НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №2

з дисципліни

“Програмування мобільних систем”

Виконав:

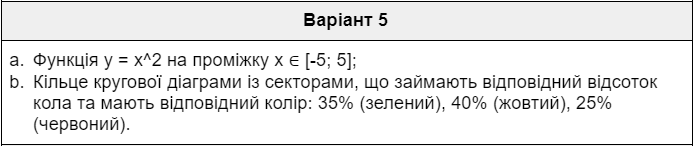
студент групи ІВ-82

ЗК ІВ-8224

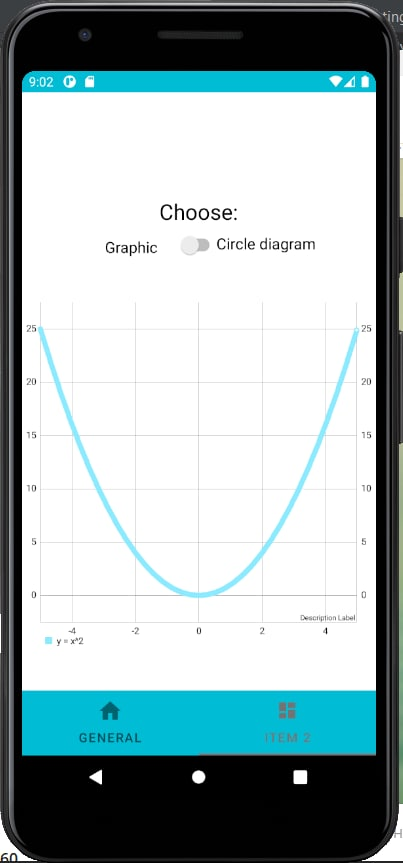
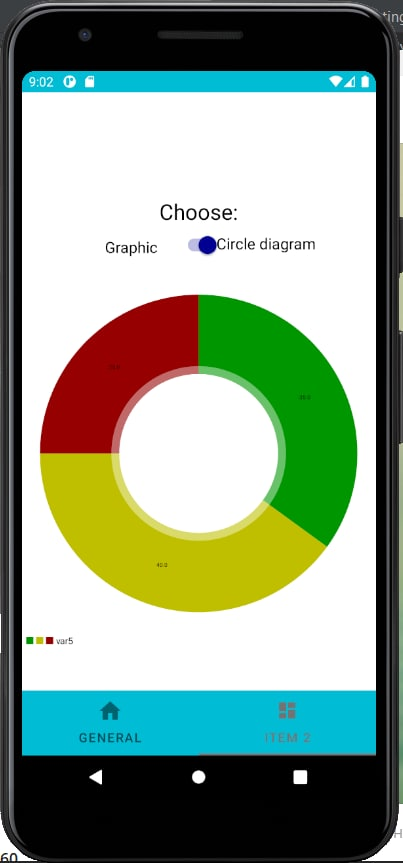
Старшинов Кирило

Київ 2021

**Варіант 8224 % 6 + 1 = 5**



Скріншоти роботи додатка

1)  2) 

Лістинг коду

**Lab2.java**

package ua.kpi.comsys.iv8224.third.lab2;

import ua.kpi.comsys.iv8224.third.R;

import android.graphics.Color;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Switch;

import androidx.annotation.NonNull;

import androidx.core.content.ContextCompat;

import androidx.fragment.app.Fragment;

import com.github.mikephil.charting.charts.LineChart;

import com.github.mikephil.charting.charts.PieChart;

import com.github.mikephil.charting.components.AxisBase;

import com.github.mikephil.charting.data.Entry;

import com.github.mikephil.charting.data.LineData;

import com.github.mikephil.charting.data.LineDataSet;

import com.github.mikephil.charting.data.PieData;

import com.github.mikephil.charting.data.PieDataSet;

import com.github.mikephil.charting.data.PieEntry;

import com.github.mikephil.charting.interfaces.datasets.ILineDataSet;

import com.github.mikephil.charting.components.XAxis;

import com.github.mikephil.charting.components.YAxis;

import com.github.mikephil.charting.components.Description;

import androidx.annotation.Nullable;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import java.util.SortedMap;

import java.util.TreeMap;

public class Lab2 extends Fragment {

private LineChart line;

private PieChart pieChart;

public View onCreateView(@NonNull LayoutInflater inflater,

ViewGroup container, Bundle savedInstanceState) {

View root = inflater.inflate(R.layout.*lab2*, container, false);

line = root.findViewById(R.id.*chart\_line*);

pieChart = root.findViewById(R.id.*chart\_pie*);

List<Entry> chartData\_main = new ArrayList<>();

List<Entry> chartData\_X = new ArrayList<>();

List<Entry> chartData\_Y = new ArrayList<>();

setGraphicsOnChart(line, chartData\_main, chartData\_X, chartData\_Y);

Switch Switch = root.findViewById(R.id.*graphSwitch*);

Switch.setOnCheckedChangeListener((buttonView, isChecked) -> {

if (isChecked) {

line.setVisibility(View.*GONE*);

pieChart.setVisibility(View.*VISIBLE*);

} else {

pieChart.setVisibility(View.*GONE*);

line.setVisibility(View.*VISIBLE*);

}

});

updateGraphic(root, line, chartData\_main, chartData\_X, chartData\_Y);

updateDiagram(root, pieChart);

return root;

}

private void updateDiagram(View v, PieChart pieChart) {

pieChart.getDescription().setEnabled(false);

pieChart.setRotationEnabled(false);

pieChart.setRotationAngle(0);

pieChart.setHighlightPerTapEnabled(false);

pieChart.getLegend().setCustom(new ArrayList<>());

