



Programación avanzada

Tiempo empleado:

Nombres

Usuario

Cristhian Javier Villamarin Gaona

Cvilla18

Jandri Giovanni Villavicencio Villavicencio

GiovanniJV

Isaac Josué Álvarez Vásquez

IsaacAlvarez12

Kevin Ramiro Cabrera Cabrera

krcabrera

Xavier Alexander Chávez Saraguro

xavierchavez916

José Fabian Montoya Montoya

f4biaan

Nixon Javier Vuele Irene

NixonVuele

URL de un proyecto en GitHub



<https://github.com/f4biaan/TallerGrupal1-ProgAvanzada>

El listado de los números generados.

```
import java.util
import java.util.concurrent.ThreadLocalRandom
import java.util.stream.IntStream

object tareagrupal extends App{

  val numbers = IntStream.generate(() => ThreadLocalRandom.current.nextInt( origin = 10_000, bound = 1_000_000)).distinct().
    limit( maxSize = 25).boxed.toList
  numbers.stream.forEach(System.out.println : Unit)
  println("Números generados: " + numbers.size())
}
```

```
===== Listado de los números generados =====
[649450, 627990, 248236, 590520, 951711, 663276, 95169, 933763, 846128, 240258, 311417, 164119, 869015, 659372, 935230, 409292, 681605, 528762, 441546, 944111, 528649, 740
Numeros generados: 25
```


El listado de los números generados ordenados ascendente y descendente

```
println("===== Orden Descendente y Ascendente =====")
println("Lista Descendente")
var ArrayDescendente = numbers.stream().sorted().toArray()
ArrayDescendente.forEach(println)
println("Lista Ascendente")
var ArrayAscendente = ArrayDescendente.reverse
ArrayAscendente.forEach(println)
```

```
===== Orden Descendente y Ascendente =====
Lista Descendente:
95169,164119,240258,248236,276234,307649,311417,409292,441546,528649,528762,590520,627990,649450,659372,663276,681605,746310,846128,869015,933763,935230,944111,946646,951711
Lista Ascendente:
951711,946646,944111,935230,933763,869015,846128,746310,681605,663276,659372,649450,627990,590520,528762,528649,441546,409292,311417,307649,276234,248236,240258,164119,95169
```

El listado de los números generados que son pares

```
val pares = numbers.stream().filter(x => x % 2 == 0)
println(pares.toList)
```

```
===== Números generados que son pares =====
[649450, 627990, 248236, 590520, 663276, 846128, 240258, 659372, 935230, 409292, 528762, 441546, 746310, 946646, 276234]
```

El listado de los números generados que son impares

```
val impares = numbers.stream().filter(x => x % 2 == 1)
println(impares.toList)
```

```
===== Números generados que son impares =====
[951711, 95169, 933763, 311417, 164119, 869015, 681605, 944111, 528649, 307649]
```

El listado de los números generados que son primos

```
val primo = numbers.stream().filter(n => (2 ≤ until (< n).forall(k => n % k != 0)))  
println(primo.toList)
```


El listado de los números generados que son deficientes

```
val deficientes = numbers.stream().filter(  
    x => (1 ≤ to ≤ x-1).filter(d => x % d == 0).sum < x  
)  
  
println(deficientes.toList)
```

===== Números generados que son deficientes =====

[649450, 248236, 951711, 95169, 933763, 846128, 311417, 164119, 869015, 935230, 409292, 681605, 944111, 528649, 946646, 307649]

Números generados que son perfectos

El listado de los números generados que son perfectos

```
val perfectos = numbers.stream().filter(  
    x => (1 ≤ to ≤ x-1).filter(d => x % d == 0).sum == x  
)  
  
println(perfectos.toList)
```

```
===== Números generados que son perfectos =====
```

```
[]
```

```
===== Números generados que son abundantes =====
```

El listado de los números generados que son abundantes.

```
val abundantes = numbers.stream().filter(  
    x => (1 ≤ to ≤ x-1).filter(d => x % d == 0).sum > x  
)  
  
println(abundantes.toList)
```

```
===== Números generados que son abundantes =====  
[627990, 590520, 663276, 240258, 659372, 528762, 441546, 746310, 276234]
```

