



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO  
FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

DIVISIÓN SISTEMA UNIVERSIDAD ABIERTA Y EDUCACIÓN A DISTANCIA

**SUAYED**  
Sistema Universidad Abierta y  
Educación a Distancia  
UNAM

## **PROGRAMACION DE DISPOSITIV.MOVILES**

### **UNIDAD 5**

#### **ACTIVIDAD 1**

**NOMBRE DEL ALUMNO:**

PAULINA RODRIGUEZ SAMPEDRO

**NOMBRE DEL ASESOR:**

CRISTIAN CARDOSO ARELLANO

**SEXTO SEMESTRE**

GRUPO: 9696

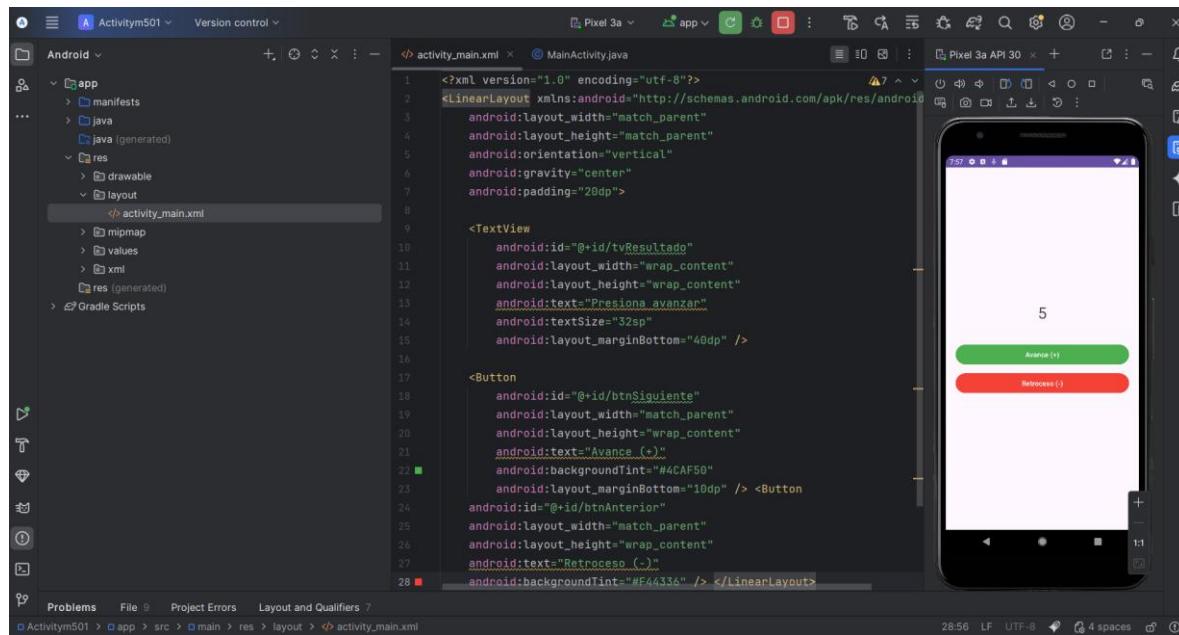
## ESTRUCTURAS CON LA SENTENCIA FOR

### Objetivo

Desarrollar una aplicación móvil en el entorno de Android Studio que permita comprender y practicar la implementación de la estructura de control for, utilizándola como el motor principal para el cálculo secuencial de la serie de Fibonacci.

### Desarrollo

Ilustración 1. Diseño XML



Se utilizó un linearLayout para organizar los elementos y para personalizar los botones con backgroundTint (verde para avanzar, rojo para retroceder) para que sea una interfaz intuitiva.

Ilustración 2. Estructura de código java

The screenshot shows the Android Studio interface. On the left, the project structure is displayed under the 'Android' tab, showing the 'app' module with its sub-directories: manifests, java, res (containing drawable and layout), and xml. The 'layout' directory contains the file 'activity\_main.xml', which is selected. On the right, the code editor displays 'MainActivity.java'. The code is as follows:

```
18 public class MainActivity extends AppCompatActivity { ▲ 4 ✓ 49 ↻
19     private TextView tvResultado;
20     2 usages
21     private Button btnSiguiente, btnAnterior;
22
23     // Variables de control
24     4 usages
25     private int posicionActual = 0;
26
27     // Se utiliza un HashMap para guardar resultados ya calculados
28     // IMPORTANTE: La llave es la posición (Integer) y el valor es el
29     // resultado (Long)
30     3 usages
31     private HashMap<Integer, Long> memo = new HashMap<>();
32
33     @Override
34     protected void onCreate(Bundle savedInstanceState) {
35         super.onCreate(savedInstanceState);
36         setContentView(R.layout.activity_main);
37
38         // Llamamos a los módulos de configuración
39         vincularComponentes();
40         configurarEventos();
41     }
42
43     // Vinculación de componentes XML
44     1 usage
45     private void vincularComponentes() {
46         tvResultado = findViewById(R.id.tvResultado);
47         btnSiguiente = findViewById(R.id.btnSiguiente);
48     }
49 }
```

Para la arquitectura del código, se implementó una estructura basada en la separación de responsabilidades, lo cual considero, permite que el código se vea más limpio. Y como se observa, el método principal `onCreate` actúa únicamente como un coordinador, delegando las tareas específicas a módulos independientes: el método `vincularComponentes` se encarga exclusivamente de enlazar la lógica de Java con los elementos visuales del XML, mientras que `configurarEventos` centraliza la gestión de los escuchadores de clics para los botones.

