Міністерство освіти і науки України
КПІ ім. Ігоря Сікорського
Факультет інформатики та обчислюваної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №5

з дисципліни: «Розробка мобільних застосунвань під Android»

Перевірив:
Викладач, PhD
Орленко С.П.

Виконав:
студент гр. ІК-23
Волнушкін В.І.

2025 р.

Мета роботи: ознайомитись з можливостями вбудованих датчиків мобільних пристроїв та дослідити способи їх використання для збору та обробки даних.

Завдання:

– Виконано

БАЗОВЕ (10/20 балів). Написати програму під платформу Андроїд, яка має інтерфейс для виведення даних з обраного вбудованого датчика (тип обирається самостійно, можна відслідковувати зміни значень і з декількох датчиків).

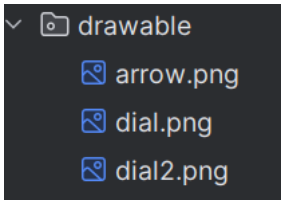
ПОВНЕ (20/20). Функціональність базового додатку додатково розширюється обробкою отриманих даних та виведенням їх у відповідній формі.

Примітка: конкретного варіанту не передбачено, студент сам обирає завдання та вигляд програми. Приклади очікуваних робіт:

- компас з ілюстрацією стрілки (циферблату з позначеними сторонами світу);

Хід виконання

Додаємо до проєкту два зображення (стороні світу – dial.png або dial2.png та стрілка – arrow.png)



Обмежуємо rotation портретним режимом у AndroidManifest.XML:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplicationLab5"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
```

```
            android:exported="true"
            android:screenOrientation="portrait"
            android:configChanges="orientation"
            android:label=""
            android:theme="@style/Theme.MyApplicationLab5">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

MainActivity.kt з усією технічною реалізацією застосунку «Компас»:

```
package com.example.myapplicationlab5

import android.hardware.*
import android.os.Bundle
import android.view.animation.RotateAnimation
import android.widget.ImageView
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity(), SensorEventListener {

    private lateinit var sensorManager: SensorManager
    private var accelerometerReading = FloatArray(3)
    private var magnetometerReading = FloatArray(3)

    private lateinit var arrowImage: ImageView
    private lateinit var directionText: TextView
    private var currentAzimuth: Float = 0f

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        arrowImage = findViewById(R.id.compass_arrow)
        directionText = findViewById(R.id.direction_text)
        sensorManager = getSystemService(SENSOR_SERVICE) as SensorManager
    }

    override fun onResume() {
        super.onResume()
        sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)?.also {
            sensorManager.registerListener(this, it,
SensorManager.SENSOR_DELAY_UI)
        }
        sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD)?.also {
            sensorManager.registerListener(this, it,
SensorManager.SENSOR_DELAY_UI)
        }
    }

    override fun onPause() {
        super.onPause()
```

```
        sensorManager.unregisterListener(this)
    }

    override fun onSensorChanged(event: SensorEvent) {
        when (event.sensor.type) {
            Sensor.TYPE_ACCELEROMETER -> accelerometerReading =
event.values.clone()
            Sensor.TYPE_MAGNETIC_FIELD -> magnetometerReading =
event.values.clone()
        }

        if (accelerometerReading.isNotEmpty() &&
magnetometerReading.isNotEmpty()) {
            val rotationMatrix = FloatArray(9)
            val success = SensorManager.getRotationMatrix(
                rotationMatrix,
                null,
                accelerometerReading,
                magnetometerReading
            )
            if (success) {
                val orientationAngles = FloatArray(3)
                SensorManager.getOrientation(rotationMatrix,
orientationAngles)
                val azimuthInRadians = orientationAngles[0]
                val azimuthInDegrees =
Math.toDegrees(azimuthInRadians.toDouble()).toFloat()
                val newAzimuth = (azimuthInDegrees + 360) % 360

                // Плавне обертання стрілки та тексту навколо центру
                smoothRotate(arrowImage, currentAzimuth, -newAzimuth)
                smoothRotate(directionText, currentAzimuth, -newAzimuth)

                // Оновлення напрямку з градусами
                val direction = getDirectionLabel(newAzimuth)
                val degreeText = "$direction (${newAzimuth.toInt()}°)"
                directionText.text = degreeText

                currentAzimuth = -newAzimuth
            }
        }
    }

    private fun smoothRotate(view: android.view.View, from: Float, to:
Float) {
        val rotateAnimation = RotateAnimation(
            from,
            to,
            RotateAnimation.RELATIVE_TO_SELF, 0.5f,
            RotateAnimation.RELATIVE_TO_SELF, 0.5f
        )
        rotateAnimation.duration = 200  // Плавний час для анімації
        rotateAnimation.fillAfter = true
        view.startAnimation(rotateAnimation)
    }

    private fun getDirectionLabel(degrees: Float): String {
        return when (degrees) {
            in 337.5..360.0, in 0.0..22.5 -> "Північ"
            in 22.5..67.5 -> "Північний схід"
            in 67.5..112.5 -> "Схід"
```

```
            in 112.5..157.5 -> "Південний схід"
            in 157.5..202.5 -> "Південь"
            in 202.5..247.5 -> "Південний захід"
            in 247.5..292.5 -> "Захід"
            in 292.5..337.5 -> "Північний захід"
            else -> ""
        }
    }

    override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {}

    override fun onSaveInstanceState(outState: Bundle) {
        super.onSaveInstanceState(outState)
        outState.putFloat("azimuth", currentAzimuth)
    }

    override fun onRestoreInstanceState(savedInstanceState: Bundle) {
        super.onRestoreInstanceState(savedInstanceState)
        currentAzimuth = savedInstanceState.getFloat("azimuth")
        arrowImage.rotation = currentAzimuth
        directionText.rotation = currentAzimuth
    }
}
```

