Mergesort

Mergesort er en rekursiv algoritme, som bruker "divide and conquer" for sortering.

Mergesort vil først dele opp arrayen i flere deler, før den sorter hver sub-array.

Dette fører til at etter hver gang en sub-arrays venstre og høyre side blir sortert, så vil den som kallet mergesort for disse "merge" disse to delene sammen.

Mergesort er en out-of-place sorting algoritme, det vil si at den bruker en "hjelpe" array for å sortere.

Mergesort er stable, det vil si at keys av samme verdi (eks E og E) vil ikke bytte plass.

Forklaringer

- *i Itererer oppover, sammenlignes med *k
- *j Itererer nedover, sammenlignes med *k
- *k Den indeksen jeg skal sette inn i a[] arrayen.

Markerer b[] (hejlpe) arrayen.

Forklaring av "sorteringen"

Bytt den minste inn i *k

Hvis *j bytter, la den gå bakover

Hvis *i bytter, la den gå fremover

Forklaring av opprettelse av b[] array

i går fra "midten" og nedover så lenge den er større enn "l".

Den kopierer a[i-1] til b[i-1] (Mao, en ren kopi av arrayen under "m")

j går fra "midten" og oppover så lenge den er mindre enn "r".

Den kopierer den siste inn i j-1, neste gang den nest siste i j-1 osv.

Mergesort - kall 1

Mergesort blir kalt med 0 som venstre, 11 som høyre. Siden 11 > 0 så kjøres resten av koden.

Middle = (11+0)/2 = 5

	0	1	2	3	4	5	6	7	8	9	10	11
a)	E	A	S	Υ	Q	U*	E	S	T	1	0	N

Mergesort - kall 1-1

Mergesort blir kalt med 0 (I) som venstre, 5 (m) som høyre. Siden 5 > 0 så kjører koden.

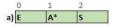
Middle = (5+0)/2 = 2

a) E	Α	S*	Y	Q	U	
0	1	2	3	4	5	

Mergesort - kall 1-1-1

Mergesort blir kalt med 0 (I) som venstre, 2 (m) som høyre. Siden 2 > 0 så kjører koden.

Middle = (2+0)/2 = 1



Mergesort - kall 1-1-1-1

Mergesort blir kalt med 0 (I) som venstre, 1 (m) som høyre. Siden 1 > 0 så kjører koden.

Middle = (1+0)/2 = 0

Mergesort - kall 1-1-1-1-1

Mergesort blir kalt med 0 (I) som venstre, 0 (m) som høyre. Siden 0 > 0 ikke stemmer, så kjører ikke koden.

Mergesort - kall 1-1-1-1-2

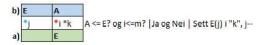
Mergesort blir kalt med 1 (m+1) som venstre, 1 (r) som høyre. Siden 1 > 1 ikke stemmer, så kjører ikke koden.

Sortering kjører for mergesort 1-1-1-1



b) E A i gkk en gang og er 0 | j gikk en gang og er 1

b)	E	A	
	*i *k	*j	E<=A og i<=m? Nei og Ja Sett A(j) i "k", j
a)	A		THE PUBLISHED DURAS CONSIDERATION OF THE PROPERTY OF THE PROPERTY OF THE PROPERTY OF THE PUBLISHED OF THE PU



a) A E Sub-array er ferdig sortert

Mergesort - kall 1-1-1-2

Mergesort blir kalt med 2 (m+1) som venstre, 2 (r) som høyre. Siden 2 > 2 ikke stemmer, så kjører ikke koden.

