Міністерство освіти і науки України  
Національний технічний університет України

«Київський політехнічний інститут ім. І. Сікорського»

Кафедра інженерії програмного забезпечення в енергетиці

Лабораторна робота № 1

Варіант 11(1)

з курсу: «Розробка програмного забезпечення мобільних пристроїв»

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

Гундяк Валерія Русланівна

Посилання на GitHub репозиторій: https://github.com/pojuipolynu/PW\_TV-11\_Hundiak\_Valeriia\_Ruslanivna.git

**Перевірив:**

Недашківський О.Л.

Київ 2024/2025

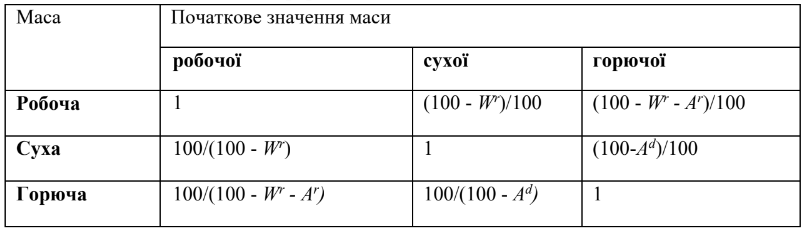
Лабораторна робота № 1

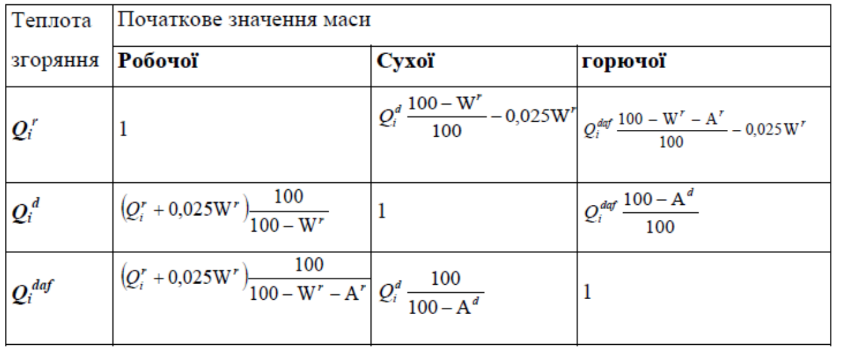
**Завдання:**

1. Написати мобільний калькулятор для розрахунку складу сухої та горючої маси палива та нижчої теплоти згоряння для робочої, сухої та горючої маси за заданим складом компонентів палива, що задаються у вигляді значень окремих компонентів типу: HP, %; CP, %; SP, %; NP, %; OP, %; WP, %; AP, % (див. табл. 1.3.).

2. Написати мобільний калькулятор для перерахунку елементарного складу та нижчої теплоти згоряння мазуту на робочу масу для складу горючої маси мазуту, що задається наступними параметрами: вуглець, %; водень, %; кисень, %; сірка, %; нижча теплота згоряння горючої маси мазуту, МДж/кг; вологість робочої маси палива, %; зольність сухої маси, %; вміст ванадію (V), мг/кг.

**Хід виконання:**

Основні формули для роботи калькулятора були задані у практичному матеріалі  




**Код програми:**

Основний код програми має вигляд:

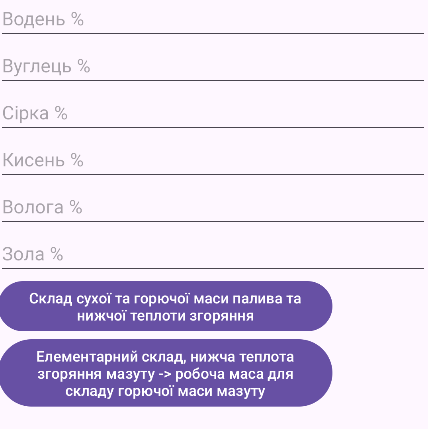
package com.example.physics\_app  
  
import android.os.Bundle  
import android.view.View  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.view.ViewCompat  
import androidx.core.view.WindowInsetsCompat  
import java.math.BigDecimal  
import java.math.RoundingMode  
  
class MainActivity : AppCompatActivity() {  
  
 //Декларація компонентів інтерфейсу  
 private lateinit var inputFields: List<EditText>  
 private lateinit var buttonComposition: Button  
 private lateinit var buttonRecount: Button  
 private lateinit var buttonCalculate: Button  
 private lateinit var buttonChangeInputs: Button  
 private lateinit var resultTextView: TextView  
 private lateinit var inputNP: EditText  
 private lateinit var inputQ: EditText  
 private lateinit var inputVenadii: EditText  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_main*)  
  