Resultados

```
===== Listado de los números generados =====  
[649450, 627990, 248236, 590520, 951711, 663276, 95169, 933763, 846128, 240258, 311417, 164119, 869015, 659372, 935230, 409292, 681605, 528762, 441546, 944111, 528649, 746646, 951711]  
Numeros generados: 25
```

```
===== Orden Descendente y Ascendente =====  
Lista Descendente:  
95169,164119,240258,248236,276234,307649,311417,409292,441546,528649,528762,590520,627990,649450,659372,663276,681605,746310,846128,869015,933763,935230,944111,946646,951711  
Lista Ascendente:  
951711,946646,944111,935230,933763,869015,846128,746310,681605,663276,659372,649450,627990,590520,528762,528649,441546,409292,311417,307649,276234,248236,240258,164119,951711
```

```
===== Números generados que son pares =====  
[649450, 627990, 248236, 590520, 663276, 846128, 240258, 659372, 935230, 409292, 528762, 441546, 746310, 946646, 276234]
```

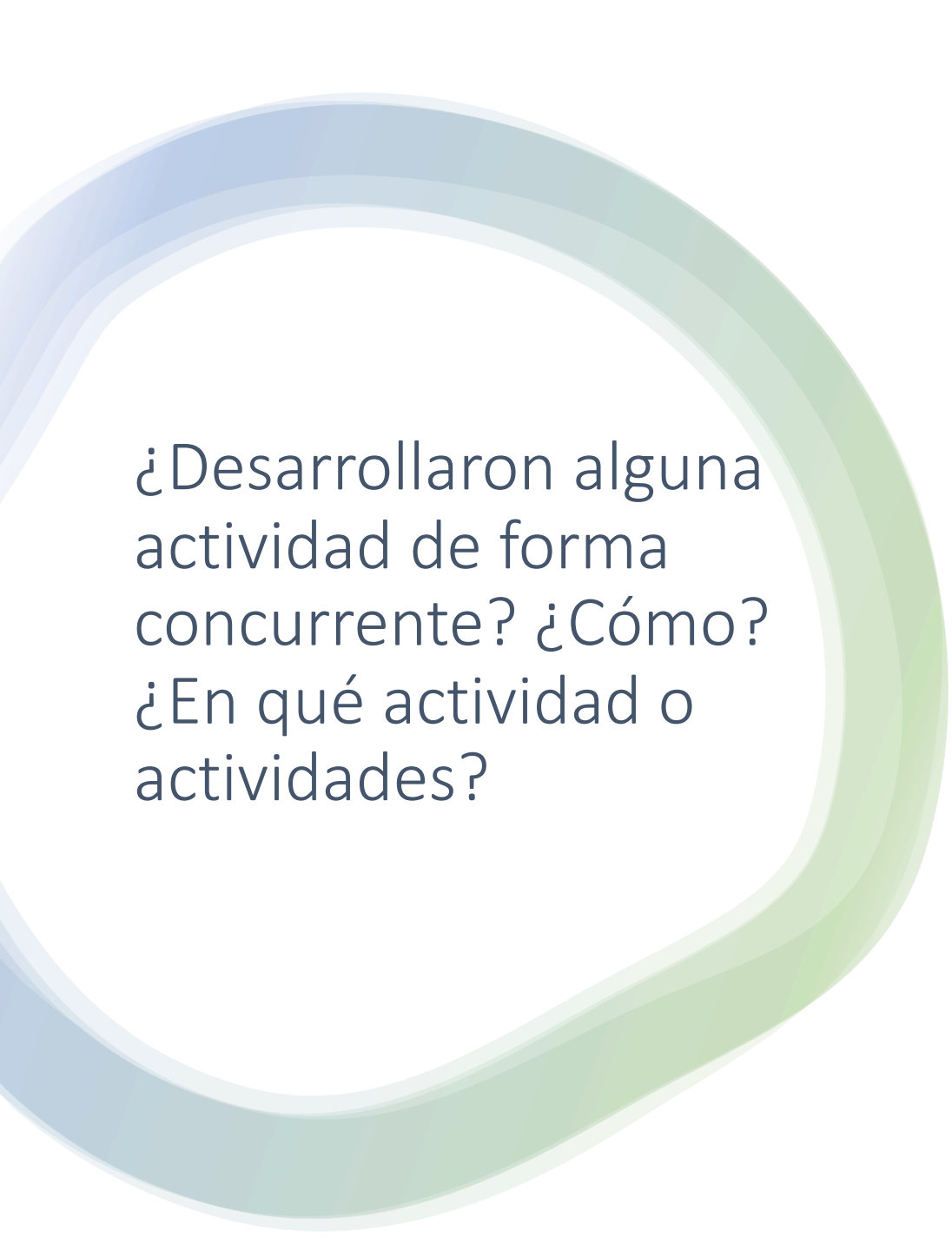
```
===== Números generados que son impares =====  
[951711, 95169, 933763, 311417, 164119, 869015, 681605, 944111, 528649, 307649]
```

```
===== Números generados que son primos =====  
[]
```

```
===== Números generados que son deficientes =====  
[649450, 248236, 951711, 95169, 933763, 846128, 311417, 164119, 869015, 935230, 409292, 681605, 944111, 528649, 946646, 307649]
```