Sortering kjører for mergesort 1-1-1

	0	1	2	
a)	Α	E	S	
LI		-		

υJ	А	E	ls.	di .
b)	30.7	\$ 00	S	
b)	Α	E	S	i gikk en gang og er 0 j gikk fra 1-2
b)	Α	E	S	
	*i *k		*j	A<=S og i<=m? Ja og Ja Sett A(i) i "k", i++
a)	Α			A STATE OF THE STA
	<u> </u>	- 1		
b)	Α	E	S	
		*i *k	*j	E<=S? og i<=m? Ja og Ja Sett E(i) i "k", i++
a)		E		Alles SC STATE SC STATES AND
	(ii)		10.	
b)	Α	E	S	
	8	53	*i *k *j	S<=S? og i<=m? Ja og Nei Sett S(j) i "k", j
a)			S	The statement with the control of the statement of the st
		37:	70	-
al	Δ	E	S	Sub-array er ferdig sortert

Mergesort - kall 1-1-2

 $Mergesort\ blir\ kalt\ med\ 3\ (m+1)\ som\ venstre,\ 5\ (r)\ som\ høyre.\ Siden\ 5>3\ så\ kjører\ koden.$

Middle =
$$(5+3)/2=4$$

Mergesort - kall 1-1-2-1

Mergesort blir kalt med 3(I) som venstre, 4(m) som høyre. Siden 4 > 3 så kjører koden.

Middle =
$$(3+4)/2 = 3$$

Mergesort - kall 1-1-2-1-1

Mergesort blir kalt med 3(I) som venstre, 3(m) som høyre. Siden 3 > 3 ikke stemmer, så kjører ikke koden.

Mergesort - kall 1-1-2-1-2

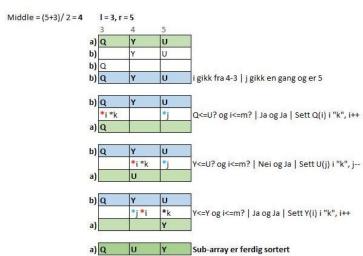
 $Mergesort \ blir \ kalt \ med \ 4(m+1) \ som \ venstre, \ 4(r) \ som \ høyre. \ Siden \ 4 < 4 \ ikke \ stemmer, \ så \ kjører \ ikke \ koden.$

Sortering kjører for mergesort 1-1-2-1

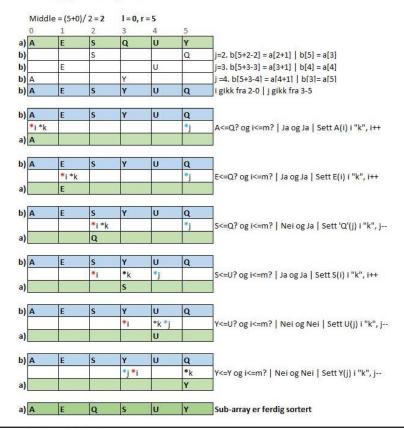
Mergesort - kall 1-1-2-2

 $Mergesort \ blir \ kalt \ med \ 5 (m+1) \ som \ venstre, 5 \ (r) \ som \ høyre. \ Siden \ 5 < 5 \ ikke \ stemmer, så \ kjører \ ikke \ koden.$

Sortering kjører for mergesort 1-1-2



Sortering kjører for mergesort 1-1



Mergesort - kall 1-2

Mergesort blir kalt med 6(m+1) som venstre, 11 (r) som høyre. Siden 11 > 6 så kjører koden.

Middle = (11+6)/2=8

a) E	S	T*		0	N	3
6	7	8	9	10	11	

Mergesort - kall 1-2-1

Mergesort blir kalt med 6(1) som venstre, 8(m) som høyre. Siden 8 > 6 så kjører koden.

Middle = (8+6)/2 = 7

Mergesort - kall 1-2-1-1

Mergesort blir kalt med 6(I) som venstre, 7(m) som høyre. Siden 7 > 6 så kjører koden.

Middle = (7+6)/2 = 6

Mergesort - kall 1-2-1-1-1

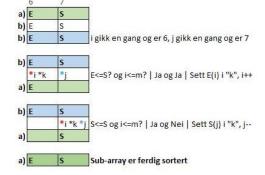
Mergesort blir kalt med 6(I) som venstre, 6(m) som høyre. Siden 6 > 6 ikke stemmer, så kjører ikke koden.

Mergesort - kall 1-2-1-1-2

Mergesort blir kalt med 7(m+1) som venstre, 7(r) som høyre. Siden 7 > 7 ikke stemmer, så kjører ikke koden.

