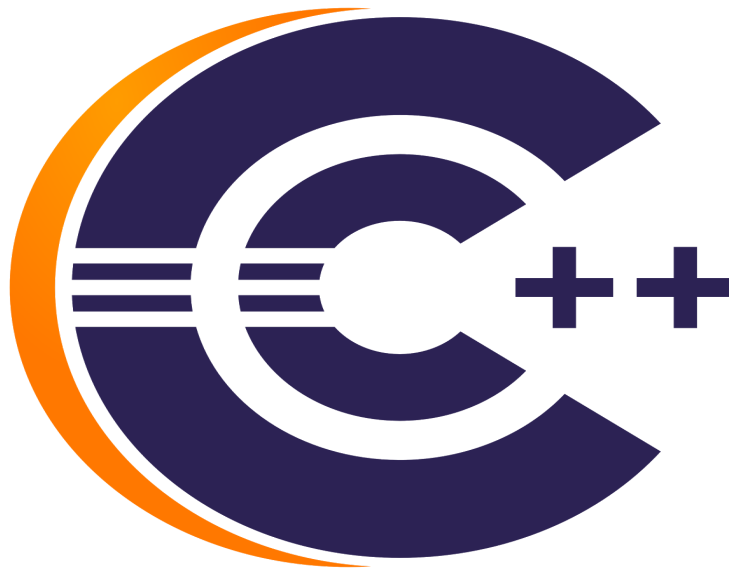




Universidad  
Católica del Norte



# Taller IV

*Paralelismo v/s A. secuenciales*

Integrantes:

-Carlos Cortes

-Ignacio Chirino

Estructura de Datos - Ayd. Ricardo Pizarro

# Introducción

En la computación existen diversas formas de abordar distintos problemas, a través de la programación, habitualmente cuando se programa, se tiende a pensar en una solución secuencial pero, ¿que tal si se puede mejorar y/o aprovechar de mejor manera los recursos disponibles? aquí emergen los términos *Paralelismo* y *Concurrencia*, que cabe destacar que no son lo mismo, pero van muy de la mano.

Cuando hablamos de *Paralelismo* nos referimos al uso extra de recursos para facilitar la realización de diversas tareas. Cuando hablamos de *Concurrencia* nos referimos al uso y manejo correcto de estos recursos, con tal de que no se entorpezca la implementación o que termine siendo más perjudicial que beneficiosos para nuestro objetivo.

# Desarrollo

## Modelamiento del Problema:

El taller consistía en implementar dos algoritmos de ordenamiento, tales como el Quick Sort, Bubble Sort, Merge Sort, Heap Sort, etc. cada uno en su versión secuencial y su versión con paralelismo. Para este taller se eligieron los algoritmos quickSort y mergeSort.

A continuación se procede a dejar una breve explicación de cada uno de los algoritmos que fueron utilizados en el taller.

- QuickSort:

Es un algoritmo que ordena mediante la selección de un “*pivote*” y mediante recursión y *dividir para conquistar* se llama al mismo método para seguir ordenando las partes restantes del arreglo.

