

**Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки**

Лабораторна робота 1

**на тему: “Дослідження роботи з елементами керування”
з дисципліни “Розробка мобільних застосунків під Android”**

Виконала:
студентка групи ІМ-32
Сірик Яна Олександрівна

Перевірив:
Орленко С.П.

Київ 2026

Мета роботи: дослідити створення простого застосунку під платформу Андроїд та набути практичні навички з використання елементів керування інтерфейсу, мов програмування Java чи Kotlin.

ЗАВДАННЯ

Написати програму під платформу Андроїд, яка має інтерфейс для введення або/та вибору даних згідно варіанту (таблиця) і відображає результат взаємодії з цим інтерфейсом у деяке текстове поле цього інтерфейсу. Передбачити наступне: якщо не всі дані введені або обрані, а користувач натискає кнопку для отримання результату, то відобразити вікно, що спливає, з повідомленням завершити введення всіх даних.

ВИЗНАЧЕННЯ ВАРІАНТУ

Згідно з інструкцією: 18-й студент - 1-й варіант.

Мій номер - **110**.

110%17=8-й варіант.

Варіант	<i>Вікно інтерфейсу</i>
8.	Вікно містить два текстові поля, групу опцій (операції), тобто радіо-батони, та кнопку «ОК». Вивести результат обчислення обраної операції над введеними даними при натисканні на кнопку «ОК» у деяке текстове поле.

ХІД РОБОТИ

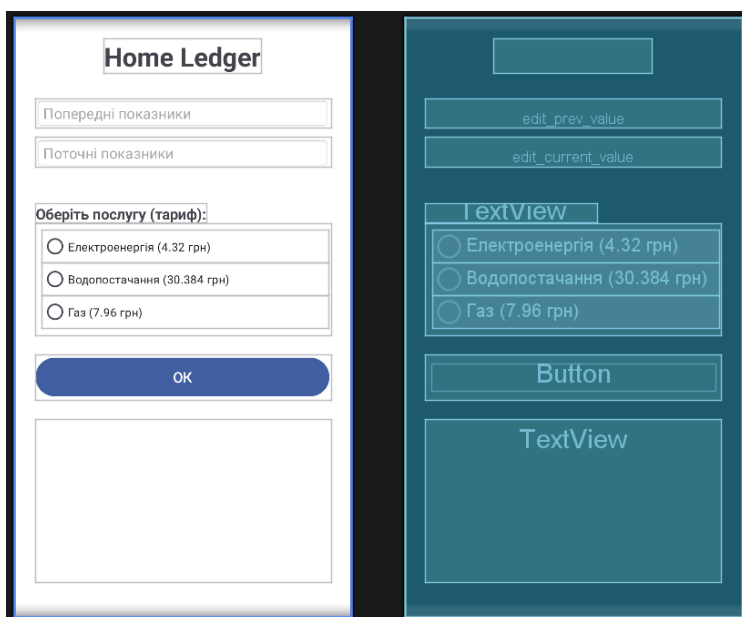
1. Опис структури інтерфейсу (Layout).

Для створення графічного інтерфейсу було використано XML-розмітку з кореневим контейнером LinearLayout.

Основні компоненти інтерфейсу:

- Заголовок (Header).
- Поля введення (EditText):
 - `edit_prev_value` та `edit_current_value`: Призначені для введення показників лічильника. Важливо: Встановлено атрибут `android:inputType="numberDecimal"`. Це обмежує введення лише цифрами та крапкою, запобігаючи помилкам типу `NumberFormatException` при розрахунках.
- Вибір послуги (RadioGroup):
 - Група містить три `RadioButton` (Електроенергія, Вода, Газ). Використання `RadioGroup` забезпечує взаємовиключний вибір: користувач може обрати лише одну послугу одночасно.
- Кнопка дії (Button):
 - `btn_calculate_bill`: Запускає процес обробки даних у коді Kotlin.
- Поле виводу (TextView):
 - `tv_bill_result`: Динамічне поле, де відображаються розрахунки.

Вигляд програми:



2. Програмна логіка.

Логіка застосунку зосереджена у файлі `MainActivity.kt` і розділена на три етапи: ініціалізація, валідація та розрахунок.