 // Ініціалізація загальних для обох розрахунків комірок вводу данних  
 inputFields = *listOf*(  
 findViewById(R.id.*input\_hp*),  
 findViewById(R.id.*input\_cp*),  
 findViewById(R.id.*input\_sp*),  
 findViewById(R.id.*input\_op*),  
 findViewById(R.id.*input\_wp*),  
 findViewById(R.id.*input\_ap*)  
 )  
  
 // Ініціалізація кнопок та виводу результату  
 buttonComposition = findViewById(R.id.*button\_composition*)  
 buttonRecount = findViewById(R.id.*button\_recount*)  
 buttonCalculate = findViewById(R.id.*button\_calculate*)  
 buttonChangeInputs = findViewById(R.id.*button\_change\_inputs*)  
 resultTextView = findViewById(R.id.*result*)  
  
 // Ініціалізація унікальних для кожного з розрахунків комірок вводу данних  
 inputNP = findViewById(R.id.*input\_np*)  
 inputQ = findViewById(R.id.*input\_q*)  
 inputVenadii = findViewById(R.id.*input\_venadii*)  
  
 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.*main*)) **{** v, insets **->** val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())  
 v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)  
 insets  
 **}** //Ініціалізація ходу роботи кнопки розрахунку складу сухої та горючої маси палива та нижчої теплоти згоряння  
 buttonComposition.setOnClickListener **{** showCompositionInputs()  
 **}** //Ініціалізація ходу роботи кнопки перерахунку елементарного складу, нижчої теплоти згоряння мазуту в робочу масу для  
 //складу горючої маси мазуту  
 buttonRecount.setOnClickListener **{** showRecountInputs()  
 **}** //Ініціалізація самого розрахунку  
 buttonCalculate.setOnClickListener **{** //Перевірка заповненості полів  
 if (!areInputsValid()) {  
 resultTextView.*text* = "Усі поля мають бути заповнені"  
 resultTextView.*visibility* = View.*VISIBLE* return@setOnClickListener  
 }  
 //Перевірка чи сума всіх елементів складає 100, за умови, якщо видима комірка азоту (тобто була натиснута кнопка розрахунку складу  
 if (inputNP.*visibility* == View.*VISIBLE*) {  
 if (!isSum100()) {  
 resultTextView.*text* = "Сума елементів повинна дорівнювати 100%."  
 resultTextView.*visibility* = View.*VISIBLE* return@setOnClickListener  
 }  
 showCompositionResult(calculateComposition())  
 } else {  
 showRecountResult(calculateRecount())  
 }  
 **}** //Ініціалізація кнопки повернення назад(зміни даних)  
 buttonChangeInputs.setOnClickListener **{** toggleInputs(show = true)  
 **}** }  
  
 //Функція перевірки заповненості полів (перевірка щодо ВИДИМИХ полів)  
 private fun areInputsValid(): Boolean {  
 val visibleFields = inputFields.*filter* **{ it**.*visibility* == View.*VISIBLE* **}** + *listOf*(inputNP, inputQ, inputVenadii).*filter* **{ it**.*visibility* == View.*VISIBLE* **}** return visibleFields.*all* **{** field **->** field.*text*.toString().*isNotBlank*()  
 **}** }  
  
 //Функція перевірки суми елементів  
 private fun isSum100(): Boolean {  
 val inputValues = getInputValues()  
 val sum = inputValues.*sum*() + inputNP.*text*.toString().*toDouble*()  
 return sum == 100.0  
 }  
  
