Operaciones entre archivos.

Apareo.

Se tiene un archivo Maestro y uno o más archivos de Novedades. Los archivos están ordenados por la misma clave y tienen el mismo formato de registro, con la salvedad de que el registro del archivo de Novedades incluye el código de la operación a realizar (A: alta, B: baja, M:modificación).
Algoritmo:
Busco mínimo, si el mínimo es:

Maestro	Grabo en salida registro M, avanzo M.
Novedades	A: grabo en salida registro N, avanzo N. B: informo error de borrado, avanzo N. M: informo error de modificación, avanzo N.
Ambos	A: informo error de inserción, avanzo N y M. B: avanzo N y M. M: grabo en salida registro N, avanzo N y M.

Cuando se tienen varios archivos de novedades el registro se actualiza en memoria y recién se graba en el archivo de salida una vez que cambia la clave mínima. Hay que tener en cuenta los criteríos para procesar novedades sobre la misma clave.

Intersección.

Se tienen dos o más archivos con registros de igual formato, y ordenados por la misma clave, y se quiere grabar en el archivo de salida únicamente los registros que figuren en todos los archivos de entrada.

Algoritmo:

Busco Mínimo	Al menos 1 archivo sin clave Mínima.	Avanzo los que tienen clave minima.
	Todos los archivos con clave mínima	Grabo el registro en salida y avanzo todos.

Cuando uno de los archivos llega al EOF implica la finalización del proceso.

Unión.

Se tienen dos o más archivos con registros de igual formato, y ordenados por la misma clave, y se quiere grabar en el archivo de salida únicamente los registros que figuren en todos los archivos de entrada.

Algoritmo:

Busco Mínimo

En los archivos con clave mínima	Grabo el registro en salida y los avanzo.

Cuando todos los archivos llegan al EOF implica la finalización del proceso.

Merge.

Se tiene un determinado número de archivos (particiones) ordenados por la misma clave. Se quiere obtener un único archivo ordenado por la clave.

Merge Balanceado N-Vías.

Se pueden tener N archivos abiertos, se usan N/2 archivos de Output y el resto de Input. Si N es impar la cantidad de archivos de Input es mayor que la de Output en uno.

Ejemplo:

N = 6, Cantidad de particiones: 13.

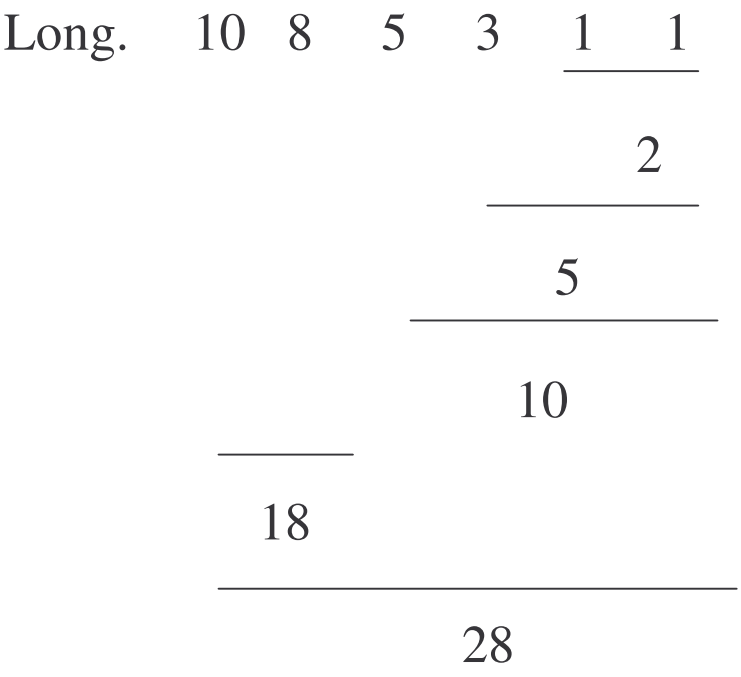
Input 1	P01	P04	P07	P10	P13			P01-09	
Input 2	P02	P05	P08	P11				P10-13	
Input 3	P02	P06	P09	P12					
Output 1						P01-03	P10-12		P01-13
Output 2						P04-06	P13		
Output 3						P07-09			

Una partición se lee múltiples veces.

Optimal Merge.

Mantiene una lista ordenada de todas las particiones según el tamaño de la partición. Se pueden tener abiertos N archivos. Se toman las N – 1 particiones de menor tamaño para Input y 1 partición de Output. Las particiones pequeñas son las que van a ser leídas más de una vez (de ser necesario), y se deja las particiones de mayor tamaño para el final.

Ejemplo:
N = 3, Cantidad de particiones: 6



Ventajas: Al tener en cuenta las longitudes se leen menos registros.
Desventajas: Mantenimiento de la lista.