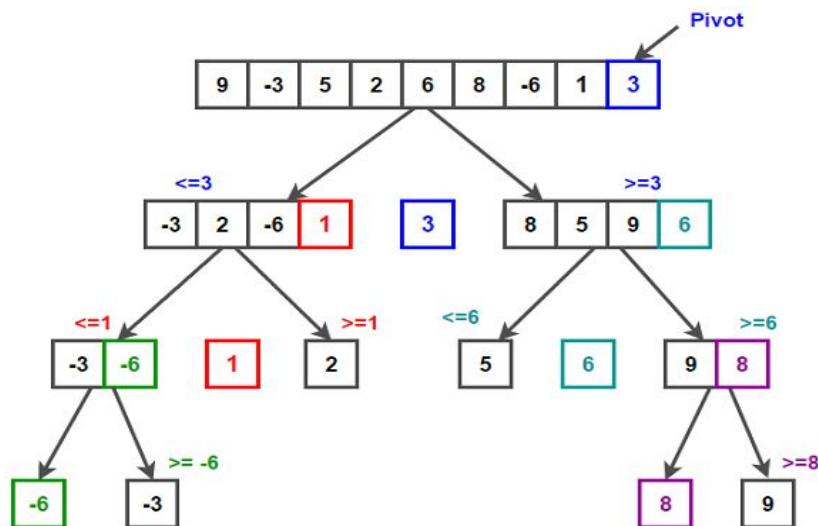


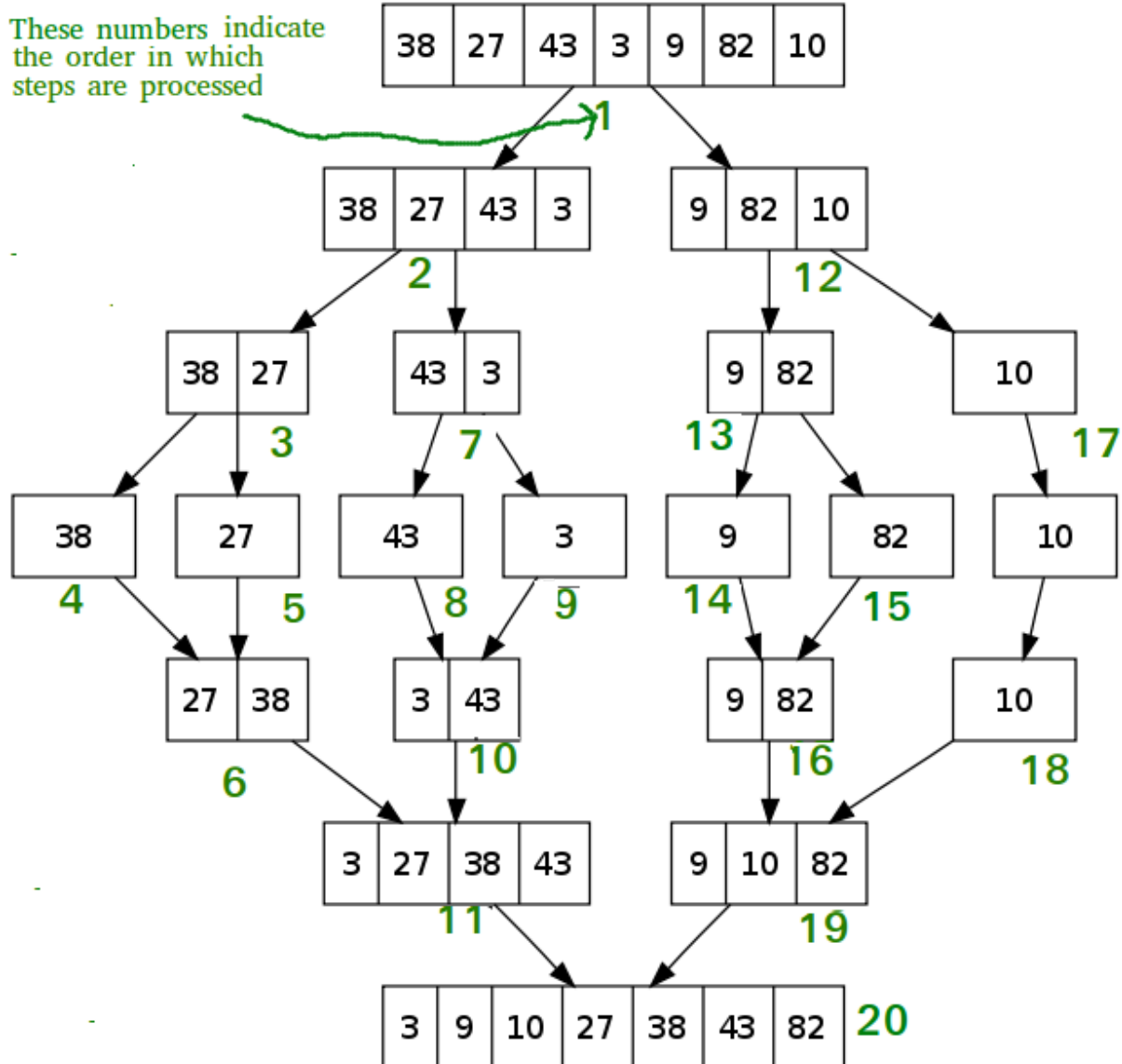
Ilustración quicksort, pivote inicial = 3, se subdivide el arreglo y por recursividad el pivote ira cambiando para cada subarreglo.