 //Функція для виведення результату розрахунку складу  
 private fun showCompositionResult(results: Map<String, Any>) {  
 val inputValues = results["inputValues"] as List<Double>  
 val recountedValuesKpc = results["recalculatedKpc"] as DoubleArray  
 val recountedValuesKph = results["recalculatedKph"] as DoubleArray  
 resultTextView.*text* = """  
 Для палива з компонентним складом:   
 HP=${inputValues[0]}%;   
 CP=${inputValues[1]}%;   
 SP=${inputValues[2]}%;   
 NP=${results["np"]}%;   
 OP=${inputValues[3]}%;  
 WP=${inputValues[4]}%;   
 AP=${inputValues[5]} %:  
 - Коефіцієнт переходу від робочої до сухої маси становить: ${results["kpc"]};  
 - Коефіцієнт переходу від робочої до горючої маси становить: ${results["kph"]};  
 - Склад сухої маси палива становитиме:   
 HС=${recountedValuesKpc[0]}%;   
 CС=${recountedValuesKpc[1]}%;   
 SС=${recountedValuesKpc[2]}%;   
 NС=${recountedValuesKpc[3]};   
 OС=${recountedValuesKpc[4]}%,   
 АС=${recountedValuesKpc[5]}%;  
 - Склад горючої маси палива становитиме:   
 HГ=${recountedValuesKph[0]}%;   
 CГ=${recountedValuesKph[1]}%;   
 SГ=${recountedValuesKph[2]}%;   
 NГ=${recountedValuesKph[3]};  
 OГ=${recountedValuesKph[4]}%;  
 - Нижча теплота згоряння для робочої маси за заданим складом компонентів палива становить: ${results["qph"]}, МДж/кг;  
 - Нижча теплота згоряння для сухої маси за заданим складом компонентів палива становить: ${results["qch"]} МДж/кг;  
 - Нижча теплота згоряння для горючої маси за заданим складом компонентів палива становить: ${results["qhh"]} МДж/кг.  
 """.*trimIndent*()  
  
 resultTextView.*visibility* = View.*VISIBLE* toggleInputs(show = false)  
 }  
  
 //Фунцкія для виведення результату перерахунку  
 private fun showRecountResult(results: Map<String, Any>) {  
 val inputValues = results["inputValues"] as List<Double>  
 val recountedValues = results["recalculatedComposition"] as DoubleArray  
 resultTextView.*text* = """  
 Для складу горючої маси мазуту, що задано наступними параметрами:   
 HГ=${inputValues[0]}%;  
 CГ=${inputValues[1]}%;   
 SГ=${inputValues[2]}%;   
 OГ=${inputValues[3]}%;   
 VГ=${inputValues[1]};   
 WГ=${inputValues[4]}%;   
 AГ=${results["venadii"]};   
 та нижчою теплотою згоряння горючої маси мазуту Qidaf = ${results["q"]} МДж/кг:  
 - Склад робочої маси мазуту становитиме:   
 СР=${recountedValues[0]}%;   
 НР=${recountedValues[1]}%;   
 SР=${recountedValues[2]}%;   
 OР=${recountedValues[3]}%,  
 VР=${results["mvp"]} мг/кг;   
 АР=${results["ma"]}%;  
 - Нижча теплота згоряння мазуту на робочу масу для робочої маси за заданим складом  
 компонентів палива становить: ${results["qri"]} МДж/кг.  
 """.*trimIndent*()  
  
 resultTextView.*visibility* = View.*VISIBLE* toggleInputs(show = false)  
 }  
  
 //  
 private fun toggleInputs(show: Boolean) {  
 val visibility = if (show) View.*VISIBLE* else View.*GONE* inputFields.*forEach* **{ it**.*visibility* = visibility **}** buttonComposition.*visibility* = visibility  
 buttonRecount.*visibility* = visibility  
  
 buttonChangeInputs.*visibility* = if (show) View.*GONE* else View.*VISIBLE* resultTextView.*visibility* = if (show) View.*GONE* else View.*VISIBLE* inputNP.*visibility* = View.*GONE* inputQ.*visibility* = View.*GONE* inputVenadii.*visibility* = View.*GONE* buttonCalculate.*visibility* = View.*GONE* }  
  