Етап 1: Ініціалізація та зв'язок

Використано метод `findViewById`, який пов'язує XML-елементи з об'єктами в коді. Це дозволяє програмі «зчитувати» текст із полів та «писати» результат у `TextView`.

Етап 2: Валідація даних (Захист від помилок)

Згідно з технічним завданням, реалізовано перевірку:

1. Повнота даних: Якщо хоча б одне поле пусте або не обрано тариф, функція `Toast.makeText(...).show()` виводить спливаюче вікно з попередженням.
2. Логічність показників: Додано перевірку `if (currValue < prevValue)`, щоб не дозволити розрахунок, якщо поточні показники лічильника менші за попередні.

Етап 3: Обчислення за допомогою конструкції `when`

Для визначення ціни використано оператор `when`:

- `rb_electric` присвоює `rate = 4.32`
- `rb_water` присвоює `rate = 30.384`
- `rb_gas` присвоює `rate = 7.96`

3. Форматування та виведення результату.

Для того, щоб результат виглядав гарно, було застосовано:

1. `String.format("%.2f", value)`: Це обмежує кількість знаків після коми до двох. Це критично для фінансових розрахунків, де результат не може мати вигляд `150.33333333`.
2. `trimIndent()`: Дозволяє оформити багаторядковий текст прямо в коді з відступами, що покращує читабельність.

КОД ПРОГРАМИ

Посилання на GitHub: <https://github.com/hinoneko/Android-Lab1>

РОБОТА ПРОГРАМИ

Основний вигляд

Home Ledger

Попередні показники

Поточні показники

Оберіть послугу (тариф):

- ☐ Електроенергія (4.32 грн)
- ☐ Водопостачання (30.384 грн)
- ☐ Газ (7.96 грн)

ОК

Введення значень

Home Ledger

766.12

1210.96

Оберіть послугу (тариф):

- ☒ Електроенергія (4.32 грн)
- ☐ Водопостачання (30.384 грн)
- ☐ Газ (7.96 грн)



OK

Виведення результату

Home Ledger

766.12

1210.96

Оберіть послугу (тариф):

☒ Електроенергія (4.32 грн)

☐ Водопостачання (30.384 грн)

☐ Газ (7.96 грн)

≡

OK

Послуга:

