Міністерство освіти і науки України  
Національний технічний університет України

«Київський політехнічний інститут ім. І. Сікорського»

Кафедра інженерії програмного забезпечення в енергетиці

Лабораторна робота № 3

Варіант 5

з курсу: «Розробка програмного забезпечення мобільних пристроїв»

**Виконала:**  
студентка 4-го курсу,  
групи ТВ-11

Барабаш Маріна Володимирівна

Посилання на GitHub репозиторій:

https://github.com/Aylosteraa/PW\_TV-11\_Barabash\_Marina\_Volodymyrivna.git

**Перевірив:**

Недашківський О.Л.

Київ 2024/2025

Лабораторна робота № 3

**Завдання:**

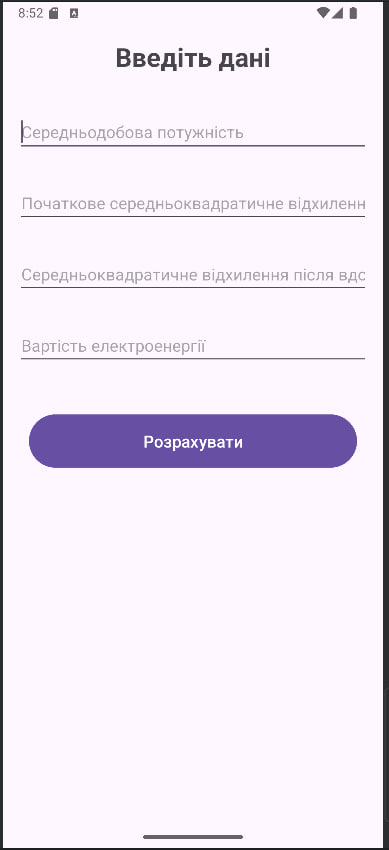
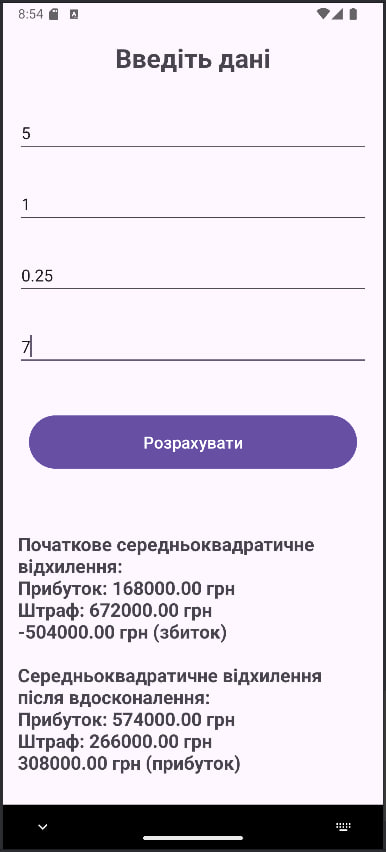
Створіть мобільний калькулятор розрахунку прибутку від сонячних електростанцій з встановленою системою прогнозування сонячної потужності

**Код програми:**

Основний код програми має вигляд:

package com.example.myapplication  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
import kotlin.math.round  
import kotlin.math.*PI*import kotlin.math.exp  
import kotlin.math.sqrt  
import kotlin.math.pow  
class MainActivity : AppCompatActivity() {  
 private lateinit var result: TextView  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 val btn = findViewById<Button>(R.id.*CalculateButton*)  
 val p = findViewById<EditText>(R.id.*Input\_P*)  
 val sigma1 = findViewById<EditText>(R.id.*Input\_Sigma1*)  
 val sigma2 = findViewById<EditText>(R.id.*Input\_Sigma2*)  
 val cost = findViewById<EditText>(R.id.*Input\_Cost*)  
 result = findViewById(R.id.*ResultTextView*)  
 btn.setOnClickListener **{** val p\_double = p.*text*.toString().*toDoubleOrNull*()  
 val sigma1\_double = sigma1.*text*.toString().*toDoubleOrNull*()  
 val sigma2\_double = sigma2.*text*.toString().*toDoubleOrNull*()  
 val cost\_double = cost.*text*.toString().*toDoubleOrNull*()  
 if (p\_double != null && sigma1\_double != null && sigma2\_double != null && cost\_double != null) {  
 printResult(p\_double, cost\_double, sigma1\_double, sigma2\_double)  
 } else {  
 result.*text* = "Введіть усі значення."  
 }  
 **}** }  
 private fun calculate(power: Double, cost: Double, sigma: Double): Triple<Double, Double, Double> {  
 val delta = power \* 0.05  
 val b1 = power - delta  
 val b2 = power + delta  
 val step = 0.001  
 var energyShare = 0.0  
 for (p in *generateSequence*(b1) **{ it** + step **}**.*takeWhile* **{ it** < b2 **}**) {  
 val pd = (1 / (sigma \* *sqrt*(2 \* *PI*))) \* *exp*(-((p - power).*pow*(2)) / (2 \* sigma.*pow*(2)))  
 energyShare += pd \* step  
 }  
 val energyWithoutImbalance = *round*(power \* 24 \* energyShare)  
 val profit = energyWithoutImbalance \* cost \* 1000  
 val energyWithImbalance = *round*(power \* 24 \* (1 - energyShare))  
 val penalty = energyWithImbalance \* cost \* 1000  
 return Triple(profit, penalty, profit - penalty)  
 }  
 private fun printResult(power: Double, cost: Double, sigma1: Double, sigma2: Double) {  
 val sigma1Result = calculate(power, cost, sigma1)  
 val sigma2Result = calculate(power, cost, sigma2)  
 result.*text* = """  
 Початкове середньоквадратичне відхилення:  
 Прибуток: ${"%.2f".*format*(sigma1Result.first)} грн  
 Штраф: ${"%.2f".*format*(sigma1Result.second)} грн  
 ${"%.2f".*format*(sigma1Result.third)} грн (${if (sigma1Result.third >= 0) "прибуток" else "збиток"})  
   
 Середньоквадратичне відхилення після вдосконалення:  
 Прибуток: ${"%.2f".*format*(sigma2Result.first)} грн  
 Штраф: ${"%.2f".*format*(sigma2Result.second)} грн  
 ${"%.2f".*format*(sigma2Result.third)} грн (${if (sigma2Result.third >= 0) "прибуток" else "збиток"})  
 """.*trimIndent*()  
 }  
}

**Результат виконання роботи:**

**Висновок:**

У лабораторній роботі 3 було розроблено програму для мобільного застосунку у вигляді калькулятору для обрахунку прибутків та збитків від сонячних електростанцій.