



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

PROGRAMACION DE DISPOSITIV.MOVILES

UNIDAD 1

ACTIVIDAD 3

NOMBRE DEL ALUMNO:

PAULINA RODRIGUEZ SAMPEDRO

NOMBRE DEL ASESOR:

CRISTIAN CARDOSO ARELLANO

SEXTO SEMESTRE

GRUPO: 9696

ARREGLOS Y TIEMPOS DE EJECUCIÓN

Objetivos de aprendizaje

- Aprender a estructurar proyectos, manejar los componentes de una Activity (pantalla) y usar recursos del sistema como Logcat para la depuración y solución de problemas.
- Ejecutar una sentencia básica de programación en lenguaje Java

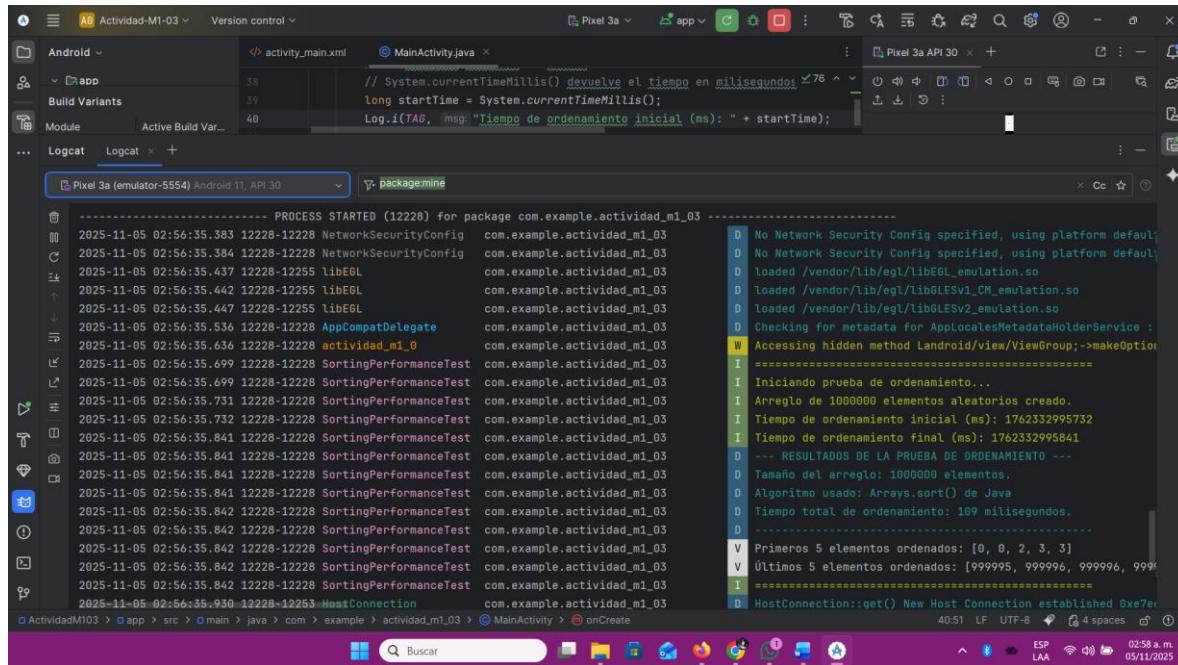
Conocimientos previos requeridos

- Sistemas operativos
- Lenguaje de programación Java
- Instalación de Android Studio

Instrucciones de la actividad

- Crear tu primer activity
- En lenguaje de programación JAVA dentro de Android Studio, crea una app vacía.
- Crear un arreglo con 1 millón de elementos enteros de manera aleatoria.
- Ordenar los elementos.
- Medir el tiempo de inicio a fin.
- Imprimir en el logcat los resultados obtenidos.