## PRUEBAS

Ilustración 1. Aplicación abierta

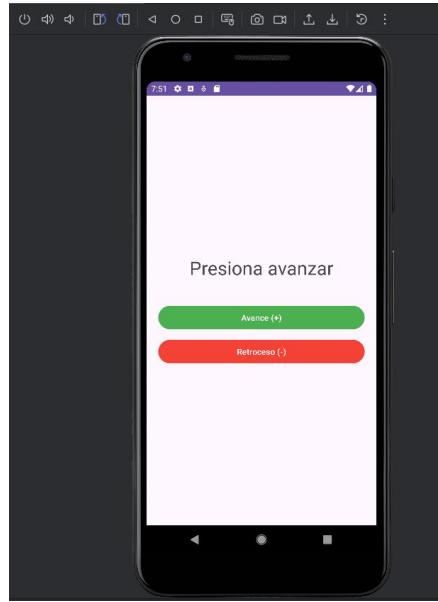


Ilustración 2. Pantalla después de 3 clics

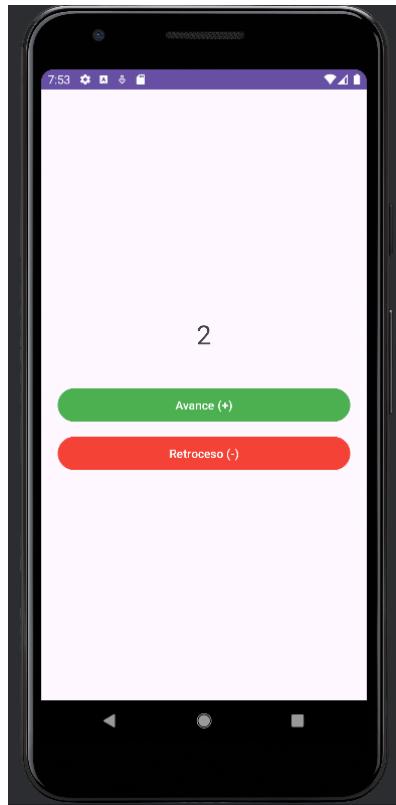
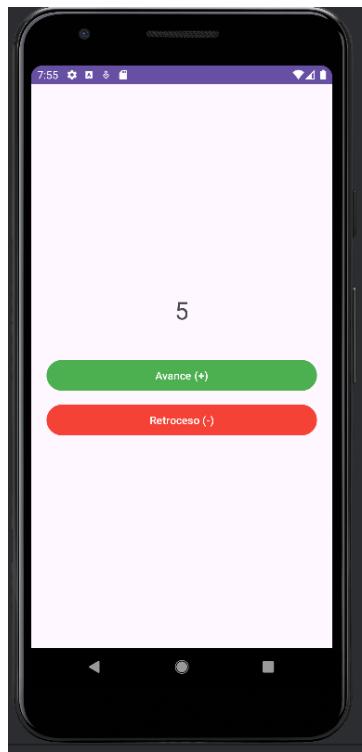


Ilustración 3. Pantalla después de 6 clics de avance



Ilustración 4. Pantalla después de un clic de retroceso



## Conclusión

La realización de esta actividad me permitió comprender la estrecha relación entre la teoría matemática y la optimización de algoritmos en el desarrollo de software. La secuencia de Fibonacci, más allá de ser una sucesión numérica donde cada valor es la suma de los dos anteriores, sirve como el escenario perfecto para aplicar estructuras de control fundamentales como el ciclo for, el cual permite procesar iteraciones de manera controlada y predecible. Sin embargo, el valor más significativo para un profesional de informática radica en la implementación de la memorización, siendo una técnica de optimización que transforma un algoritmo común en uno de alto rendimiento al utilizar un HashMap para almacenar resultados previamente calculados, evitando la redundancia de procesos y reduciendo drásticamente el costo computacional.

## Referencias

- Google Developers. (2024). *Build your first Android app*. Developer.android.com
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3º ed.). MIT Press.
- Oracle. (2024). *HashMap (Java Platform SE 8)*. Docs.oracle.com.

Sedgewick, R., & Wayne, K. (2011). *Algorithms* (4° ed.). Addison-Wesley Professional.