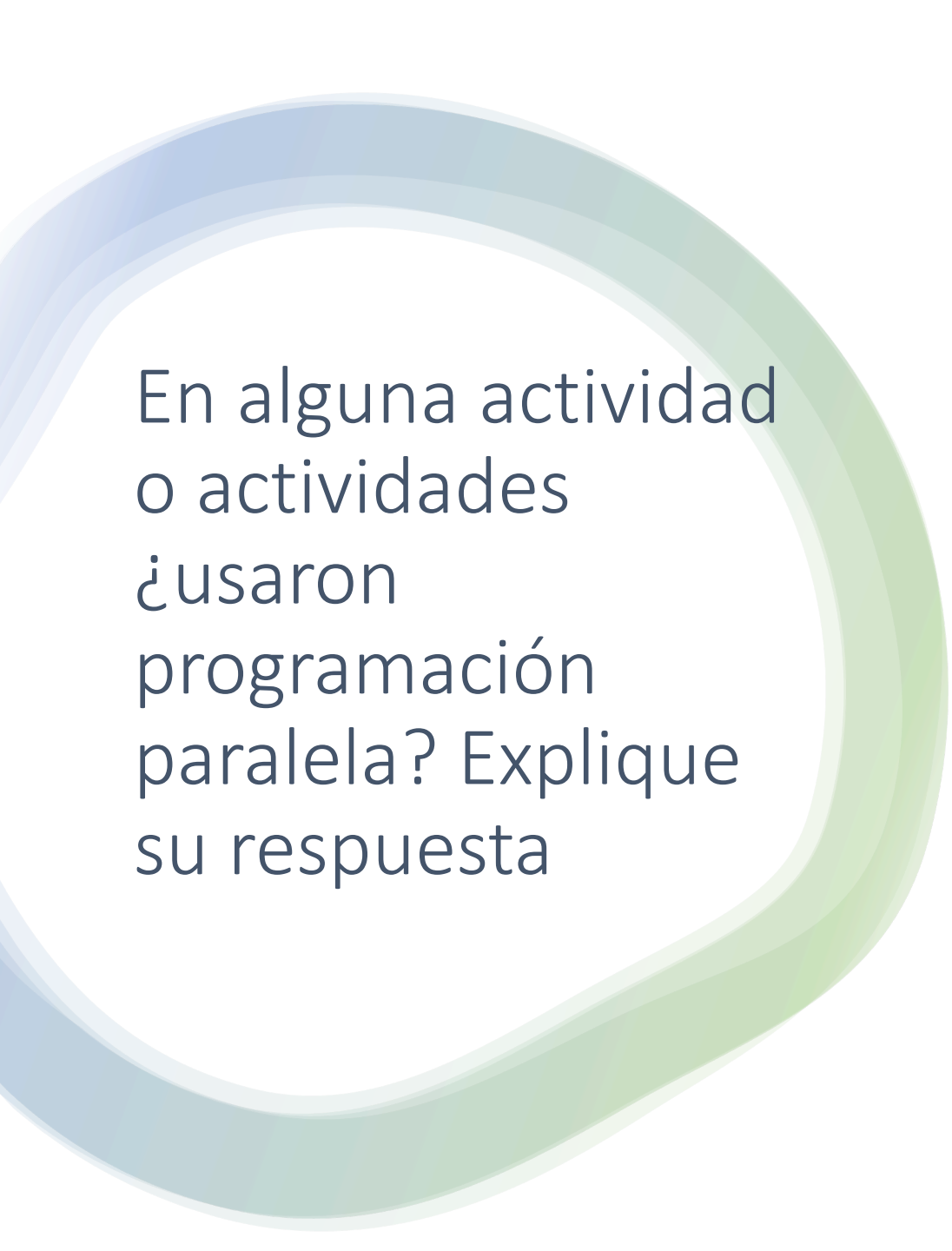
```
===== Números generados que son perfectos =====  
[]
```

```
===== Números generados que son abundantes =====  
[627990, 590520, 663276, 240258, 659372, 528762, 441546, 746310, 276234]
```