Este algoritmo posee implementaciones iterativas y recursivas.

- Merge Sort

Comparte ciertas similitudes con el quicksort, ya que los dos ocupan las técnicas de dividir y conquistar. Este divide un arreglo en dos partes y se llama a sí mismo por las dos mitades para luego combinar las dos mitades ordenadas.

El siguiente diagrama muestra el proceso completo del algoritmo merge sort para el ejemplo del arreglo {38,27,43,3,9,82,10}. Si se toma una mirada al diagrama, podemos ver que el arreglo se divide recursivamente en dos mitades hasta que el tamaño se convierte en 1. Una vez que el tamaño se convierte en 1, el proceso de fusión toma acción y empieza a fusionar las matrices hasta que se fusiona el arreglo completo.



## Quicksort y MergeSort con Paralelismo

En un principio, se planteó la idea de ir subdividiendo el arreglo como se vio en ayudantía mediante la función `main`, un `size` y un `grain`, el cual otorgaría particiones “exactas” del arreglo, y así poder implementar los hilos requeridos por el taller, pero no se pudo concretar esta idea.

Se implementó el `quicksort` y `mergesort` de manera similar, mediante dos hilos que ordenaban el arreglo desde el inicio hasta la mitad, como muestra la figura.

```
201 //Intento del metodo de quicksort con n hilos, no se llega a algo coherente//
262 void parallelQuicksort(int* arreglo, int lowerIndex, int highIndex, int nhebras, int largo) { ... }
302 /*Metodo Quicksort con paralelismo, se utilizan 2 hebras, cada una para una mitad del arreglo respectivo*/
303 void quicksortParrallel(int* arreglo, int ini, int fin, int pivot) {
304
305     int mitad = (ini + fin) / 2;
306
307     if (ini < fin) {
308
309         thread hebra1{&quickSortPivot, ref(arreglo), ini, mitad, pivot}; //Hebra que realiza el ordenamiento de la primera mitad del arreglo
310         thread hebra2{ &quickSortPivot, ref(arreglo), mitad+1, fin, pivot}; //Hebra que realiza el ordenamiento de la segunda mitad del arreglo
311
312         hebra2.join(); // -----> join
313         hebra1.join();
314
315     }
316
317 }
318
348
```

Mediante varias pruebas, el código mostró que los algoritmos iterativos curiosamente tardaban menos en ordenar los arreglos que los implementados mediante paralelismo.

En este taller no se modeló el programa mediante un diagrama de clases ni mediante POO, se implementó todo en la ventana principal, pero el `main` solamente llama al método `menu()`.