Merge Polifásico.

Se pueden tener abiertos N archivos. Se toman N – 1 particiones para Input y 1 partición de Output. La cantidad de particiones tiene que pertenecer a la serie de Fibonacci de orden N-1

Particiones Dummy (Para llevar el número de particiones a un elemento de la serie de Fibonacci).

Para 32 particiones de tamaño n (4 vías): 57-32 = 25 dummies.

f(3) = 1 1 1 3 5 9 17 31 57

20 (9(0), 11(n))	24 (8(0), 16(n))	0	13 (8(0), 5(n))
7 (7(n))	11 (11(n))	13 (8(0), 1(2n), 4(3n))	0
0	4 (4(n))	6 (1(0), 1(2n), 4(3n))	7 (7(2n))
4 (1(3n), 1(5n), 2(6n))	0	2 (2(3n))	3 (3(2n))
2 (2(6n))	2 (1(8n), 1(10n))	0	1 (1(2n))
1 (1(6n))	1 (1(10n))	1 (1(16n))	0
0	0	0	1 (1(32n))

1	0	0	0	1
3	1	1	1	0
5	2	2	0	1
9	4	0	2	3
17	0	4	6	7
31	7	11	13	0
57	20	24	0	13

Lecturas:

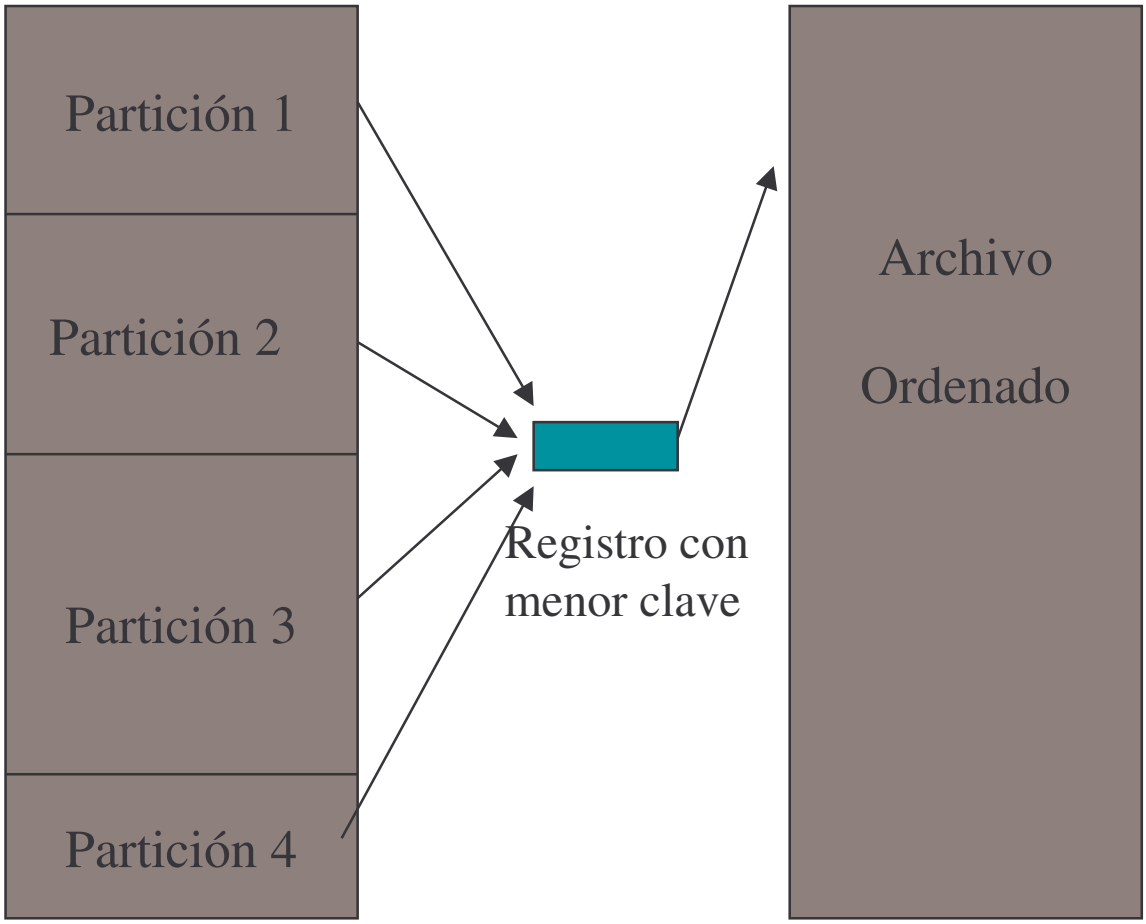
13*2n + 4*3n
7*2n
1*3n + 1*5n + 2*6n
1*8n + 1*10n
1*16n
1*32n

One Way Merge.