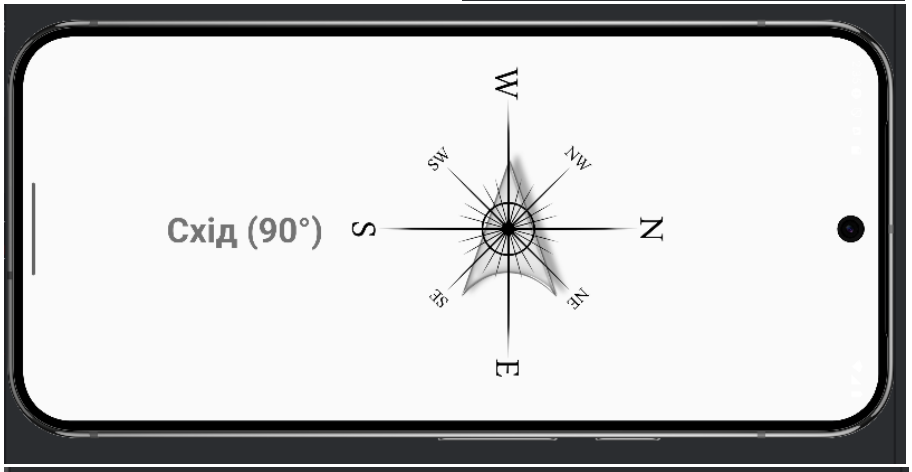
activity_main.xml

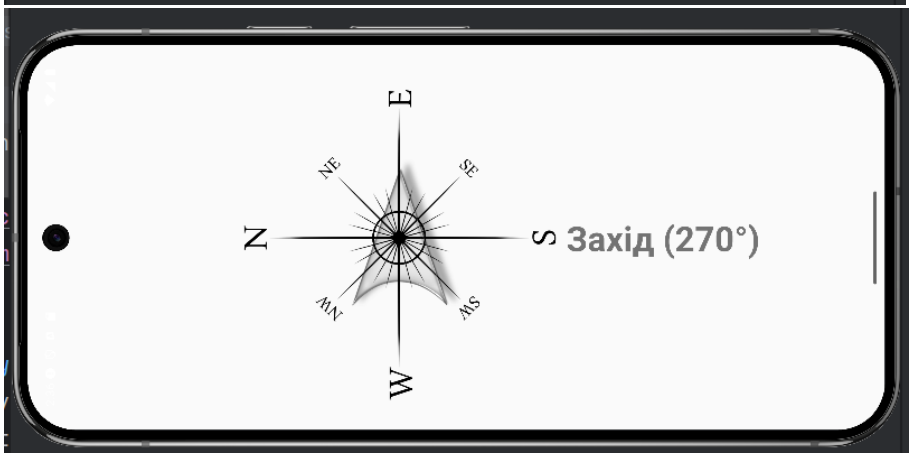
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">