 //Функція виведення додаткових комірок вводу та прихвування зайвих кнопок для розрахунку складу  
 private fun showCompositionInputs() {  
 inputNP.*visibility* = View.*VISIBLE* buttonCalculate.*visibility* = View.*VISIBLE* buttonComposition.*visibility* = View.*GONE* buttonRecount.*visibility* = View.*GONE* buttonChangeInputs.*visibility* = View.*VISIBLE* }  
 //Функція виведення додаткових комірок вводу та прихвування зайвих кнопок для перерахунку  
 private fun showRecountInputs() {  
 inputQ.*visibility* = View.*VISIBLE* inputVenadii.*visibility* = View.*VISIBLE* buttonCalculate.*visibility* = View.*VISIBLE* buttonComposition.*visibility* = View.*GONE* buttonRecount.*visibility* = View.*GONE* buttonChangeInputs.*visibility* = View.*VISIBLE* }  
 //Отримання данних з комірок  
 private fun getInputValues(): List<Double> {  
 return inputFields.*map* **{ it**.*text*.toString().*toDouble*() **}** }  
 //Додаткова функця дл спрощення перемноження елементів  
 private fun multiplyMultipleValues(values: DoubleArray, k: Double): DoubleArray {  
 return values.*map* **{** BigDecimal((**it**\*k)).setScale(2, RoundingMode.*HALF\_UP*).toDouble() **}**.*toDoubleArray*()  
 }  
  
 //Функція розрахунку складу  
 private fun calculateComposition(): Map<String, Any> {  
 //Отримання початкових даних  
 val inputValues = getInputValues()  
 val np = inputNP.*text*.toString().*toDouble*()  
 val hp = inputValues[0]  
 val cp = inputValues[1]  
 val sp = inputValues[2]  
 val op = inputValues[3]  
 val wp = inputValues[4]  
 val ap = inputValues[5]  
  
 //Розрахунок коефіціентів переходу  
 val kpc = BigDecimal((100 / (100 - wp))).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
 val kph = BigDecimal((100 / (100 - wp - ap))).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
 //Розрахунок складу сухої та горючої маси за допомогою коефціентів  
 val recalculatedKpc = multiplyMultipleValues(*doubleArrayOf*(hp, cp, sp, np, op, ap), kpc)  
 val recalculatedKph = multiplyMultipleValues(*doubleArrayOf*(hp, cp, sp, np, op), kph)  
 //Розрахунок нижчих тепліт згорання  
 val qph = BigDecimal(((339 \* cp + 1030 \* hp - 108.8 \* (op - sp) - 25 \* wp) / 1000)).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
 val qch = BigDecimal(((qph + 0.025 \* wp) \* kpc)).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
 val qhh = BigDecimal(((qph + 0.025 \* wp) \* kph)).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
  
 return *mapOf*(  
 "inputValues" *to* inputValues,  
 "recalculatedKpc" *to* recalculatedKpc,  
 "recalculatedKph" *to* recalculatedKph,  
 "np" *to* np,  
 "kpc" *to* kpc,  
 "kph" *to* kph,  
 "qph" *to* qph,  
 "qch" *to* qch,  
 "qhh" *to* qhh  
 )  
 }  
  
 //Фунцкія для перерахунку складу  
 private fun calculateRecount(): Map<String, Any> {  
 //Отримання початкових даних  
 val inputValues = getInputValues()  
 val hp = inputValues[0]  
 val cp = inputValues[1]  
 val sp = inputValues[2]  
 val op = inputValues[3]  
 val wp = inputValues[4]  
 val ap = inputValues[5]  
 val q = inputQ.*text*.toString().*toDouble*()  
 val venadii = inputVenadii.*text*.toString().*toDouble*()  
  
 //Отримання множників для перерахунку складу палива на робочу масу  
 val m = BigDecimal(((100 - wp - ap) / 100)).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
 val mwa = BigDecimal(((100 - wp) / 100)).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
 //Обрахунок складу робочої маси мазуту  
 val recalculatedComposition = multiplyMultipleValues(*doubleArrayOf*(hp, cp, sp, op), m)  
  
 val ma = ap \* mwa  
 val mvp = venadii \* mwa  
 //Перерахунок теплоти згоряння з горючої маси на робочу  
 val qri = BigDecimal((q \* ((100 - wp - ap) / 100) - 0.025 \* wp)).setScale(2, RoundingMode.*HALF\_UP*).toDouble()  
  
 return *mapOf*(  
 "q" *to* q,  
 "venadii" *to* venadii,  
 "inputValues" *to* inputValues,  
 "qri" *to* qri,  
 "recalculatedComposition" *to* recalculatedComposition,  
 "ma" *to* ma,  
 "mvp" *to* mvp)  
 }  
  