Todas las particiones (con sus datos ordenados) están grabadas en un mismo archivo.

Se cuenta con la información de donde comienza y donde termina cada partición.

Se realiza el mismo procedimiento de un merge convencional pero tomando los registros de un mismo archivo en lugar de muchos.
Se usa con archivos relativos o streams y solo requiere 2 vias (una de entrada y una de salida)



Ordenamiento de Archivos

Se tiene un archivo desordenado y se desea obtener el mismo archivo ordenado por uno o más campos.

Sort Interno.

Se carga el archivo completo en memoria, se ordena y se graba nuevamente. Solo es posible usarlo cuando el archivo entra completo en memoria

Key-Sort.

Para evitar el problema del método anterior, se cargan en memoria solo las claves por la que se ordena, se hace el sort de las mismas y se graba el nuevo archivo utilizando el ordenamiento dado por las claves. Además de lasclaves deberá cargarse una referencia al registro.
Solo sirve para Relativos y Streams.

Sort Interno + Merge.

Se carga el archivo parcialmente en memoria, se ordena y se graban copias parciales (particiones). Luego se necesita un proceso de Merge para obtener el archivo total ordenado.

Existen varios métodos que utilizan esta aproximación, buscando todos minimizar la cantidad de particiones.

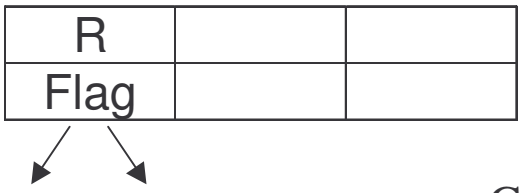
Primer método

Mientras haya registros.
    Leo a memoria
    Mientras (haya registros) y (memoria libre > Long(registro))
        Leo registros
    Fin Mientras
    Ordeno memoria
    Grabo partición
Fin Mientras

Una vez finalizado, quedarán generadas particiones del mismo tamaño, con excepción, quizás de la última, por lo que suele usarse un merge polifásico par obtener el archivo ordenado final.

Replacement Selection.

Genera particiones de tamaño = 2*(Memoria Libre), en el caso medio.
Es mejor que los métodos anteriores, donde la partición es tan grande como el tamaño de la memoria disponible.



C D

C: Congelado
D: Disponible

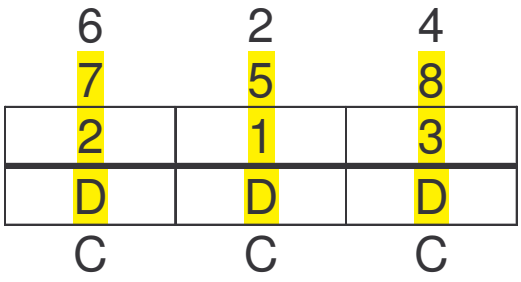
Algoritmo:

Llenar la memoria
Buscar la menor clave en la partición, entre los registros disponibles (no congelados)
Grabar el registro en el disco
Leer el siguiente registro del archivo e insertarlo en el lugar del registro grabado, si la clave es menor ponerle el flag de congelado.

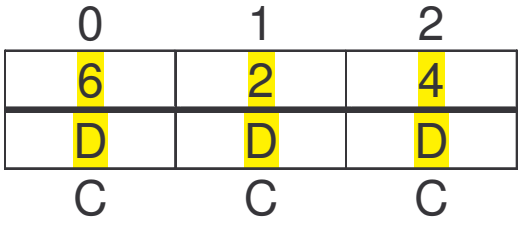
Esto se repite hasta que no queden registros o hasta que todos los registros de la partición estén congelados. Al tener toda la partición congelada se cierra el archivo de salida, se abre otro se descongela la partición y se repite el método.

Ejemplo de Replacement Selection:

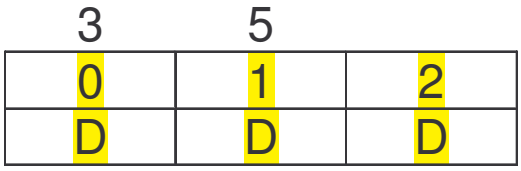
2, 1, 3, 5, 7, 8, 2, 6, 4, 1, 2, 0, 3, 5



1, 2, 3, 5, 7, 8 /



2, 4, 6 /



0, 1, 2, 3, 5 /

Mejor Caso: Si el archivo ya estaba ordenado lo único que hace es pasarlo, genera una única partición que contiene todo el archivo.
Peor Caso: Si el archivo está ordenado en el órden inverso (rebatido) genera particiones del tamaño de la memoria disponible, como lo haría un sort interno.

Natural Selection.