# Planificación

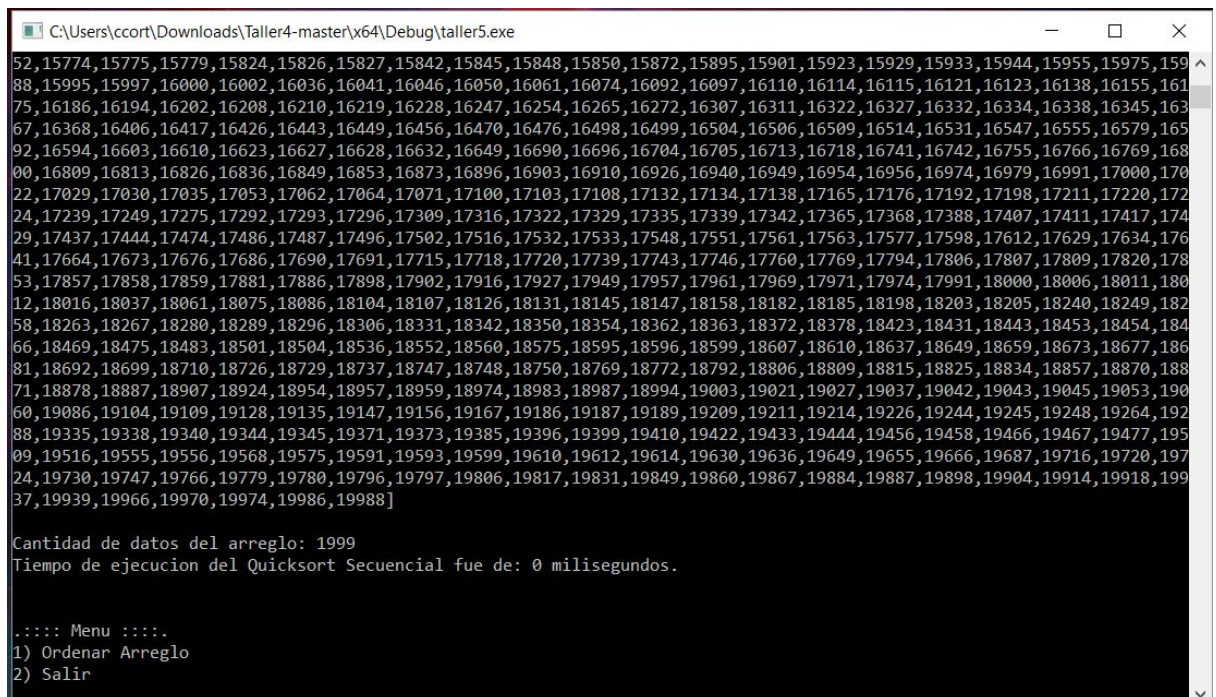
El taller se planificó con una semana aprox de antelación a la fecha de entrega, cabe destacar que esta última se extendió en tres días, lo cual fue una gran noticia para el grupo.

La división de las tareas fue bastante sencilla, cada uno eligió un algoritmo de los que se mencionó anteriormente e investigó sus implementaciones, y modelamos alguna idea para implementarlos mediante paralelismo.

- a) Fecha planificacion( 09/12/17): Se dividieron los temas a investigar entre los integrantes del grupo.
- b) Segunda Fecha(12/12/17): Se realizó una junta de alrededor dos horas con tal de complementar y discrepar acerca de las ideas investigadas y las posibles implementaciones de los algoritmos con paralelismo.
- c) Tercera Fecha(13/12/17): Esta junta fue para pasar las ideas a código, alrededor de tres horas, implementando el menú principal, los métodos asociados, y los algoritmos *secuenciales* de ordenamiento, se intentó la implementación de los *N hilos*.
- d) Cuarta Fecha(15/12/17): Tras varios intentos y desilusiones por no poder implementar los N hilos, se sugirió implementarlo tan solo con dos hebras para ambas mitades de cualquier arreglo generico, cabe destacar que en el código igual se pide ingresar las n hebras por consola, pero en realidad solo utiliza dos, esto fue para darle mejor aspecto al menu.
- e) Quinta Fecha(19/12/17): Se finiquito lo que mas pudo el código, se realizaron varias validaciones para los *input's*, y se realizó *la primera parte del informe*.
- f) Sexta Fecha(20/12/17): Última junta para probar el código mediante diversos archivos.txt de distinto largo, en la sección siguiente se especificarán las pruebas y las conclusiones que se sacaron de este estudio.

