## Selection Sort Algorithm

**CS2002D Program Design** 

## Selection Sort - Visualization

Which among the two

takes more swaps?



- One among the simple comparison based sorting technique.
- In each iteration, the array is divided into two parts such that the first part of the array is sorted and the second part is unsorted.
- The algorithm begins with an empty first part and the entire array in the second part.
- The minimum element from the unsorted part is appended to the sorted first part in each iterations.
- At the end, the second part vanishes and the whole array constitute the first part - the sorted array.

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	
2	2	
4	10	
3	$\infty$	
2	4	min = 6 pos = 1
1	9	E d
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	
Ŋ	2	
4	10	
3	$\infty$	
2	4	min = 4 pos = 2
	9	E d
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12		
9	14		
2	2		
4	10		
3	$\infty$	min = 4 pos = 2	
2	4	u d	J
1	9		
index	value		

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	
5	2	
4	10	min = 4 pos = 2
3	$\infty$	
2	4	
1	9	
ndex	zalue .	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	
5	7	min = 2 pos = 5
4	10	n d
3	$\infty$	
2	4	
1	9	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12				
9	14	<	\	nin = 2	bos = 5
5	2			_ n	<u>a</u>
4	10				
33	$\infty$				
2	4				
-	9				
index	value				

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if \left( min > A[j] \right)$$

Update min=A[j]

and pos=j

7	12		min = 2 pos = 5
9	14		n d
5	2		
4	10		
33	$\infty$		
2	4		
<b>1</b>	9		
index	value		

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

index	-	2	3	4	2	9	7
value	9	4	$\infty$	10	2	14	12
	<						
	\						
	min = 2 pos = <b>5</b>						

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

12	
14	
9	
10	
$\infty$	
4	
2	min = 2 pos = 5
value	
	<b>2</b> 4 8 10 6 14

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

index	1	2	3	4	5	9	7
value	2	4	$\infty$	10	9	14	12
		<					
		1					
		min = 4					
		pos = 2					

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

index	1	2	3	4	5	9	7
value	2	4	$\infty$	10	9	14	12
		<					
		\					
		min = 4					
		pos = 2					

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	
2	9	
4	10	<b>m</b>
3	$\infty$	min = 8 pos = 3
2	4	
1	2	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

7	12	
9	14	
2	9	<b>M</b>
4	10	 min = 8 pos = 3
3	$\infty$	
2	4	
1	2	
ndex	alue	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if \left( min > A[j] \right)$$

Update min=A[j]

and pos=j

7	12	
9	14	
r <sub>2</sub>	9	min = 8 pos = 3
4	10	
33	$\infty$	
2	4	
	7	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

	1	
7	12	
9	14	
5	9	min = 6 pos = 5
4	10	
3	$\infty$	
2	4	
	7	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12			
9	14	1	min = 6	bos = 5
<sub>1</sub> C	9			
4	10			
33	$\infty$			
2	4			
-	2			
index	value			

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i

b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	min = 6	0 - 0
9	14	E S	3
r.	9		
4	10		
33	$\infty$		
2	4		
1	2		
index	value		

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

3 4 5 6 7	8 10 6 14 12		in = 6	pos = 5
2	4		min	bos
1	7			
index	value			

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

3 4 5 6 7	6         10         8         14         12		9:	bos = 5
2	4		E	
1	2			
index	value			

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

2 9	14 12			
5	$\infty$			0
4	10	<	/	min = 10
33	9			
2	4			
H	2			
index	value			

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	01
5	<b>&amp;</b>	min = 10 pos = 4
4	10	
33	9	
2	4	
-	2	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

	1	
7	12	
9	14	
72	œ	min = 8 pos = 5
4	10	
3	9	
2	4	
1	7	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	<b>m</b>
9	14	min = 8 pos = 5
<sub>1</sub> C	$\infty$	
4	10	
33	9	
2	4	
	2	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i

b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

7	12	min = 8 pos = 5
9	14	n d
72	$\infty$	
4	10	
3	9	
2	4	
1	2	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