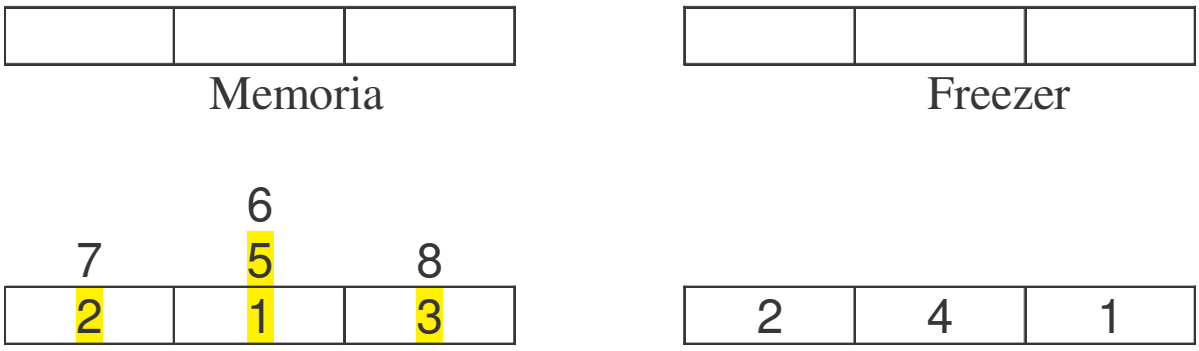
Genera particiones de tamaño = 3*(Memoria Libre), en el caso medio.
En vez de guardar los registros congelados con un flag los va guardando en un archivo en disco (Freezer).
El tamaño del freezer debe ser, a lo sumo, del mismo tamaño que la memoria utilizada para ordenamiento.
Cuando se llena el freezer los registros se pasan a la memoria.

Algoritmo:
Llenar la memoria
Buscar la menor clave en la partición en memoria
Grabar el registro en el disco
Leer el siguiente registro del archivo
Si es mayor que el menor que el registro grabado, se lo incluye en la partición en memoria
Si es menor se graba en el freezer.

Esto se repite hasta que no queden registros o hasta que haya que grabar un registro en el freezer y esté lleno.
Si el freezer está lleno voy tomando el menor registro de la memoria y lo voy grabando en el archivo de salida hasta vaciarla. Luego se cierra la partición, y se graban todos los registros del freezer a la memoria y se vuelve a repetir el método hasta agotar los registros.
Si al terminar el archivo quedan registros en el freezer, se pasan a memoria y se graban ordenados en una última partición

Ejemplo de Natural Selection:

2, 1, 3, 5, 7, 8, 2, 6, 4, 1, 2, 0, 3, 5



1, 2, 3, 5, 6, 7, 8 /

1, 2, 2, 3, 4, 5 /

0 /

Mejor Caso: Si el archivo ya estaba ordenado lo único que hace es pasarlo, genera una única partición que contiene todo el archivo.
Peor Caso: Si el archivo está ordenado en el órden inverso (rebatido) genera particiones del tamaño de la memoria disponible, como lo haría un sort interno.

Sort de Tarjetas.

Se separan los registros del archivo en particiones de acuerdo a los diferentes bytes que conforman la clave. Primero se tiene en cuenta el último, luego el penúltimo, el antepenúltimo, y así hasta el primero. Al final del proceso queda el archivo ordenado.

Ejemplo Sort de Tarjetas:

1015	5	8	3	2	9
3058					
1263	1015	3058	1263	1492	1969
1492			4083	5012	
1969					
4083					
5012					

Como punto de partida del paso siguiente se toman las particiones obtenidas ordenadas por número.

1492	9	1	6	8	5
5012					
1263	1492	5012	1263	4083	3058
4083		1015	1969		
1015					
3058					
1969					

Ejemplo Sort de Tarjetas (continuación) :

5012	0	2	9	4
1015				
3058	5012	1263	1969	1492
1263	1015			
1969	3058			
4083	4083			
1492				

5012	5	1	3	4
1015				
3058	5012	1015	3058	4083
4083		1263		
1263		1492		
1492		1969		
1969				

1015, 1263, 1492, 1969, 3058, 4083, 5012

Método del Cartero.

Se generan tantas particiones como valores posibles tenga el primer byte de la clave. Luego se toma cada partición y se mira el segundo byte de la clave, si es que hay más de un registro dentro de cada partición.

1015	A:	1015	AA:	1015	AAA:	1015
3058		1025		1025	AAB:	1025
1263		1492	AB:	1492		
1492		1969	AC:	1969		
1969	B:	3058				
4083	C:	4083				
5012	D:	5012				

1015, 1263, 1492, 1969, 3058, 4083, 5012

En ambos métodos hay un costo de mantener ordenadas las particiones. El método del cartero nunca vuelve a comparar claves que ya ordenó, y tarda menos según las características del archivo. El sort de tarjetas tarda siempre lo mismo, aún cuando el archivo ya esté ordenado.