### Resultados ejecución del programa

En el siguiente se detallan los resultados entregados en la ejecución del programa, desde el archivo “datos.txt” se recopilaron los datos que corresponden a 2000 números enteros diferentes. Antes de tomar en cuenta los resultados, por motivos de no poder encontrar respuesta al inciso 2 el cual pedía que el usuario indicará la cantidad de hilos se tomo la decisión de dejar la cantidad de hilos predeterminada, cuya cantidad corresponde a dos. Las imágenes a continuación muestran el tiempo de cada uno de los item.



```
C:\Users\ccort\Downloads\Taller4-master\Debug\taller5.exe
52,15774,15775,15779,15824,15826,15827,15842,15845,15848,15850,15872,15895,15901,15923,15929,15933,15944,15955,15975,159
88,15995,15997,16000,16002,16036,16041,16046,16050,16061,16074,16092,16097,16110,16114,16115,16121,16123,16138,16155,161
75,16186,16194,16202,16208,16210,16219,16228,16247,16254,16265,16272,16307,16311,16322,16327,16332,16334,16338,16345,163
67,16368,16406,16417,16426,16443,16449,16456,16470,16476,16498,16499,16504,16506,16509,16514,16531,16547,16555,16579,165
92,16594,16603,16610,16623,16627,16628,16632,16649,16690,16696,16704,16705,16713,16718,16741,16742,16755,16766,16769,168
00,16809,16813,16826,16836,16849,16853,16873,16896,16903,16910,16926,16940,16949,16954,16956,16974,16979,16991,17000,170
22,17029,17030,17035,17053,17062,17064,17071,17100,17103,17108,17132,17134,17138,17165,17176,17192,17198,17211,17220,172
24,17239,17249,17275,17292,17293,17296,17309,17316,17322,17329,17335,17339,17342,17365,17368,17388,17407,17411,17417,174
29,17437,17444,17474,17486,17487,17496,17502,17516,17532,17533,17548,17551,17561,17563,17577,17598,17612,17629,17634,176
41,17664,17673,17676,17686,17690,17691,17715,17718,17720,17739,17743,17746,17760,17769,17794,17806,17807,17809,17820,178
53,17857,17858,17859,17881,17886,17898,17902,17916,17927,17949,17957,17961,17969,17971,17974,17991,18000,18006,18011,180
12,18016,18037,18061,18075,18086,18104,18107,18126,18131,18145,18147,18158,18182,18185,18198,18203,18205,18240,18249,182
58,18263,18267,18280,18289,18296,18306,18331,18342,18350,18354,18362,18363,18372,18378,18423,18431,18443,18453,18454,184
66,18469,18475,18483,18501,18504,18536,18552,18560,18575,18595,18596,18599,18607,18610,18637,18649,18659,18673,18677,186
81,18692,18699,18710,18726,18729,18737,18747,18748,18750,18769,18772,18792,18806,18809,18815,18825,18834,18857,18870,188
71,18878,18887,18907,18924,18954,18957,18959,18974,18983,18987,18994,19003,19021,19027,19037,19042,19043,19045,19053,190
60,19086,19104,19109,19128,19135,19147,19156,19167,19186,19187,19189,19209,19211,19214,19226,19244,19245,19248,19264,192
88,19335,19338,19340,19344,19345,19371,19373,19385,19396,19399,19410,19422,19433,19444,19456,19458,19466,19467,19477,195
09,19516,19555,19556,19568,19575,19591,19593,19599,19610,19612,19614,19630,19636,19649,19655,19666,19687,19716,19720,197
24,19730,19747,19766,19779,19780,19796,19797,19806,19817,19831,19849,19860,19867,19884,19887,19898,19904,19914,19918,199
37,19939,19966,19970,19974,19986,19988]

Cantidad de datos del arreglo: 1999
Tiempo de ejecucion del Quicksort Secuencial fue de: 0 milisegundos.

....: Menu :....
1) Ordenar Arreglo
2) Salir
```