ArrayList<PieEntry> pieEntries = new ArrayList<>();

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

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

typeAmountMap.add(40);

colors.add(Color.*YELLOW*);

typeAmountMap.add(35);

colors.add(Color.*GREEN*);

typeAmountMap.add(25);

colors.add(Color.*RED*);

for(int i=0; i<typeAmountMap.size(); i++){

pieEntries.add(new PieEntry(typeAmountMap.get(i).floatValue(), ""));

}

PieDataSet pieDataSet = new PieDataSet(pieEntries, "");

pieDataSet.setValueTextSize(0f);

pieDataSet.setColors(colors);

PieData pieData = new PieData(pieDataSet);

pieData.setValueTextColor(Color.*WHITE*);

pieChart.setData(pieData);

pieChart.invalidate();

}

private void updateGraphic(View v, LineChart lineChart,

List<Entry> chartData\_main,

List<Entry> chartData\_X,

List<Entry> chartData\_Y){

SortedMap<Double, Double> graph\_main = new TreeMap<>();

SortedMap<Double, Double> graph\_X = new TreeMap<>();

SortedMap<Double, Double> graph\_Y = new TreeMap<>();

lineChart.getXAxis().setAxisMinimum(-5);

lineChart.getAxisLeft().setAxisMinimum(-1);

lineChart.getAxisRight().setAxisMinimum(-1);

lineChart.getXAxis().setAxisMaximum(5);

lineChart.getAxisLeft().setAxisMaximum(25);

lineChart.getAxisRight().setAxisMaximum(25);

lineChart.getAxisLeft().setCenterAxisLabels(true);

for (double i = -5; i <= 5; i+=0.05) {

if (i != 1 & i !=0) {

drawGraph(lineChart, graph\_main, chartData\_main, 0, i, i\*i);

}

}

}

private void setGraphicsOnChart(LineChart line,

List<Entry> chartData\_main,

List<Entry> chartData\_X,

List<Entry> chartData\_Y){

line.getDescription().setEnabled(false);

line.getLegend().setCustom(new ArrayList<>());

setAxisParams(line.getXAxis());

setAxisParams(line.getAxisLeft());

setAxisParams(line.getAxisRight());

XAxis xAxis = line.getXAxis();

xAxis.setDrawGridLines(true);

xAxis.setDrawAxisLine(true);

xAxis.setPosition(XAxis.XAxisPosition.*BOTTOM*);

YAxis yAxisRight = line.getAxisRight();

yAxisRight.setEnabled(true);

YAxis yAxisLeft = line.getAxisLeft();

yAxisLeft.setDrawAxisLine(true);

yAxisLeft.setDrawGridLines(true);

yAxisLeft.setDrawZeroLine(true);

line.getXAxis().setAxisMinimum(1f);

line.getXAxis().setAxisMaximum(5f);

LineDataSet chartDataSet\_main = new LineDataSet(chartData\_main, "Function");

chartDataSet\_main.setColor(ContextCompat.*getColor*(getContext(), R.color.*purple\_500*));

LineDataSet chartDataSet\_X = new LineDataSet(chartData\_X, "X");

chartDataSet\_X.setColor(Color.*GRAY*);

LineDataSet chartDataSet\_Y = new LineDataSet(chartData\_Y, "Y");

chartDataSet\_Y.setColor(Color.*GRAY*);

setDataSetParams(chartDataSet\_main, 1.5f, 5f, false);

setDataSetParams(chartDataSet\_X, 1.5f, 5f, false);

setDataSetParams(chartDataSet\_Y, 1.5f, 5f, false);

List<ILineDataSet> charDataSets = new ArrayList<>();

charDataSets.add(chartDataSet\_main);

charDataSets.add(chartDataSet\_X);

charDataSets.add(chartDataSet\_Y);

ArrayList<Entry> dataSet = new ArrayList<>();

ArrayList<ILineDataSet> iLineDataSets = new ArrayList<>();

iLineDataSets.add(new LineDataSet(dataSet, "y = x^2"));

LineData lineData = new LineData(charDataSets);

line.setData(lineData);

Description desc = new Description();

line.setDescription(desc);

line.setData(lineData);

line.invalidate();

}

private void drawGraph(LineChart l, Map<Double, Double> m, List data,

int index, double key, double value){

m.put(key, value);

data.clear();

for (double v: m.keySet()) {

data.add(new Entry((float)v, m.get(v).floatValue()));

}

LineDataSet set = (LineDataSet)l.getData().getDataSetByIndex(index);

set.setValues(data);

set.notifyDataSetChanged();

l.getData().notifyDataChanged();

l.notifyDataSetChanged();

l.invalidate();

}

private void setAxisParams(AxisBase axis){

axis.setDrawLabels(true);

axis.setDrawAxisLine(true);

axis.setDrawGridLinesBehindData(true);

axis.setDrawLimitLinesBehindData(true);

axis.setDrawGridLines(true);

}

private void setDataSetParams(LineDataSet dataSet,

float lineWidth, float circleRadius, boolean drawCircle){

dataSet.setLineWidth(lineWidth);

dataSet.setCircleRadius(circleRadius);

dataSet.setDrawCircleHole(true);

dataSet.setFormLineWidth(1f);

dataSet.setFormSize(15.f);

dataSet.setValueTextSize(9f);

dataSet.setDrawValues(false);

dataSet.setDrawCircles(drawCircle);

}

}