  
 HashMap<String, String[]> result\_dict = new HashMap<>();  
  
 for (String key: dict.keySet()) {  
 String[] students = new String[]{};  
 for (String student: dict.get(key).keySet()){  
 if (dict.get(key).get(student) >= 60){  
 String[] new\_students = new String[students.length + 1];  
 System.*arraycopy*(students, 0, new\_students, 0, students.length);  
 new\_students[new\_students.length - 1] = student;  
 students = new\_students;  
 }  
  
 }  
 result\_dict.put(key, students);  
  
 }  
 return result\_dict;  
 }  
  
 public static String getTask5(HashMap<String, String[]> dict) {  
  
 System.*out*.println("\nTask 5\n");  
  
  
 System.*out*.print("[");  
 String[] key\_strings = new String[dict.keySet().toArray().length];  
 int main\_index = 0;  
 for (String key: dict.keySet()) {  
 String group\_string = String.*format*("\"%s\": [", key);  
 String[] strings = new String[dict.get(key).length];  
 int index = 0;  
 for (String student: dict.get(key)) {  
 strings[index] = String.*format*("\"%s\"", student);  
 index += 1;  
 }  
 key\_strings[main\_index] = group\_string + String.*join*(", ", strings) + "]";  
 main\_index += 1;  
 }  
 return String.*join*(", \n", key\_strings) + "]";  
 }  
  
  
 public static int random\_value(int max\_value) {  
 Random random = new Random();  
 switch (random.nextInt(6)) {  
 case 1:  
 return (int)(Math.*ceil*((float)(max\_value) \* 0.7));  
 case 2:  
 return (int)(Math.*ceil*((float)(max\_value) \* 0.9));  
 case 3:  
 case 4:  
 case 5:  
 return max\_value;  
 default:  
 return 0;  
  
 }  
 }  
   
  
 public static void main(String[] args) {  
 String main\_string = "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";  
 String[] default\_array = main\_string.split("; ");  
  
 HashMap<String, String[]> studentsByGroups = *task1*(default\_array);  
 System.*out*.println(*getTask1*(studentsByGroups));  
  
 HashMap<String, HashMap<String, int[]>> studentsByPoints = *task2*(studentsByGroups);  
 System.*out*.println(*getTask2*(studentsByPoints));  
  
 HashMap<String, HashMap<String, Integer>> sumPoints = *task3*(studentsByPoints);  
 System.*out*.println(*getTask3*(sumPoints));  
  
 HashMap<String, Float> groupAvg = *task4*(sumPoints);  
 System.*out*.println(*getTask4*(groupAvg));  
  
 HashMap<String, String[]> notDopka = *task5*(sumPoints);  
 System.*out*.println(*getTask5*(notDopka));  
 }  
}

Лістинг коду частини 2 (Варіант 8218 % 2 + 1 = 1)

**TimeMP.java**

package com.company;  
import java.util.Calendar;  
public class TimeMP {  
 private final int minutes;  
 private final int hours;  
 private final int seconds;  
  
 public TimeMP(){  
 this.hours = 0;  
 this.minutes = 0;  
 this.seconds = 0;  
 }  
  
 public TimeMP(int hours, int minutes, int seconds){  
 if (hours < 0 || hours > 23)  
 throw new IllegalArgumentException("You input wrong hour`s value");  
 else if (minutes < 0 || minutes > 59)  
 throw new IllegalArgumentException("You input wrong minute's value");  
 else if (seconds < 0 || seconds > 59)  
 throw new IllegalArgumentException("You input wrong second's value");  
 else{  
 this.hours = hours;  
 this.minutes = minutes;  
 this.seconds = seconds;  
 }}  
  
 public TimeMP(Calendar calendar){  
 this.hours = calendar.get(Calendar.*HOUR*);  
 this.minutes = calendar.get(Calendar.*MINUTE*);  
 this.seconds = calendar.get(Calendar.*SECOND*);  
 }  
  
 public String getTime(){  
 if (hours < 12)  
 return String.*format*("%02d:%02d:%02d", getHours(),getMinutes(),getSeconds()) + " AM";  
 else if (hours == 12){  
 return String.*format*("%02d:%02d:%02d", getHours(),getMinutes(),getSeconds()) + " PM";  
 }  
 else  
 return String.*format*("%02d:%02d:%02d", getHours() - 12,getMinutes(),getSeconds()) + " PM";  
 }  
  
 public int getSeconds(){  
 return this.seconds;  
 }  
  
 public int getHours(){  
 return this.hours;  
 }  
  
 public int getMinutes(){  
 return this.minutes;  
 }  
 public TimeMP plus2Time(TimeMP obj) {  
 int totalSecs = (this.hours + obj.getHours()) \* 3600 + (this.minutes + obj.getMinutes()) \* 60 + this.seconds + obj.getSeconds();  
 int hours = totalSecs / 60 / 60 % 24;  
 int minutes = totalSecs / 60 % 60;  
 int seconds = totalSecs % 60;  
 return new TimeMP(hours,minutes,seconds);  
 }  
  
  
 public static TimeMP plus2Time(TimeMP obj1, TimeMP obj2) {  
 int totalSecs = (obj1.getHours() + obj2.getHours()) \* 3600 + (obj1.getMinutes() + obj2.getMinutes()) \* 60 + obj1.getSeconds() + obj2.getSeconds();  
 int hours = totalSecs / 60 / 60 % 24;  
 int minutes = totalSecs / 60 % 60;  
 int seconds = totalSecs % 60;  
 return new TimeMP(hours,minutes,seconds);  
 }  
  