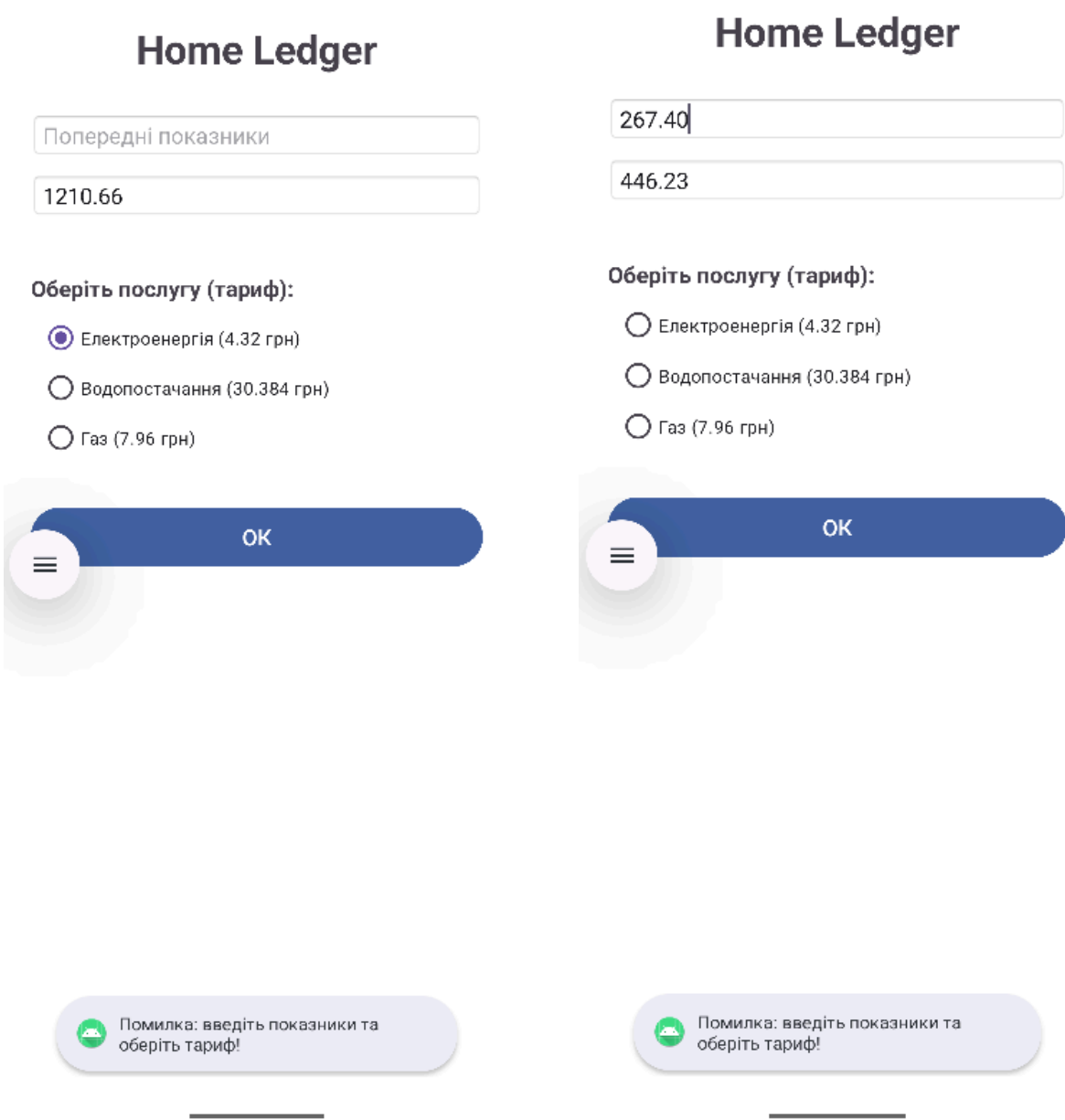
Електроенергія (4.32 грн)

Використано: 444.84 од.

Разом до сплати: 1921.71 грн



Спливаюче вікно про помилку з недостатньою кількістю даних



Спливаюче вікно про помилку з некоректно введеними даними

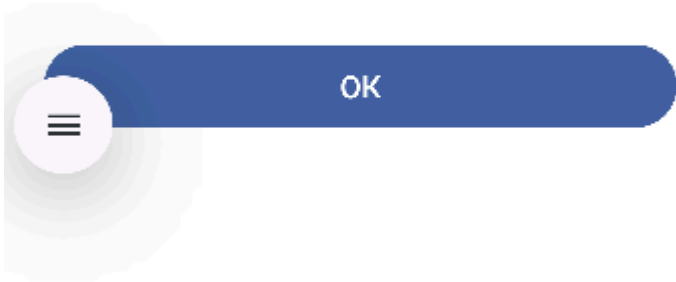
Home Ledger


267.40

133.29

Оберіть послугу (тариф):

- ☐ Електроенергія (4.32 грн)
- ☐ Водопостачання (30.384 грн)
- ☒ Газ (7.96 грн)





Помилка: поточні показники менші за попередні!



ВИСНОВОК

Під час виконання лабораторної роботи №1 я опанувала основи розробки мобільних застосунків для платформи Android з використанням мови Kotlin та середовища Android Studio.

За результатами роботи було досягнуто наступних цілей:

1. Проектування інтерфейсу (UI): Я навчився створювати декларативну розмітку за допомогою XML.
2. Робота з елементами керування: На практиці застосовано базові віджети:
 - EditText із налаштованим типом вводу (numberDecimal) для отримання числових даних;
 - RadioGroup та RadioButton для реалізації логіки вибору тарифу (взаємовиключний вибір);
 - Button як основний тригер для виконання обчислень.
3. Логіка та валідація: Реалізовано механізм обробки подій за допомогою `setOnClickListener`. Особливу увагу приділено валідації введених даних: розроблено систему перевірки на порожні поля та логічну коректність показників (поточні показники не можуть бути меншими за попередні). Для інформування користувача про помилки використано спливаючі повідомлення `Toast`.
4. Обробка даних у Kotlin: Закріплено навички роботи з умовними конструкціями (`when`), математичними розрахунками та форматуванням рядків. Використання методу `String.format("%.2f", ...)` дозволило забезпечити фінансову точність виводу (два знаки після коми).