Ilustración 1. Resultados obtenidos en el logcat



The screenshot shows the Android Studio interface with the Logcat tab selected. The log output displays the following content:

```
----- PROCESS STARTED (12228) for package com.example.actividad_m1_03 -----
2025-11-05 02:56:35.383 12228-12228 NetworkSecurityConfig com.example.actividad_m1_03
2025-11-05 02:56:35.384 12228-12228 NetworkSecurityConfig com.example.actividad_m1_03
2025-11-05 02:56:35.437 12228-12255 libEGL com.example.actividad_m1_03
2025-11-05 02:56:35.442 12228-12255 libEGL com.example.actividad_m1_03
2025-11-05 02:56:35.447 12228-12255 libEGL com.example.actividad_m1_03
2025-11-05 02:56:35.536 12228-12228 AppCompatDelegate com.example.actividad_m1_03
2025-11-05 02:56:35.636 12228-12228 actividad_m1_0 com.example.actividad_m1_03
2025-11-05 02:56:35.699 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.731 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.732 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.841 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.841 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.841 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.842 12228-12228 SortingPerformanceTest com.example.actividad_m1_03
2025-11-05 02:56:35.936 12228-12253 HostConnection com.example.actividad_m1_03
D No Network Security Config specified, using platform default;
D No Network Security Config specified, using platform default;
D Loaded /vendor/lib/egl/libEGL_emulation.so
D Loaded /vendor/lib/egl/libGLESv1_CM_emulation.so
D Loaded /vendor/lib/egl/libGLESv2_emulation.so
D Checking for metadata for AppLocalesMetadataHolderService :
W Accessing hidden method Landroid/view/ViewGroup;->makeOptionalBackgroundVisible()
I =====
I Iniciando prueba de ordenamiento...
I Arreglo de 1000000 elementos aleatorios creado.
I Tiempo de ordenamiento inicial (ms): 1762332995732
I Tiempo de ordenamiento final (ms): 1762332995841
D --- RESULTADOS DE LA PRUEBA DE ORDENAMIENTO ---
D Tamaño del arreglo: 1000000 elementos.
D Algoritmo usado: Arrays.sort() de Java
D Tiempo total de ordenamiento: 109 milisegundos.
D =====
V Primeros 5 elementos ordenados: [0, 0, 2, 3, 3]
V Últimos 5 elementos ordenados: [999995, 999996, 999996, 999996, 999996]
D HostConnection::get() New Host Connection established 0xe7e1
40:51 LF UTF-8 ⌂ 4 spaces ⌂ ①
05/11/2025
```

Conclusión

El resultado final en el Logcat demuestra que la máquina virtual de Java es extremadamente eficiente, y al utilizar el algoritmo optimizado Dual-Pivot Quicksort a través de `Arrays.sort()`, se pudo crear, llenar y ordenar un arreglo de un millón de números enteros en un tiempo muy corto, específicamente en 109 milisegundos.

Además, la utilización de las librerías de Android son el esqueleto de la aplicación, por ejemplo `androidx.appcompat.app.AppCompatActivity` dota a nuestro código (`MainActivity`) de todas las funcionalidades de una pantalla móvil (la Activity), incluyendo su ciclo de vida y compatibilidad con diferentes versiones de Android. Con `android.os.Bundle` es crucial dentro del método `onCreate()`, ya que se encarga de gestionar y preservar el estado de la aplicación. Para cumplir con el requisito de mostrar los resultados de forma interna, se importó `android.util.Log` esta herramienta de depuración es vital para el desarrollo Android, ya que permite imprimir los tiempos de inicio y fin, el tamaño del arreglo y la duración total del ordenamiento en la pestaña Logcat, cumpliendo así con el sexto paso de la actividad sin necesidad de modificar la interfaz de usuario.

Finalmente, las librerías Java estándar se encargaron de la lógica matemática y de rendimiento. `java.util.Random` fue esencial para la creación del millón de elementos enteros, permitiendo llenar el arreglo con valores aleatorios. El paquete `java.util.Arrays` es el corazón del proceso de rendimiento, pues proporciona el método `Arrays.sort()`, que se utilizó para ordenar eficientemente el millón de elementos, y también incluye utilidades para convertir el arreglo en una cadena legible para su impresión en Logcat. Estas dos últimas librerías son las que ejecutan el cálculo, mientras que las de Android se aseguran de que ese cálculo se pueda mostrar y ejecutar dentro de un entorno móvil.

Referencias

Android Studio (2025).<https://developer.android.com/studio?hl=es-419>