Sortering kjører for mergesort 1-2-1-1

Middle =
$$(7+6)/2=6$$
 $l=6, r=7$



Mergesort - kall 1-2-1-2

Mergesort blir kalt med 8(m+1) som venstre, 8(r) som høyre. Siden 8 > 8 ikke stemmer, så kjører ikke koden.

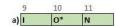
Sortering kjører for mergesort 1-2-1

	6	7	8	
a)	E	S	T	
b)	E		T	
b)		S		
b)	E	S	T	i gikk fra 7-6 j gikk en gang og er 8
b)	E	S	T	
	*i *k		*j	E T og i<=m? Ja og Ja Sett E(I) i "k", i++</td
a)	E			
b)	E	S	T	
		*i *k	*j	S<=T? og i<=m Ja og Ja Sett S(i) i "k", i++
a)		S		and seems that the second second second
	9	rii.		
b)	E	S	Т	
			*i *k *j	T<=T og i<=m Ja og Nei Sett T(j) i "k", j
a)			T	and the second s
	<u> </u>	100	77	
a)	E	S	T	Sub-array er ferdig sortert

Mergesort - kall 1-2-2

Mergesort blir kalt med 9(m+1) som venstre, 11(r) som høyre. Siden 11 > 9 så kjører koden.

Middle = (11+9)/ 2 = 10



Mergesort - kall 1-2-2-1-1

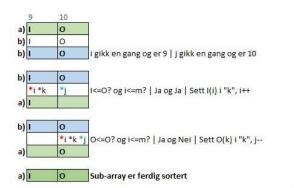
Mergesort blir kalt med 9(I) som venstre, 9(m) som høyre. Siden 9 > 9 ikke stemmer, så kjører ikke koden.

Mergesort - kall 1-2-2-1-2

 $Mergesort \ blir \ kalt \ med \ 10 (m+1) \ som \ venstre, \ 10 (r) \ som \ høyre. \ Siden \ 10 > 10 \ ikke \ stemmer, \ så \ kjører \ ikke \ koden.$

Sortering kjører for mergesort 1-2-2-1

Middle = (10+9)/2=9 l=9, r=10

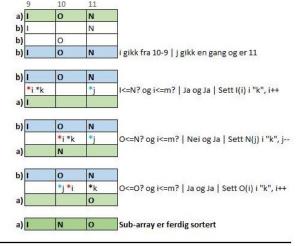


Mergesort - kall 1-2-2-2

Mergesort blir kalt med 11(m+1) som venstre, 11(r) som høyre. Siden 11 > 11 ikke stemmer, så kjører ikke koden.

Sortering kjører for mergesort 1-2-2

Middle = (11+9)/2=10 l=9, r=11



Sortering kjører for mergesort 1-2

C .	7		0	10	4.4	
6	1	0	9	10	1.1	

a)	E	S	T	I	N	0	
0)		3	Т			1	j = 8. b[11+8-8] = a[8+1]
0)		S	le e		N		j = 9. b[11+8-9] = a[9+1]
0)	E	8		0	į.	II.	j=10. b[11+8-10] = a[10+1]
)	E	S	T	0	N	ı	
1	E	s	Т	0	N	ı	
,	*i *k				-	*;	E<=i? og i<=m? Ja og Ja Sett E(i) i "k", i++
1)	E				2000		E car og ramin pad og sa pacitic (i) i k , i i i
			1050	100			
)	E	S	Т	0	N	ı	
		*i *k				*j	S<=I og i<=m? Nei og Ja Sett I(j) i "k", j
1)		I					
1	E	S	T	0	N	- li	
		*i	*k		*;		S<=N? og i<=m? Nei og Ja Sett N(j) i "k", j
1)			N				
							_
)	E	S	T	0	N	ı	
		*i		*k *j			S<=O og i<=m? Nei og Nei Sett O(j) i "k", j
1)				0			
1	E	s	T	0	N	1	
•		*;	*1		*k		S<=T? og i<=m? Ja og Ja Sett S(i) i "k", i++
1)					S		
			-		1	-1.	
))	E	S	T	0	N	1	T . T
			*j *i			*k	T<=T? og i<=m? Ja og Ja Sett T(i) i "k", i++
1)						Т	
	V	4	10	2	S		_