  
  
 public TimeMP minus2Time(TimeMP obj) {  
 int totalSecs = (this.hours - obj.getHours()) \* 3600 + (this.minutes - obj.getMinutes()) \* 60 + this.seconds - obj.getSeconds();  
 if (totalSecs < 0){  
 int newTotalSecs = 24 \* 3600 + totalSecs;  
 int hours = newTotalSecs / 60 / 60 % 24;  
 int minutes = newTotalSecs / 60 % 60;  
 int seconds = newTotalSecs % 60;  
 return new TimeMP(hours, minutes, seconds);  
 }  
 else{  
 int hours = totalSecs / 60 / 60 % 24;  
 int minutes = totalSecs / 60 % 60;  
 int seconds = totalSecs % 60;  
 return new TimeMP(hours, minutes, seconds);  
 }  
 }  
  
 public static TimeMP minus2Time(TimeMP obj1, TimeMP obj2) {  
 int totalSecs = (obj1.getHours() - obj2.getHours()) \* 3600 + (obj1.getMinutes() - obj2.getMinutes()) \* 60 + obj1.getSeconds() - obj2.getSeconds();  
 if (totalSecs < 0){  
 int newTotalSecs = 24 \* 3600 + totalSecs;  
 int hours = newTotalSecs / 60 / 60 % 24;  
 int minutes = newTotalSecs / 60 % 60;  
 int seconds = newTotalSecs % 60;  
 return new TimeMP(hours, minutes, seconds);  
 }  
 else{  
 int hours = totalSecs / 60 / 60 % 24;  
 int minutes = totalSecs / 60 % 60;  
 int seconds = totalSecs % 60;  
 return new TimeMP(hours, minutes, seconds);  
 }  
 }  
}

**MainActivity.java**

**package** ua.kpi.comsys.iv8218;  
  
**import** android.os.Bundle;  
  
**import** com.google.android.material.bottomnavigation.BottomNavigationView;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
**import** androidx.navigation.NavController;  
**import** androidx.navigation.Navigation;  
**import** androidx.navigation.ui.AppBarConfiguration;  
**import** androidx.navigation.ui.NavigationUI;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 BottomNavigationView navView = findViewById(R.id.***nav\_view***);  
 *// Passing each menu ID as a set of Ids because each  
 // menu should be considered as top level destinations.* AppBarConfiguration appBarConfiguration = **new** AppBarConfiguration.Builder(  
 R.id.***navigation\_home***, R.id.***navigation\_dashboard***)  
 .build();  
 NavController navController = Navigation.*findNavController*(**this**, R.id.***nav\_host\_fragment***);  
 NavigationUI.*setupActionBarWithNavController*(**this**, navController, appBarConfiguration);  
 NavigationUI.*setupWithNavController*(navView, navController);  
 }  
  
}

**HomeViewModel.java**

**package** ua.kpi.comsys.iv8218.ui.home;  
  
**import** androidx.lifecycle.LiveData;  
**import** androidx.lifecycle.MutableLiveData;  
**import** androidx.lifecycle.ViewModel;  
  
**public class** HomeViewModel **extends** ViewModel {  
  
 **private** MutableLiveData<String> **mText**;  
  
 **public** HomeViewModel() {  
 **mText** = **new** MutableLiveData<>();  
 **mText**.setValue(**"Пінчук Микита\nГрупа ІВ-82\nЗК ІВ-8218"**);  
 }  
  
 **public** LiveData<String> getText() {  
 **return mText**;  
 }  
}

**DashBoardViewModel.java**

**package** ua.kpi.comsys.iv8218.ui.dashboard;  
  
**import** androidx.lifecycle.LiveData;  
**import** androidx.lifecycle.MutableLiveData;  
**import** androidx.lifecycle.ViewModel;  
  
**import** java.util.Calendar;  
**import** java.util.GregorianCalendar;  
  
**import** ua.kpi.comsys.iv8218.lab1\_2.TimeMP;  
  
**public class** DashboardViewModel **extends** ViewModel {  
  
 **private** MutableLiveData<String> **mText**;  
  
 **public** DashboardViewModel() {  
 String error = **""**;  
 **try**{  
 TimeMP lastTest = **new** TimeMP(23, 62, 60);  
 }**catch** (IllegalArgumentException e){  
 error = e.getMessage();  
 }  
  
 TimeMP test1 = **new** TimeMP(4, 7, 49);  
 TimeMP test2 = **new** TimeMP(16, 11, 48);  
 TimeMP test2\_1 = **new** TimeMP(23, 59, 59);  
 **mText** = **new** MutableLiveData<>();  
 Calendar calendar = **new** GregorianCalendar(2001, 9, 11);  
 calendar.set(Calendar.***HOUR***, 13);  
 calendar.set(Calendar.***MINUTE***, 20);  
 calendar.set(Calendar.***SECOND***, 51);  
  
  
 **mText**.setValue(test1.getTime() +  
 **"\n"** + test2.getTime() + **"+"** + test2\_1.getTime() + **"="** + test2.plus2Time(test2\_1).getTime() +  
 **"\n"** + test2.getTime() + **"-"** + test2\_1.getTime() + **"="** + test2.minus2Time(test2\_1).getTime() +  
 **"\n"** + TimeMP.*minus2Time*(**new** TimeMP(), **new** TimeMP(6, 5, 50)).getTime() +  
 **"\n"** + TimeMP.*plus2Time*(**new** TimeMP(12, 10, 0), **new** TimeMP(calendar)).getTime()+  
 **"\n"** + error);  
  
 }  
 **public** LiveData<String> getText() {  
 **return mText**;  
 }  
  
}

**Висновок**

В даній лабораторній роботі було розроблено мобільний додаток, який показує «читабельний» для людини час, та арифметичні операції над часом. Також, було виконане завдання пов’язане з використанням HashMap у мові Java.

Обидві програми працюють коректно. Кінцева мета досягнута.