}

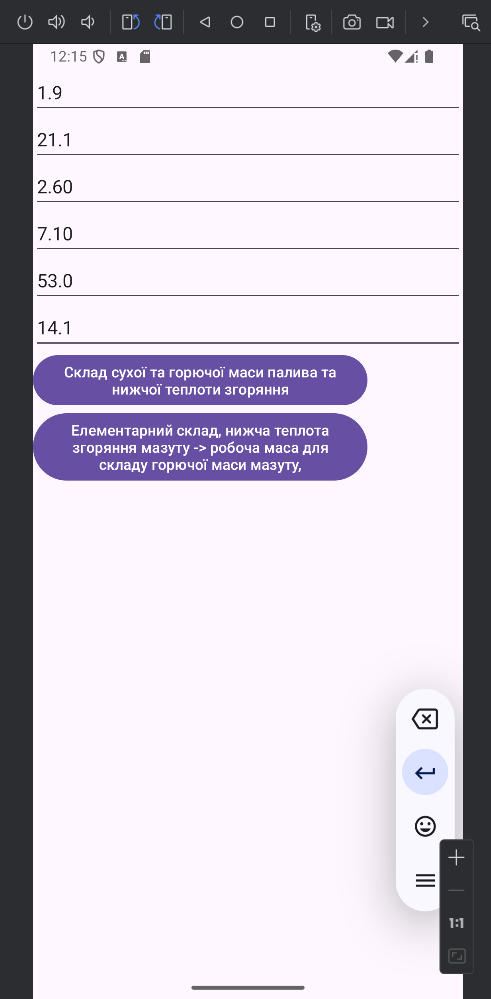
В сій частині коду відбувається отримання даних з рядків вводу інтерфейсу. Опісля залежно від натиснутої кнопки з’являються додаткові поля конкретно орієнтовані на обраний розрахунок, після заповнення всіх полів(відбувається перевірка чи всі поля заповнені) відбуваються обрахунки. Окрім того для обрахунку складу проводиться перевірка суми введених даних.   
Обрахунки округлюються до сотих.  
  
Після обрання, що саме користувач бажає обрахувати з’являється додаткова кнопка, яка дозволяє повернутись назад.

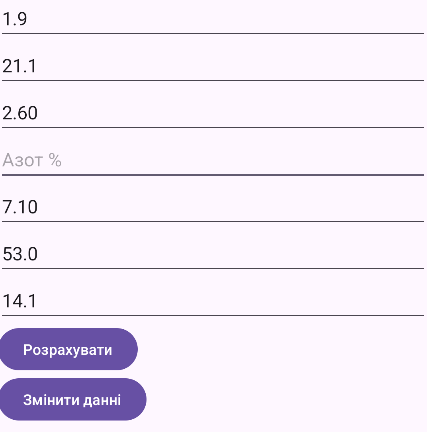
Код, що формує вигляд інтерфейсу має вигляд:  
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp">  
  
 <EditText  
 android:id="@+id/input\_hp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/voden" />  
  
 <EditText  
 android:id="@+id/input\_cp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/vuhletz" />  
  
 <EditText  
 android:id="@+id/input\_sp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/sirka" />  
  
 <EditText  
 android:id="@+id/input\_np"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/azot"  
 android:visibility="gone" />  
  
 <EditText  
 android:id="@+id/input\_op"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/o2" />  
  
 <EditText  
 android:id="@+id/input\_wp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/voloha"/>  
  
 <EditText  
 android:id="@+id/input\_ap"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/zola" />  
  
  
 <EditText  
 android:id="@+id/input\_q"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/teplota\_zhoraniia"  
 android:visibility="gone" />  
  
 <EditText  
 android:id="@+id/input\_venadii"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/vernadii"  
 android:visibility="gone" />  
  
 <Button  
 android:id="@+id/button\_composition"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/composition" />  
  
 <Button  
 android:id="@+id/button\_recount"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/recount" />  
  
 <Button  
 android:id="@+id/button\_calculate"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/enter"  
 android:visibility="gone" />  
  
