Algoritmiske metoder

Elementære sorteringer

Utskrivbare versjoner

Mats Ove Mandt Skjærstein
09.07.2018

Selection sort

Finner det minste elementet i arrayen, dvs. den "skanner" arrayen og tar vare på den minste.
Når den minste er funnet, bytter den plass med den førte (på første loop).
Slik går den fremover og slik vil arrayen alltid ha de minste elementene bak seg mens sorteringen pågår.
Selection sort er en **in-place sorting algoritme**, det vil si at den ikke bruker "hjelpe" arrayer for å sortere.
Selection sort er **ikke stable,** det vil si at keys av samme verdi (eks E og E) vil bytte plass.

Forklaringer

* "Meg selv", den jeg skal bytte med den minste.
* Den minste jeg har funnet.
* Elementer jeg skal sammenligne med (alle elementer i høyere indeks enn "meg selv")
Disse feltene betyr at en swap oppsto

0	1	2	3	4	5	6	7	8	9	10	11	12	
	E	**A**	**S**	**Y**	**Q**	**U**	**E**	**S**	**T**	**I**	**O**	**N**	
	*A	*E	*S	*Y	*Q	*U	*E	*S	*T	*I	*O	*N	A var den minste. E < A? Nei, vi swapper
		*E	*S	*Y	*Q	*U	*E	*S	*T	*I	*O	*N	E var den minste. E < E? Nei, vi swapper
			*E	*Y	*Q	*U	*S	*S	*T	*I	*O	*N	E var den minste. S < E? Nei, vi swapper
				*I	*Q	*U	*S	*S	*T	*Y	*O	*N	I var den minste. Y < I? Nei vi swapper
					*N	*U	*S	*S	*T	*Y	*O	*Q	N var den minste. Q < N? Nei, vi swapper
						*O	*S	*S	*T	*Y	*U	*Q	O var den minste. U < O? Nei, vi swapper
							*Q	*S	*T	*Y	*U	*S	Q var den minste. S < Q? Nei, vi swapper
								*S	*T	*Y	*U	*S	S var den minste. S < S? Nei, vi swapper
									*S	*Y	*U	*T	S var den minste. T < S? Nei, vi swapper
										*T	*U	*Y	T var den minste. Y < T? Nei, vi swapper
											*U	*Y	U er den minste U < U? Nei vi swapper (med seg selv)
	A	**E**	**E**	**I**	**N**	**O**	**Q**	**S**	**S**	**T**	**U**	**Y**	**Ferdig sortert**

Comparrisons:		**Swaps:**	
Sum:	**66**	Sum:	**11**

Insertion sort

Insertion sort begynner på indeks 2 og sammenligner seg selv med den bak.
Hvis den bak er av høyere verdi, så bytter de plass.
Den bytter plass slik bakover helt den har kommet på sin rette plass.
Insertion sort bruker en **sentinal key** i indeks 0.
Insertion sort er en **in-place sorting algoritme,** det vil si at den ikke bruker "hjelpe" arrayer for å sortere.
Insertion sort er **stable,** det vil si at keys av samme verdi (eks E og E) vil ikke bytte plass

Forklaringer

* "Meg selv", den jeg skal "inserte" på riktig plass.
* Den jeg skal compare med i første omgang, hvis jeg swapper om denne så skal jeg compare "meg selv" med "*"
* Elementer som det skal compares med
Disse feltene betyr at en swap oppsto