Загальний результат: Створено функціональний застосунок для розрахунку комунальних платежів, який повністю відповідає вимогам варіанту №8 та стандартам розробки під ОС Android.

КОНТРОЛЬНІ ПИТАННЯ

1. Архітектура застосунку під платформу Андроїд.

Андроїд базується на багатошаровій архітектурі:

- Linux Kernel: Керує драйверами, пам'яттю та процесами.
- Hardware Abstraction Layer (HAL): Надає стандартні інтерфейси для доступу до заліза (камера, Bluetooth).
- Android Runtime (ART): Виконує байт-код застосунків.
- Native C/C++ Libraries: Бібліотеки для роботи з графікою, медіа та базами даних.
- Java API Framework: Набір інструментів, які ми використовуємо в Android Studio (View System, Resource Manager).
- System Apps: Пошта, контакти, календар та твій проект SmartBill.

2. Загальний огляд компонентів застосунку.

Існує чотири основні типи компонентів:

1. Activities (Діяльності): Візуальний інтерфейс користувача (як твоя MainActivity).
2. Services (Служби): Виконують роботу у фоновому режимі (наприклад, програвання музики).
3. Broadcast Receivers: Приймають сповіщення від системи або інших програм (наприклад, сигнал про низький заряд батареї).
4. Content Providers: Керують доступом до спільних даних (наприклад, списку контактів).

3. Життєвий цикл компоненту “Діяльність”.

Це послідовність станів, через які проходить вікно програми:

- onCreate(): Викликається при створенні (ініціалізація setContentView).
- onStart(): Діяльність стає видимою користувачеві.
- onResume(): Користувач може взаємодіяти з програмою.
- onPause(): Програма частково перекрита (наприклад, діалоговим вікном).
- onStop(): Програма повністю прихована.
- onDestroy(): Знищення діяльності для вивільнення ресурсів.

4. Життєвий цикл компоненту “Служба”.

Залежить від типу запуску:

- Started Service: onCreate() -> onStartCommand() -> Працює -> onDestroy().
- Bound Service (Прив'язана): onCreate() -> onBind() -> Клієнт взаємодіє -> onUnbind() -> onDestroy().

5. Опис процесів платформи Андроїд.

Система намагається тримати процеси якомога довше, але при нестачі пам'яті вбиває їх за пріоритетом:

1. Foreground process (Передній план): Те, що користувач бачить зараз.
2. Visible process: Видимий, але не активний (наприклад, прихований діалогом).
3. Service process: Працює фонові служба.
4. Cached process (Кешований): Непотрібний зараз процес, зберігається для швидкого перезапуску.

6. Яким чином активуються компоненти застосунку.

Компоненти (крім Content Providers) активуються за допомогою Intents (Намірів). Це асинхронні повідомлення, які кажуть системі: "Я хочу запустити Activity X" або "Я хочу надіслати повідомлення системі".

7. Призначення файлу маніфесту та його структура.

AndroidManifest.xml - це "паспорт" програми. Він каже системі:

- Яка назва пакета програми.
- Які компоненти в ній є (Activities, Services).
- Які дозволи потрібні (доступ до інтернету, камери).
- Яка іконка та назва застосунку. Структура: Тег <manifest>, всередині якого <application>, а в ньому - теги <activity>, <service> тощо.

8. Поняття ресурсу та яким чином визначаються ресурси.

Ресурси - це все нетекстове наповнення програми (зображення, рядки, кольори, макети XML).

- Вони зберігаються в папці res/.
- Кожен ресурс отримує унікальний ID у згенерованому класі R.java (наприклад, R.id.btn_ok).
- Це дозволяє змінювати дизайн (наприклад, перекладати текст на іншу мову), не змінюючи програмний код.