5 6 7	8 14 12			
4	10	<	<b>1</b>	min = 8
2 3	4 6		(	
1	2			
index	value			

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

	1	
	12	
9	14	
<sub>1</sub>	10	
4	<b>∞</b>	min = 8 pos = 5
33	9	n d
2	4	
	2	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i

b. for j=i+1 to A.length

$$if\left(min{>}A[j]\right)$$

Update min=A[j]

and pos=j

7	12		
9	14	01	
5	10	min = 10	pos = 5
4	<b>\(\omega\)</b>		
3	9		
2	4		
T	2		
index	value		

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	10 14	min = 10 pos = 5
r <sub>C</sub>	10	n o
4	<b>∞</b>	
3	9	
2	4	
T	7	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if\left( min {>} A[j] \right)$$

Update min=A[j]

and pos=j

7	12	<	۸ <u>ــــ</u>	min = 10	os = 5
9	14			_ n	ď
72	10				
4	<b>∞</b>				
33	9				
2	4				
1	2				
index	value				

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	
9	14	
5	10	min = 10 pos = 5
4	<b>&amp;</b>	H d
3	9	
2	4	
1	2	
ldex	alue	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	4
9	14	min = 14 pos = 6
rC	10	E a
4	<b>∞</b>	
3	9	
2	4	
T	2	
index	value	

#### SELECTION\_SORT(A)

- 1. for i=1 to A.length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	min = 12 pos = 7
9	14	u d
5	10	
4	8	
3	9	
2	4	
1	2	
index	value	

#### SELECTION\_SORT(A)

1. for i=1 to A length - 1

index

value

- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

Update min=A[j]

and pos=j

7	12	2
9	14	min = 12 pos = 7
<sub>1</sub> C	10	n d
4	<b>∞</b>	
33	9	
2	4	
1	2	

#### SELECTION\_SORT(A)

- 1. for i=1 to A length 1
- a. Initialize min=A[i], pos=i
- b. for j=i+1 to A.length

$$if \left( min > A[j] \right)$$

Update min=A[j]

and pos=j

7	14			2	
9	10 12	<	1	nin = 1	pos = 7
5	10			, n	<u>d</u>
4	<b>∞</b>				
3	9				
2	4				
	2				
index	value				

- 1. Loop invariant: (ascending order)
- a. At the start of ith iteration of the for loop, the subarray
- A[1... i-1] is sorted, which contains the least 'i-1' elements.
- # comparisons (step 1.b) to sort the array in asce. order,
- (n(n-1)/2)If the input is in asce, order
- b. If the input is in desc. order

(n(n-1)/2)

Minimum and maximum number of comparisons and swaps?  $\sim$ 

#### Questions

contains the first i-1 elements of the original array A, in sorted 1. At the start of i<sup>th</sup> iteration of the for loop, the subarray A[1... i-1] order

a. True for insertion sort

b. False for selection sort

2. To sort 'n' elements,

copy operations (steps 6 and 8)? a. Insertion sort requires\_

swaps (step 1.c)? Selection sort requires \_\_\_ **þ** 

comparisons (step 1.b)?  $nC_2$ Selection sort requires \_\_\_\_

Is selection sort Stable? What about the other sorts? 3

# **Proof of correctness - Selection sort**

Loop invariant: (ascending order)

At the start of i<sup>th</sup> iteration of the for loop, the subarray A[1... i-1] is sorted, which contains the least 'i-1' elements of Array A.

- Initialization: Loop invariant trivially holds before the first loop iteration. A[1] contains the minimum element, which is sorted by itself.
- **Maintenance:** At the end of i<sup>th</sup> iteration, the minimum of A[i..A.length] is selected and placed at ith position
- A[1..i-1] consists of the least i-1 elements in A, in sorted order.
- **Termination**: When the loop terminates, i = A.length and the subarray A[1... A,length-1] is sorted and A[A,length] has the max element,

# Selection Sort and Insertion sort

#### SELECTION\_SORT(A)

## INSERTION\_SORT(A)

2. 
$$key = A[j];$$

4. 
$$i = j-1$$

5. while 
$$i > 0$$
 and  $A[i] > key$ 

$$A[i+1]=A[i]$$

$$i=i-1$$

8. 
$$A[i+1] = key$$

Thank You III