0	1	2	3	4	5	6	7	8	9	10	11	12	
-1	**E**	**A**	**S**	**Y**	**Q**	**U**	**E**	**S**	**T**	**I**	**O**	**N**	
*-1	*A	*E											E > A? Ja, swapp | -1 > A? Nei
		*E	*S										E > S? Nei
			*S	*Y									S > Y? Nei
		*E	*Q	*S	*Y								Y > Q? Ja, swapp | S > Q? Ja, swapp | E > Q? Nei
				*S	*U	*Y							Y > U? Ja, swapp | S > U? Nei
		*E	*E	*Q	*S	*U	*Y						Y > E? Ja, swapp | U > E? Ja, swapp | S > E? Ja, swapp | Q > E? Ja, swapp | E > E? Nei
					*S	*S	*U	*Y					Y > S? Ja, swapp | U > S? Ja, swapp | S > S? Nei
						*S	*T	*U	*Y				Y > T? Ja, swapp | U > T? Ja, swapp | S > T? Nei
			*E	*I	*Q	*S	*S	*T	*U	*Y			Y > I? Ja, swapp | U > I? Ja, swapp | T > I? Ja, swapp | S > I? Ja, swapp | S > I? Ja, swapp | Q > I? Ja, swapp | E > I? Nei
				*I	*O	*Q	*S	*S	*T	*U	*Y		Y > O? Ja, swapp | U > O? Ja, swapp | T > 0? Ja, swapp | S > O? Ja, swapp | S > O? Ja, swapp | Q > O? Ja, swapp | I > O? Nei
				*I	*N	*O	*Q	*S	*S	*T	*U	*Y	Y > N? Ja, swapp | U > N? Ja, swapp | T > N? Ja, swapp | S > N? Ja, swapp | S > N? Ja, swapp | Q > N? Ja, swapp | O > N? Ja, swapp | I > N? Nei
-1	**A**	**E**	**E**	**I**	**N**	**O**	**Q**	**S**	**S**	**T**	**U**	**Y**	**Ferdig sortert**

Comparisons:		**Swaps:**	
Sum:	**42**	Sum:	**31**

Bubble sort

Bubble sort ser på et element og swapper det oppover til til sin rette plass.
Hvis den bak har høyere verdi, så bytter de plass. Dette gjør den mens den beveger seg oppover i arrayen.
Slik vil dette elementet "bobble" seg opp til sin daværende riktige plass.
Bubble sort er en **in-place sorting algoritme,** det vil si at den ikke bruker "hjelpe" arrayer for å sortere.
Bubble sort er **stable,** det vil si at keys av samme verdi (eks E og E) vil ikke bytte plass

Forklaringer

* "Meg selv", den som skal "bobble" opp til sin rette plass.
* Den jeg skal compare med i første omgang, hvis jeg swapper om denne så skal jeg compare den nye "meg selv" med "*"
* Elementer jeg skal compare med (alle elementer i høyere indeks enn "meg selv")
Disse feltene betyr at en swap oppsto
Disse feltene betyr at denne delen ikke lenger sjekkes (er sortert)

0	1	2	3	4	5	6	7	8	9	10	11	12	
	E	A	S	Y	Q	U	E	S	T	I	O	N	**Outer loop 1 (Underliggende "Inner loop" gjenntar seg for hver "Outer loop")**
	*A	*E											E > A? Ja, swap
		*E	*S										E > S? Nei
			*S	*Y									S > Y? Nei
				*Q	*Y								Y > Q? Ja, swap
					*U	*Y							Y > U? Ja, swap
						*E	*Y						Y > E? Ja, swap
							*S	*Y					Y > S? Ja, swap
								*T	*Y				Y > T? Ja, swap
									*I	*Y			Y > I? Ja, swap
										*O	*Y		Y > O? Ja, swap
											*N	*Y	Y > N? Ja, swap
	A	E	S	Q	U	E	S	T	I	O	N	Y	**Outer loop 2 (komprimert)**
	A	E	Q	S	E	S	T	I	O	N	U	Y	**Outer loop 3 (komprimert)**
	A	E	Q	E	S	S	I	O	N	T	U	Y	**Outer loop 4 (komprimert)**
	A	E	E	Q	S	I	O	N	S	T	U	Y	**Outer loop 5 (komprimert)**
	A	E	E	Q	I	O	N	S	S	T	U	Y	**Outer loop 6 (komprimert)**
	A	E	E	I	O	N	Q	S	S	T	U	Y	**Outer loop 7 (komprimert)**
	A	E	E	I	N	O	Q	S	S	T	U	Y	**Outer loop 8 (komprimert)**
	A	E	E	I	N	O	Q	S	S	T	U	Y	**Outer loop 9 (komprimert)**
	A	E	E	I	N	O	Q	S	S	T	U	Y	**Outer loop 10 (komprimert)**
	A	E	E	I	N	O	Q	S	S	T	U	Y	**Outer loop 11 (komprimert)**
	A	E	E	I	N	O	Q	S	S	T	U	Y	**Outer loop 12 (komprimert)**
	A	E	E	I	N	O	Q	S	S	T	U	Y	**Ferdig sortert**