Sortering kjører for mergesort 1

0	1	2	3	4	5	6	7	8	9	10	11	
A	E	Q	5	U	Υ	E	1	N	0	S	T	
	9		8	20	Υ	-	-8	-03	- 92		E	j = 5. b[11+5-5] = a[5+1]
	-		12	Ü			4			1		j = 6. b[11+5-6] = a[6+1]
	- 9	4_	S	- 88	- 8	-	-0	-	N	- 8	48	j = 7. b[11+5-7] = a[7+1]
	-	Q	-	-			-	0	-	-	-	j = 8. b[11+5-8] = a[8+1]
	Е	8	- 8	- 29	- 8	-	S	(5)	98	- 5	45	j = 9. b[11+5-9] = a[9+1]
A	-		6			T		0			-	j=10. b[11+5-10] = a[10+1]
A	E	Q	S	U	Υ	T	S	0	N	- 4	E	
A	E	Q	S	U	Υ	Т	s	0	N	1	E	
*i *k		ч	3	U	0.0	100	3	U	10	9.0	*	Ac=E2 og ic=m2 lo og lo Sott A(i) i "k" i.i.
V.												A<=E? og i<=m? Ja og Ja Sett A(i) i "k", i++
A										l l		
A	E	Q	S	U	Υ	T	s	0	N	1	E	
V.	*i *k	4	-	9	3	-	-	_	-		*;	E<=E? og i<=m? Ja og Ja Sett E(i) i "k", i++
	E										J	og is-ini Ja og Ja Sett E(i) i K , iff
	E											_
A	E	Q	S	U	Y	Т	S	0	N	1	E	
н	-	*i *k	3	U	100		3	U	IN		*;	OK-E2 og ik-m2 Noi og la Sott E/i\ i "b" i
		E	15	24					-			Q<=E? og i<=m? Nei og Ja Sett E(j) i "k", j
		E										
A	E	Q	S	U	Υ	T	S	0	N	4	E	-
	-	*	*k	-	100				-	*	-	Q<=1? og i<=m? Nei og Ja Sett I(j) i "k", j
		9	1	3	8					-	18	Quality og running for the rogar joeter (j) i k , j
					ja)	3 6		16	3	- 10	0	
A	E	Q	S	U	Y	Т	S	0	N	1	E	
	-	*;		*k					*;			Q<=N? og i<=m? Nei og Ja Sett N(j) i "k", j-
				N	3					8		Zam oga mi mer og su sem ng/ ma//
				1 7000	1/4	-1	7	11		30		
A	E	Q	S	U	Y	Т	S	0	N	1	E	
/		*;		-	*k			*1	-			Q<=O? og i<=m? Nei og Nei Sett O(j) i "k",
					0							
		21			1,50			1/2		#		
A	E	Q	S	U	Y	Т	S	0	N	4	E	
		*i				*k	*j					Q<=S? og i<=m? Ja og Ja Sett Q(i) i "k", i++
						Q						
												_
A	E	Q	S	U	Υ	T	S	0	N	1	E	
			*				*k *j					S<=S? og i<=m? Ja og Ja Sett S(i) i "k", i++
							S					
	_				10		100			10		
A	E	Q	S	U	Y	T	S	0	N	1	E	
				*i			*j	*k				U<=S? og i<=m? Nei og Ja Sett S(j) i "k", j
								S				
					12	2		7		1		
A	E	Q	S	U	Y	T	S	0	N	ı	E	
				*;		*i			*k			U<=T og i<=m? Nei og Ja Sett T(j) i "k", j
	_			- 1					т			

) A	E	Q	S	U	Υ	T	S	0	N	ı	E	
					*j *i						*k	Y<=Y? og i<=m? Ja og Ja Sett Y(i) i "k", i++
)		ij.									Y	
			-	4		0.		-			12.00	-
) A	E	E	1	N	0	0	e	c	т	11	v	Array er ferdig sortert