    <ImageView
        android:id="@+id/compass_background"
        android:layout_width="400dp"
        android:layout_height="400dp"
        android:src="@drawable/dial"
        android:layout_centerInParent="true" />

    <ImageView
        android:id="@+id/compass_arrow"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:src="@drawable/arrow"
        android:layout_centerInParent="true" />

    <TextView
        android:id="@+id/direction_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="40sp"
        android:textStyle="bold"
        android:layout_centerHorizontal="true"
        android:layout_below="@id/compass_background"
        android:layout_marginTop="80dp"/>
</RelativeLayout>
```

Таким чином отримуємо наступний результат:


N
NW
NE
W
E
SW
SE
S
Північ (0°)


Південь (180°)
S
SE
SW
E
W
NE
NW
N


Схід (90°)
W
SW
NW
S
N
SE
NE
E


E
NE
SE
N
S
Захід (270°)
NW
SW
W

Як можна бачити, хоч зміна орієнтації і не здійснюється при повороті пристрою, оскільки ми її обмежели портретним режимом, проте стрілка та текст, що описує напрямок – повертаються навколо своєї осі, «синхронізуючись» зі стрілкою.

Контрольні питання

1. Наведіть приклади вбудованих датчиків та величини які з них можна зчитати.

Датчик	Опис	Величини
Акселерометр	Вимірює прискорення пристрою по трьох осях	Прискорення по X, Y, Z (в м/с²)
Гіроскоп	Визначає кутове обертання пристрою	Кутова швидкість по X, Y, Z (рад/с)
Магнітометр (компас)	Вимірює магнітне поле Землі	Напрямок магнітного поля по X, Y, Z
Датчик освітленості	Вимірює рівень освітлення	Освітленість (люкси)
Датчик наближення	Визначає наявність об’єкта біля екрана	Відстань до об'єкта (зазвичай у см)
Барометр	Вимірює атмосферний тиск	Тиск у гПа
Датчик температури	Вимірює температуру навколишнього середовища	Температура (°C)
Датчик вологості	Вимірює відносну вологість повітря	Вологість (%)
Датчик присутності/активності	Визначає, чи є користувач біля пристрою	Логічне значення (1 або 0)

2. Наведіть особливості роботи з вбудованими датчиками.

- **SensorManager** — головний клас для роботи з усіма сенсорами.

  Метод `getSensorList(Sensor.TYPE_...)` — отримання списку доступних датчиків.

  Метод `registerListener()` — підписка на події сенсора.

- **SensorEventListener** — інтерфейс, який дозволяє отримувати дані з сенсорів через метод:

  `onSensorChanged(SensorEvent event)` — нові значення сенсора.

  `onAccuracyChanged()` — зміни точності.

- **Частота оновлення**:

Можна вказати бажану частоту (наприклад, `SENSOR_DELAY_NORMAL`, `SENSOR_DELAY_UI`, `SENSOR_DELAY_GAME`, `SENSOR_DELAY_FASTEST`).

- **Економія батареї**:

  Часті оновлення та використання декількох сенсорів одночасно — **збільшують споживання енергії**.

  Необхідно **відписуватися від сенсорів** в `onPause()` або `onStop()` методом `unregisterListener()`.

- **Деякі датчики — віртуальні**:

  Наприклад, `TYPE_ROTATION_VECTOR`, `TYPE_ORIENTATION` — це **об'єднання даних з кількох фізичних сенсорів** (акселерометр + гіроскоп + магнітометр).

- **Не всі пристрої мають усі сенсори**:

  Потрібно перевіряти їх наявність перед використанням:

```
val hasGyro =
sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE) != null
```