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-81

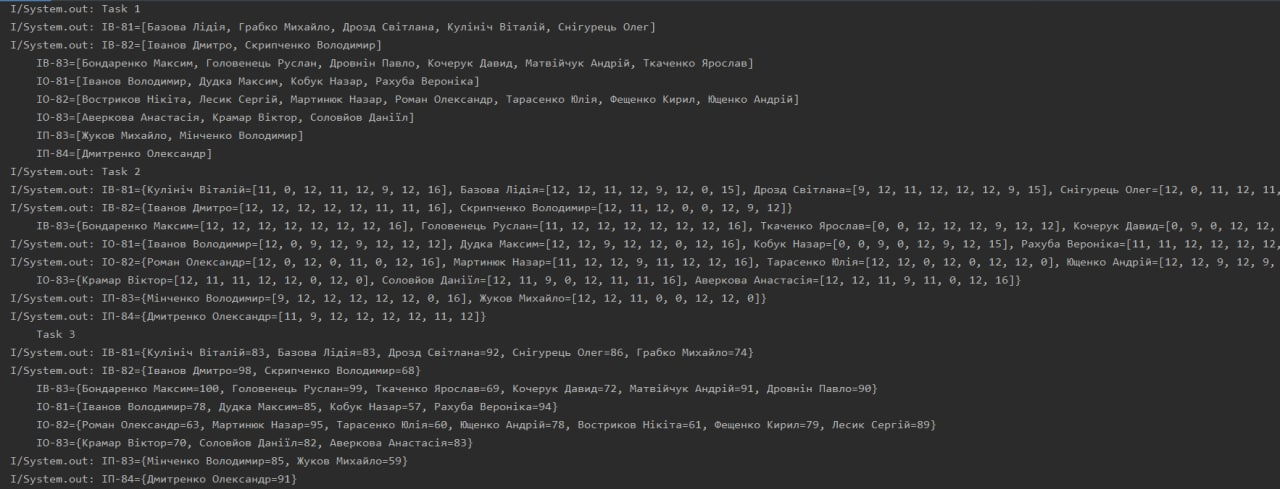
ЗК ІО-8114

Кобук Назар

Київ 2021

**Варіант № 1**

**Скріншот роботи додатка**



**Лістинг коду**

## ****Contents.java****

package ua.kpi.comsys.io8114.ui.home;

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";

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);

return studentsGroups;

}

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;

}

}

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;

}

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;

}

public HashMap<String, Float> showAvgGradesInGroups(HashMap<String,

HashMap<String, Integer>> sumGrades) {

HashMap<String, Float> middle = 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();

middle.put(group.getKey(), avgGrade);

}

middle.entrySet()

.stream()

.sorted(Map.Entry.<String, Float>comparingByKey())

.forEach(System.out::println);

return middle;

}

public HashMap<String, ArrayList<String>> showBestInGroups(HashMap<String,

HashMap<String, Integer>> sumGrades) {

HashMap<String, ArrayList<String>> top = 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());

}

top.put(group.getKey(), bestStudents);

}

top.entrySet()

.stream()

.sorted(Map.Entry.<String, ArrayList<String>>comparingByKey())

.forEach(System.out::println);

return top;

}

}

## ****TimeNK.java****

package ua.kpi.comsys.io8114.ui.home;

import androidx.annotation.NonNull;

import java.util.Date;

public class TimeNK {

int hours;

int minutes;

int seconds;

public TimeNK() {

timeSet(0, 0, 0);

}

public TimeNK(Date date) {

this.hours = date.getHours();

this.minutes = date.getMinutes();

this.seconds = date.getSeconds();

}

public TimeNK(int hours, int minutes, int seconds) {

if ((hours >= 0 && hours <=23) && (minutes >= 0 && minutes <= 59) &&

(seconds >= 0 && seconds <= 59)) {

timeSet(hours, minutes, seconds);

} else {

System.out.println("Can`t create");

}

}

public TimeNK sumTime(TimeNK time) {

TimeNK resultTime = new TimeNK(hours, minutes, seconds);

if (resultTime.hours + time.hours >= 24)

resultTime.hours = (resultTime.hours + time.hours) % 24;

else

resultTime.hours += time.hours;

if (resultTime.minutes + time.minutes >= 60) {

resultTime.minutes = (resultTime.minutes + time.minutes) % 60;

if (resultTime.hours != 23)

resultTime.hours ++;

else

resultTime.hours = 0;

} else resultTime.minutes += time.minutes;

if (resultTime.seconds + time.seconds >= 60) {

resultTime.seconds = (resultTime.seconds + time.seconds) % 60;

if (resultTime.minutes != 59)

resultTime.minutes ++;

else

resultTime.minutes = 0;

if (resultTime.hours != 23)

resultTime.hours ++;

else

resultTime.hours = 0;

} else resultTime.seconds += time.seconds;

return resultTime;

