# UT9 - Practico Domiciliario 2 - Rodrigo Perdomo

## Ejercicio 1 - Shellsort

1. Tres combinaciones de incrementos distintas:

a) Sedgewick (1986)
Secuencia: 1, 5, 19, 41, 109, ...
Referencia: Sedgewick, R. (1986). "A new upper bound for Shellsort". Journal of Algorithms, 7(1), 159-173.

b) Hibbard (1963)
Secuencia: 1, 3, 7, 15, 31, 63, ...
Referencia: Hibbard, T. (1963). "An empirical study of minimal storage sorting". Communications of the ACM, 6(5), 206-208.

c) Tokuda (1992)
Secuencia: 1, 4, 9, 20, 46, 103, ...
Referencia: Tokuda, N. (1992). "An Improved Shellsort". Information Processing Letters, 43(1), 11-16.

2. Análisis del tiempo de ejecución:
El tiempo de ejecución de Shellsort varía según la secuencia de incrementos.
- Peor caso general: entre O(n^2) y O(n log^2 n)
- Mejores secuencias como Tokuda o Sedgewick pueden alcanzar O(n^(4/3))

3. Mostrar ordenamiento del conjunto:
Conjunto: 256, 458, 655, 298, 043, 648, 778, 621, 655, 019, 124, 847
(Secuencia de pasos desarrollada en documento adicional o actividad práctica)

## Ejercicio 2 - Burbuja

1. Ordenamiento paso a paso del conjunto:
Conjunto: 44, 55, 12, 42, 94, 18, 6, 67

Iteraciones (resumen por filas):
i=1: 44, 12, 42, 55, 18, 6, 67, 94
i=2: 12, 42, 44, 18, 6, 55, 67, 94
i=3: 12, 42, 18, 6, 44, 55, 67, 94
i=4: 12, 18, 6, 42, 44, 55, 67, 94
i=5: 12, 6, 18, 42, 44, 55, 67, 94
i=6: 6, 12, 18, 42, 44, 55, 67, 94

Sí, el conjunto queda ordenado antes de la última iteración.

2. Mejoras posibles al algoritmo de Burbuja:

a) Cortar si no hay intercambios (bueno para listas casi ordenadas).
Pseudocódigo:

```
para i = 0 hasta n-1
    intercambiado = falso
    para j = 0 hasta n-i-2
```

```
        si arr[j] > arr[j+1]
            intercambiar(arr[j], arr[j+1])
            intercambiado = verdadero
    si no intercambiado
        romper
```

b) Optimización de rango del bucle con nuevo límite:

```
  limite = n-1
  repetir
      nuevoLimite = 0
      para j = 0 hasta limite-1
          si arr[j] > arr[j+1]
              intercambiar y nuevoLimite = j
      limite = nuevoLimite
  hasta que limite = 0
```

3. Las dos versiones anteriores se pueden aplicar paso a paso igual que en (1).

4. ShakerSort (Cocktail):
Es una variante que recorre de izquierda a derecha y luego de derecha a izquierda.
Cada pasada empuja el mayor al final y el menor al inicio.
Esto acelera la ordenación si los extremos están mal posicionados.