Comparisons:		**Swaps:**	
Sum:	**66**	Sum:	**31**

Shellsort

Fungerer som insertion sort, men i stedet for å sammenligne elementene ved siden av, blir det valgt et sett gap imellom.
Hvis et element "swappes" så sjekkes elementet bak, samme som insertion sort.
Denne kjører helt til gap blir 1, og vanlig insertion sort kjører på arrayen.
Shellsort er en **in-place sorting algoritme**, det vil si at den ikke bruker "hjelpe" arrayer for å sortere.
Shellsort er **stable**, det vil si at keys av samme verdi (eks E og E) vil ikke bytte plass.

Forklaringer

* "Meg selv", eller den jeg bruker for å compare med de andre elementene
* Den jeg skal compare med i første omgang, hvis jeg swapper om denne så skal jeg compare denne med "*"
* Elementer som det skal compares med
Disse feltene betyr at en swap oppsto

0	1	2	3	4	5	6	7	8	9	10	11	12	
-1	**E**	**A**	**S**	**Y**	**Q**	**U**	**E**	**S**	**T**	**I**	**O**	**N**	**Gap=7 (Gap 7 er valgt i dette eksempelet)**
	*E							*S					E > S? Nei
		*A							*T				A > T? Nei
			*I							*S			S > I? Ja, swap | "out of bonds comparison"
				*O							*Y		Y > O? Ja, swap | "out of bonds comparison"
					*N							*Q	Q > N? Ja, swap | "out of bonds comparison"
-1	**E**	**A**	**I**	**O**	**N**	**U**	**E**	**S**	**T**	**S**	**Y**	**Q**	**Gap=3**
	*E			*O									E > O ? Nei
		*A			*N								A > N? Nei
			*I			*U							I > U? Nei
	*E			*E			*O						O > E? Ja, swap | E > E? Nei (stable)
					*N			*S					N > S? Nei
			*I			*T			*U				U > T? Ja, swap | I > T? Nei
							*O			*S			O > S? Nei
								*S			*Y		S > Y? Nei
				*E		*Q			*T			*U	U > Q? Ja, swap | T > Q? Ja, swap | E > Q? Nei
-1	**E**	**A**	**I**	**E**	**N**	**Q**	**O**	**S**	**T**	**S**	**Y**	**U**	**Gap=1 (Insertion sort)**
*-1	*A	*E											E > A? Ja, swap | -1 > A? Nei
		*E	*I										E > I? Nei
		*E	*E	*I									I > E? Ja, swap | E > E? Nei
				*I	*N								I > N? Nei
					*N	*Q							N > Q? Nei
					*N	*O	*Q						Q > O? Ja, swap | N > O? Nei
							*Q	*S					Q > S? Nei
								*S	*T				S > T? Nei
								*S	*S	*T			T > S? Ja, swap | S > S? Nei
										*T	*Y		T > Y? Nei
										*T	*U	*Y	Y > U? Ja, swap | T > U? Nei
-1	**A**	**E**	**E**	**I**	**N**	**O**	**Q**	**S**	**S**	**T**	**U**	**Y**	**Ferdig sortert**

Comparrisons:		**Swaps:**	
Når gap=7	8	Når gap=7	3
Når gap=3	13	Når gap=3	4
Når gap=1	16	Når gap=1	5
Sum:	**37**	Sum:	**12**