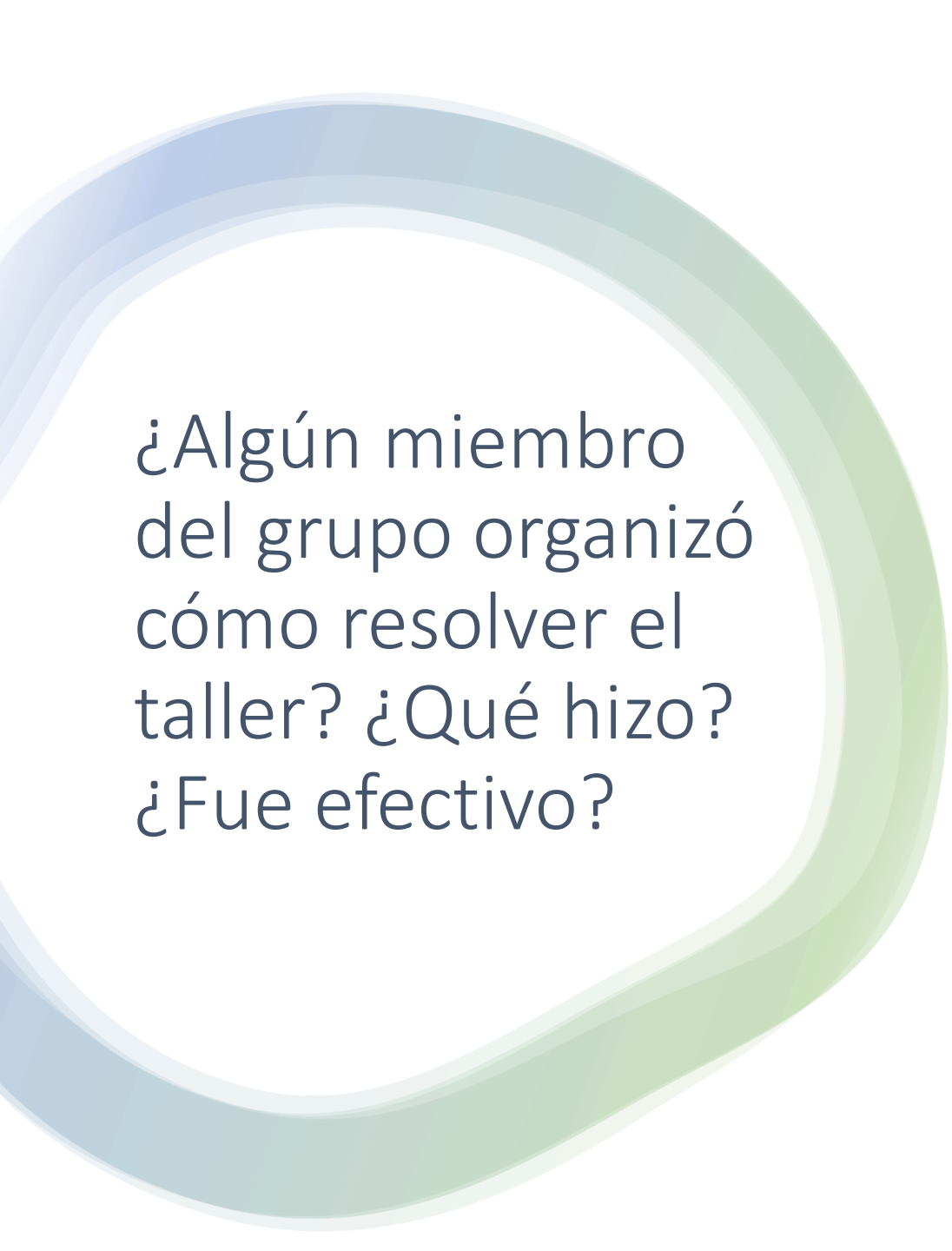
¿Desarrollaron alguna actividad de forma concurrente? ¿Cómo? ¿En qué actividad o actividades?

Si, se utilizó programación concurrente ya que todo el grupo colaboro de manera secuencial en una suerte de análisis sobre las actividades a efectuar y luego esperamos a realizar las configuraciones necesarias para trabajar en equipo. Finalmente se hizo/ uso de la programación concurrente al esperar de manera paulatina que el resto de compañeros comparta su aporte con su respectiva actividad para de esta forma poder terminar de realizar las diapositivas.



En alguna actividad
o actividades
¿usaron
programación
paralela? Explique
su respuesta

Se ha hecho de la programación paralela al momento de escribir los códigos correspondientes a ordenación de números, números pares, impares, primos, deficientes, perfectos y abundantes de manera que al mismo tiempo también se trabajaba en la interfaz y presentación de las diapositivas en realizadas en Power point.



¿Algún miembro del grupo organizó cómo resolver el taller? ¿Qué hizo? ¿Fue efectivo?

El compañero Fabian Montoya ha tomado el liderazgo del equipo asignando tareas a cada uno de los compañeros integrantes del grupo, cada uno según su capacidad haciendo de esta manera un trabajo ordenado y bastante eficiente.