}

public static TimeNK sumTwoTime(TimeNK timeOne, TimeNK timeTwo) {

TimeNK resultTime = new TimeNK(timeOne.hours, timeOne.minutes, timeOne.seconds);

if (timeOne.hours + timeTwo.hours >= 24)

resultTime.hours = (timeOne.hours + timeTwo.hours) % 24;

else

resultTime.hours = timeOne.hours + timeTwo.hours;

if (timeOne.minutes + timeTwo.minutes >= 60) {

resultTime.minutes = (timeOne.minutes + timeTwo.minutes) % 60;

if (resultTime.hours != 23)

resultTime.hours ++;

else

resultTime.hours = 0;

} else resultTime.minutes = timeOne.minutes + timeTwo.minutes;

if (timeOne.seconds + timeTwo.seconds >= 60) {

resultTime.seconds = (timeOne.seconds + timeTwo.seconds) % 60;

if (resultTime.minutes != 59)

resultTime.minutes ++;

else

resultTime.minutes = 0;

if (resultTime.hours != 23)

resultTime.hours ++;

else

resultTime.hours = 0;

} else resultTime.seconds = timeOne.seconds + timeTwo.seconds;

return resultTime;

}

public TimeNK minusTime(TimeNK time) {

TimeNK resultTime = new TimeNK(hours, minutes, seconds);

if (resultTime.hours >= time.hours)

resultTime.hours -= time.hours;

else {

if (time.hours < 24)

resultTime.hours = 24 - (time.hours - resultTime.hours);

else if (time.hours > 24)

resultTime.hours = 24 - ((time.hours - resultTime.hours) % 24);

}

if (resultTime.minutes >= time.minutes) {

resultTime.minutes -= time.minutes;

} else {

if (time.minutes < 60)

resultTime.minutes = 60 - (time.minutes - resultTime.minutes);

else if (time.minutes > 60)

resultTime.minutes = 60 - ((time.minutes - resultTime.minutes) % 60);

if (resultTime.hours != 0)

resultTime.hours --;

else

resultTime.hours = 23;

}

if (resultTime.seconds >= time.seconds) {

resultTime.seconds -= time.seconds;

} else {

if (time.seconds < 60)

resultTime.seconds = 60 - (time.seconds - resultTime.seconds);

else if (time.seconds > 60)

resultTime.seconds = 60 - ((time.seconds - resultTime.seconds) % 60);

if (resultTime.minutes != 0)

resultTime.minutes --;

else {

resultTime.minutes = 59;

if (resultTime.hours != 0)

resultTime.hours--;

else

resultTime.hours = 23;

}

}

return resultTime;

}

public static TimeNK minusTwoTime(TimeNK timeOne, TimeNK timeTwo) {

TimeNK resultTime = new TimeNK(timeOne.hours, timeOne.minutes, timeOne.seconds);

if (timeOne.hours >= timeTwo.hours)

resultTime.hours = timeOne.hours - timeTwo.hours;

else {

if (timeTwo.hours < 24)

resultTime.hours = 24 - (timeTwo.hours - timeOne.hours);

else if (timeTwo.hours > 24)

resultTime.hours = 24 - ((timeTwo.hours - timeOne.hours) % 24);

}

if (timeOne.minutes >= timeTwo.minutes) {

resultTime.minutes = timeOne.minutes - timeTwo.minutes;

} else {

if (timeTwo.minutes < 60)

resultTime.minutes = 60 - (timeTwo.minutes - timeOne.minutes);

else if (timeTwo.minutes > 60)

resultTime.minutes = 60 - ((timeTwo.minutes - timeOne.minutes) % 60);

if (resultTime.hours != 0)

resultTime.hours --;

else

resultTime.hours = 23;

}

if (timeOne.seconds >= timeTwo.seconds) {

resultTime.seconds = timeOne.seconds - timeTwo.seconds;

} else {

if (timeTwo.seconds > 60)

resultTime.seconds = 60 - ((timeTwo.seconds - timeOne.seconds) % 60);

else if (timeTwo.seconds < 60)

resultTime.seconds = 60 - (timeTwo.seconds - timeOne.seconds);

if (resultTime.minutes != 0)

resultTime.minutes --;

else {

resultTime.minutes = 59;

if (resultTime.hours != 0)

resultTime.hours--;

else

resultTime.hours = 23;

}

}

return resultTime;

}

private void timeSet(int hours, int minutes, int seconds) {

this.hours = hours;

this.minutes = minutes;

this.seconds = seconds;

}

@NonNull

@Override

public String toString() {

String hours = "";

String minutes = "";

String seconds = "";

String partOfTheDay;

if (this.hours >= 12) {

if (this.hours - 12 < 10 && this.hours != 12) {

hours += "0";

hours += this.hours - 12;

} else if (this.hours == 12) {

hours += 12;

} else {

hours += this.hours - 12;

}

partOfTheDay = "PM";

} else {

if (this.hours < 10 && this.hours != 0) {

hours += "0";

hours += this.hours;

} else if (this.hours == 0) {

hours += 12;

} else {

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;

}

}

**Висновок**

Отже в результаті виконання лаборатоної я оволодів основам роботи з мовою Java в середовищі Android studio. Було налогоджено взаємодію коду за варіантом з розробленим мобільним додатком. Було створено класи **Contents та TimeNK в яких виконано завдання за варіантом.**