Resultados algoritmo Quicksort iterativo -- 0 milisegundos.



```
C:\Users\ccort\Downloads\Taller4-master\x64\Debug\taller5.exe
52,15774,15775,15779,15824,15826,15827,15842,15845,15848,15850,15872,15895,15901,15923,15929,15933,15944,15955,15975,159
88,15995,15997,16000,16002,16036,16041,16046,16050,16061,16074,16092,16097,16110,16114,16115,16121,16123,16138,16155,161
75,16186,16194,16202,16208,16210,16219,16228,16247,16254,16265,16272,16307,16311,16322,16327,16332,16334,16338,16345,163
67,16368,16406,16417,16426,16443,16449,16456,16470,16476,16498,16499,16504,16506,16509,16514,16531,16547,16555,16579,165
92,16594,16603,16610,16623,16627,16628,16632,16649,16690,16696,16704,16705,16713,16718,16741,16742,16755,16766,16769,168
00,16809,16813,16826,16836,16849,16853,16873,16896,16903,16910,16926,16940,16949,16954,16956,16974,16979,16991,17000,170
22,17029,17030,17035,17053,17062,17064,17071,17100,17103,17108,17132,17134,17138,17165,17176,17192,17198,17211,17220,172
24,17239,17249,17275,17292,17293,17296,17309,17316,17322,17329,17335,17339,17342,17365,17368,17388,17407,17411,17417,174
29,17437,17444,17474,17486,17487,17496,17502,17516,17532,17533,17548,17551,17561,17563,17577,17598,17612,17629,17634,176
41,17664,17673,17676,17686,17690,17691,17715,17718,17720,17739,17743,17746,17760,17769,17794,17806,17807,17809,17820,178
53,17857,17858,17859,17881,17886,17898,17902,17916,17927,17949,17957,17961,17969,17971,17974,17991,18000,18006,18011,180
12,18016,18037,18061,18075,18086,18104,18107,18126,18131,18145,18147,18158,18182,18185,18198,18203,18205,18240,18249,182
58,18263,18267,18280,18289,18296,18306,18331,18342,18350,18354,18362,18363,18372,18378,18423,18431,18443,18453,18454,184
66,18469,18475,18483,18501,18504,18536,18552,18560,18575,18595,18596,18599,18607,18610,18637,18649,18659,18673,18677,186
81,18692,18699,18710,18726,18729,18737,18747,18748,18750,18769,18772,18792,18806,18809,18815,18825,18834,18857,18870,188
71,18878,18887,18907,18924,18954,18957,18959,18974,18983,18987,18994,19003,19021,19027,19037,19042,19043,19045,19053,190
60,19086,19104,19109,19128,19135,19147,19156,19167,19186,19187,19189,19209,19211,19214,19226,19244,19245,19248,19264,192
88,19335,19338,19340,19344,19345,19371,19373,19385,19396,19399,19410,19422,19433,19444,19456,19458,19466,19467,19477,195
09,19516,19555,19556,19568,19575,19591,19593,19599,19610,19612,19614,19630,19636,19649,19655,19666,19687,19716,19720,197
24,19730,19747,19766,19779,19780,19796,19797,19806,19817,19831,19849,19860,19867,19884,19887,19898,19904,19914,19918,199
37,19939,19966,19970,19974,19986,19988]

Cantidad de datos del arreglo: 1999
Tiempo de ejecucion del Quicksort con Paralelismo fue de: 8 milisegundos.

..... Menu .....
1) Ordenar Arreglo
2) Salir
```

