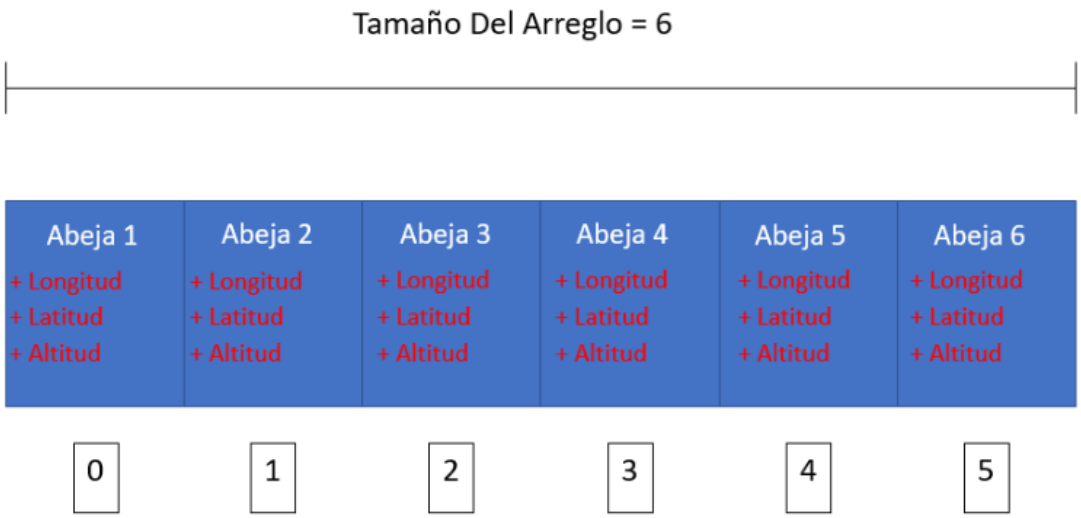
ALGORITMO PARA PREVENIR COLICIONES ENTRE ABEJAS ROBOTICAS

Juan Camilo Guerrero Alarcon
Santiago Pulgarin Vasquez

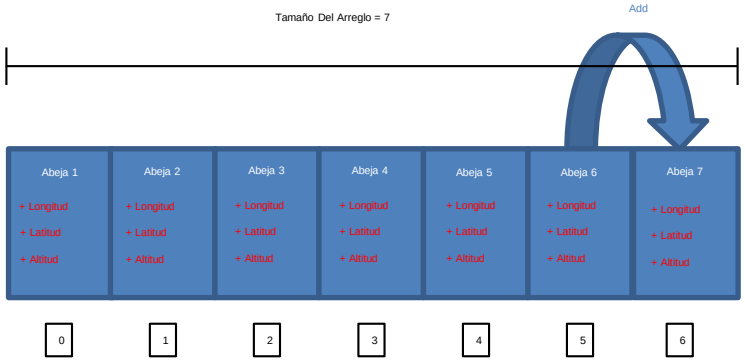
Medellín, 6 de Noviembre

Estructuras de Datos Diseñada



contiene

Operaciones de la Estructura de Datos



Gráfica 2: Imagen de una operación Add de un ArrayList.

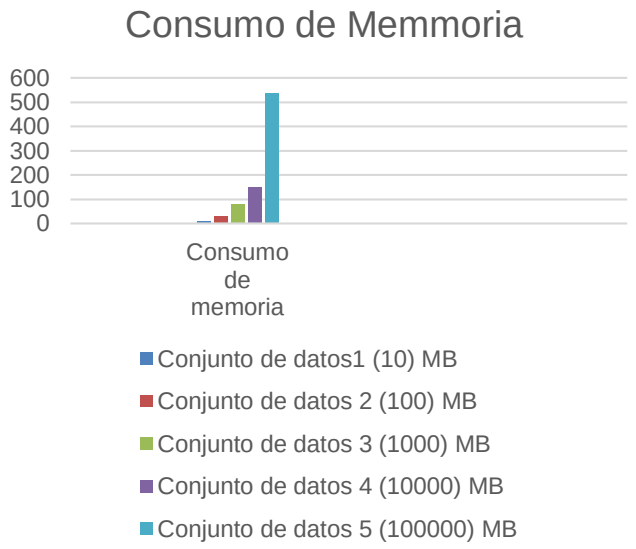
Tabla 1: Tabla para reportar la complejidad

Metodo	Complejidad
distancia()	O(n)
leerArchivo	O(n)
detectarColisiones()	O(n^2)
guardarArchivo()	O(n)
main()	O(1)

Criterios de Diseño de la Estructura de Datos

La razón por la cual escogimos la estructura de datos Arraylist es debido a que es una de las más usadas, más conocidas y fácil de implementar a tipos de problemas como este, además de su excelente implementación que facilita el análisis de los datos, aunque este tipo de estructura no es muy eficiente y nos genera un déficit en memoria y en tiempo de procesamiento lo que genera poca efectividad para grandes cantidades de datos.

Consumo de Tiempo y Memoria



	Mejor Tiempo (segundos)	Peor Tiempo (segundos)	Tiempo Promedio (segundos)	Mejor Memoria (MB)	Peor Memoria (MB)	Memoria Promedio (MB)
Conjunto de datos1 (10)	0	0	0	7	11	9
Conjunto de datos 2 (100)	0,001	0,003	0,001683168	20	40	30
Conjunto de datos 3 (1000)	0,011	0,015	0,013138614	60	100	80
Conjunto de datos 4 (10000)	0,319	0,39	0,327653465	100	200	150
Conjunto de datos 5 (100000)	18,46	20,786	19,54	400	680	540

Inserten sus propias gráficas y sus explicaciones

Software Desarrollado