 <Button  
 android:id="@+id/button\_change\_inputs"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/change\_inputs"  
 android:visibility="gone" />  
  
 <TextView  
 android:id="@+id/result"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/result"  
 android:visibility="gone" />  
</LinearLayout>

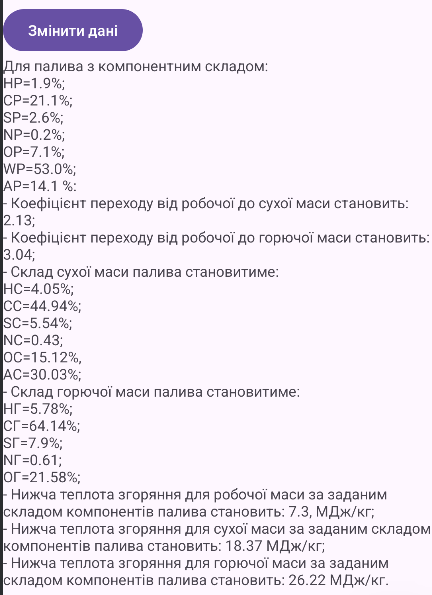
В сьому коді створюються кнопки та комірки для ведення даних.  
  
**Результат роботи програми за контрольним прикладом:**

Для першого завдання:

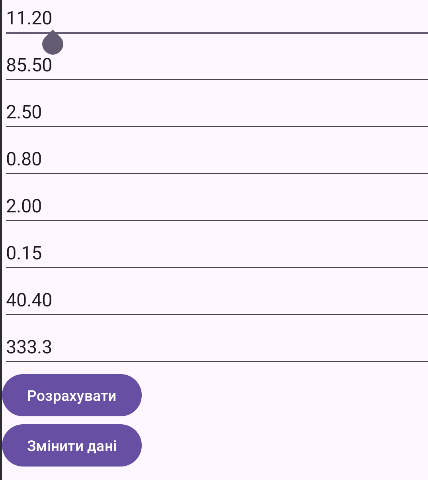


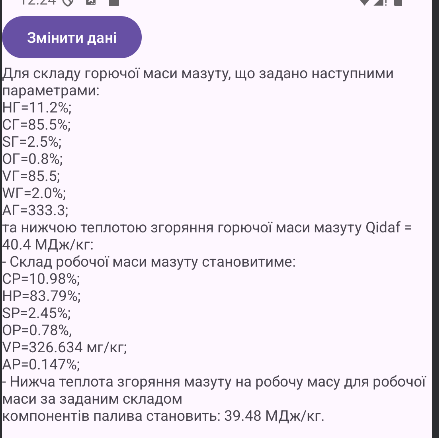




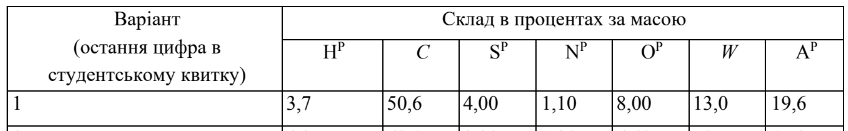


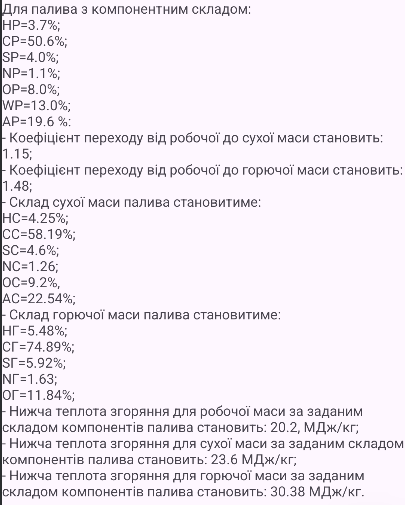
Для другого завдання:





**Результат роботи програми за варіантом (11(1)):**





**Висновок**

В результаті виконання лабораторної роботи №1 було опрацьовано алгоритм розрахунку складу сухої та горючої маси палива та нижчої теплоти згоряння для робочої, сухої та горючої маси за заданим складом компонентів палива. Також було проведено ознайомлення з Kotlin та реалізовано дані розрахунку у вигляді калькулятора Kotlin.