Resultados algoritmo Quicksort paralelismo -- 8 milisegundos

```
C:\Users\ccort\Downloads\Taller4-master\x64\Debug\taller5.exe
52,15774,15775,15779,15824,15826,15827,15842,15845,15848,15850,15872,15895,15901,15923,15929,15933,15944,15955,15975,159
88,15995,15997,16000,16002,16036,16041,16046,16050,16061,16074,16092,16097,16110,16114,16115,16121,16123,16138,16155,161
75,16186,16194,16202,16208,16210,16219,16228,16247,16254,16265,16272,16307,16311,16322,16327,16332,16334,16338,16345,163
67,16368,16406,16417,16426,16443,16449,16456,16470,16476,16498,16499,16504,16506,16509,16514,16531,16547,16555,16579,165
92,16594,16603,16610,16623,16627,16628,16632,16649,16690,16696,16704,16705,16713,16718,16741,16742,16755,16766,16769,168
00,16809,16813,16826,16836,16849,16853,16873,16896,16903,16910,16926,16940,16949,16954,16956,16974,16979,16991,17000,170
22,17029,17030,17035,17053,17062,17064,17071,17100,17103,17108,17132,17134,17138,17165,17176,17192,17198,17211,17220,172
24,17239,17249,17275,17292,17293,17296,17309,17316,17322,17329,17335,17339,17342,17365,17368,17388,17407,17411,17417,174
29,17437,17444,17474,17486,17487,17496,17502,17516,17532,17533,17548,17551,17561,17563,17577,17598,17612,17629,17634,176
41,17664,17673,17676,17686,17690,17691,17715,17718,17720,17739,17743,17746,17760,17769,17794,17806,17807,17809,17820,178
53,17857,17858,17859,17881,17886,17898,17902,17916,17927,17949,17957,17961,17969,17971,17974,17991,18000,18006,18011,180
12,18016,18037,18061,18075,18086,18104,18107,18126,18131,18145,18147,18158,18182,18185,18198,18203,18205,18240,18249,182
58,18263,18267,18280,18289,18296,18306,18331,18342,18350,18354,18362,18363,18372,18378,18423,18431,18443,18453,18454,184
66,18469,18475,18483,18501,18504,18536,18552,18560,18575,18595,18596,18599,18607,18610,18637,18649,18659,18673,18677,186
81,18692,18699,18710,18726,18729,18737,18747,18748,18750,18769,18772,18792,18806,18809,18815,18825,18834,18857,18870,188
71,18878,18887,18907,18924,18954,18957,18959,18974,18983,18987,18994,19003,19021,19027,19037,19042,19043,19045,19053,190
60,19086,19104,19109,19128,19135,19147,19156,19167,19186,19187,19189,19209,19211,19214,19226,19244,19245,19248,19264,192
88,19335,19338,19340,19344,19345,19371,19373,19385,19396,19399,19410,19422,19433,19444,19456,19458,19466,19467,19477,195
09,19516,19555,19556,19568,19575,19591,19593,19599,19610,19612,19614,19630,19636,19649,19655,19666,19687,19716,19720,197
24,19730,19747,19766,19779,19780,19796,19797,19806,19817,19831,19849,19860,19867,19884,19887,19898,19904,19914,19918,199
37,19939,19966,19970,19974,19986,19988]

Cantidad de datos del arreglo: 1999
Tiempo de ejecucion del Merge sort iterativo fue de: 1 milisegundos.

..... Menu .....
1) Ordenar Arreglo
2) Salir
```

Resultados algoritmo Mergesort Iterativo -- 1 milisegundo.



```
C:\Users\ccort\Downloads\Taller4-master\x64\Debug\taller5.exe
33,15561,15568,15576,15610,15612,15616,15622,15628,15629,15644,15650,15662,15687,15709,15715,15717,15735,15737,15741,157
52,15774,15775,15779,15824,15826,15827,15842,15845,15848,15850,15872,15895,15901,15923,15929,15933,15944,15955,15975,159
88,15995,15997,16000,16002,16036,16041,16046,16050,16061,16074,16092,16097,16110,16114,16115,16121,16123,16138,16155,161
75,16186,16194,16202,16208,16210,16219,16228,16247,16254,16265,16272,16307,16311,16322,16327,16332,16334,16338,16345,163
67,16368,16406,16417,16426,16443,16449,16456,16470,16476,16498,16499,16504,16506,16509,16514,16531,16547,16555,16579,165
92,16594,16603,16610,16623,16627,16628,16632,16649,16690,16696,16704,16705,16713,16718,16741,16742,16755,16766,16769,168
00,16809,16813,16826,16836,16849,16853,16873,16896,16903,16910,16926,16940,16949,16954,16956,16974,16979,16991,17000,170
22,17029,17030,17035,17053,17062,17064,17071,17100,17103,17108,17132,17134,17138,17165,17176,17192,17198,17211,17220,172
24,17239,17249,17275,17292,17293,17296,17309,17316,17322,17329,17335,17339,17342,17365,17368,17388,17407,17411,17417,174
29,17437,17444,17474,17486,17487,17496,17502,17516,17532,17533,17548,17551,17561,17563,17577,17598,17612,17629,17634,176
41,17664,17673,17676,17686,17690,17691,17715,17718,17720,17739,17743,17746,17760,17769,17794,17806,17807,17809,17820,178
53,17857,17858,17859,17881,17886,17898,17902,17916,17927,17949,17957,17961,17969,17971,17974,17991,18000,18006,18011,180
12,18016,18037,18061,18075,18086,18104,18107,18126,18131,18145,18147,18158,18182,18185,18198,18203,18205,18240,18249,182
58,18263,18267,18280,18289,18296,18306,18331,18342,18350,18354,18362,18363,18372,18378,18423,18431,18443,18453,18454,184
66,18469,18475,18483,18501,18504,18536,18552,18560,18575,18595,18596,18599,18607,18610,18637,18649,18659,18673,18677,186
81,18692,18699,18710,18726,18729,18737,18747,18748,18750,18769,18772,18792,18806,18809,18815,18825,18834,18857,18870,188
71,18878,18887,18907,18924,18954,18957,18959,18974,18983,18987,18994,19003,19021,19027,19037,19042,19043,19045,19053,190
60,19086,19104,19109,19128,19135,19147,19156,19167,19186,19187,19189,19209,19211,19214,19226,19244,19245,19248,19264,192
88,19335,19338,19340,19344,19345,19371,19373,19385,19396,19399,19410,19422,19433,19444,19456,19458,19466,19467,19477,195
09,19516,19555,19556,19568,19575,19591,19593,19599,19610,19612,19614,19630,19636,19649,19655,19666,19687,19716,19720,197
24,19730,19747,19766,19779,19780,19796,19797,19806,19817,19831,19849,19860,19867,19884,19887,19898,19904,19914,19918,199
37,19939,19966,19970,19974,19986,19988]

Tiempo de ejecucion del Merge sort con paralelismo fue de: 7007 milisegundos.

.:.: Menu :.:.
1) Ordenar Arreglo
2) Salir
```

Resultados algoritmo Mergesort Paralelismo -- 7007 milisegundos.

Tabla Quicksort vs Mergesort:

Tiempo (milisegundos)	Quicksort	Mergesort
Iterativo	0	1
Paralelismo	8	7007

# Conclusión

Como conclusión de este estudio, hemos podido notar que el paralelismo es una gran herramienta para facilitar y optimizar las implementaciones de los codigos, pero solo algunas veces puede realmente ser útil para nuestros requerimientos, tal como lo vimos en los algoritmos de ordenamiento, Mergesort era el que mas tarda en ordenar, al ser recursivo debía concretar muchos sub-procesos de manera recursiva mediante hebras, lo cual provocó entorpecimiento en la ejecución exitosa del código, sumando a esto la parte de la mezcla también demoraba más la ejecución. Por otra parte, el *quickSort* no tardaba tanto como el Merge Sort, hablando de la implementación con paralelismo, al no hacer una mezcla, este algoritmo no causa tanta demora.

Pudimos ver que tan solo con dos hebras, se volvia mas lento el procedimiento, si se hubiera concretado de manera exitosa la implementación de las N hebras, hubieramos visto que con más hebras se ralentizaba mucho mas, como aprendizaje esperado pudimos captar que para utilizar bien estas técnicas, se debe tener un buen modelamiento y planteamiento del problema, para así poder utilizar el paralelismo con el fin de optimizar el tiempo de ejecución del código.