Algorithms & Data Structures I
CSC 225

Ali Mashreghi

Fall 2018

Department of Computer Science, University of Victoria

Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```

10	2	8	5	7

Sorting by selection

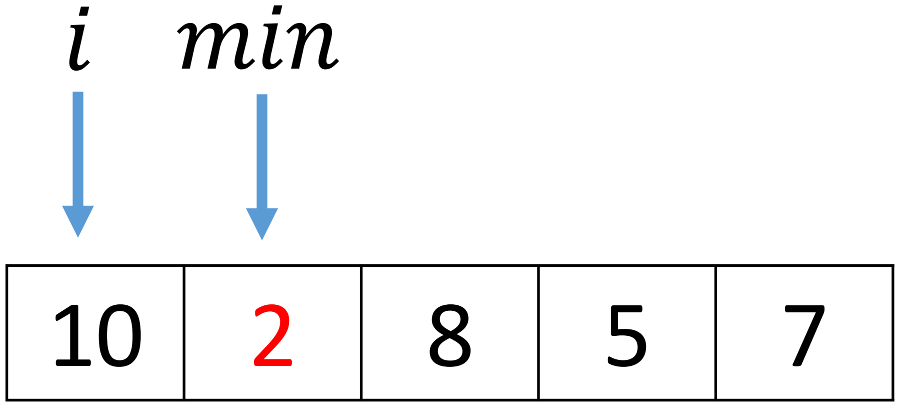
```
SELECTION-SORT(A)
    for i = 1 to A.length − 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```

i

10	2	8	5	7

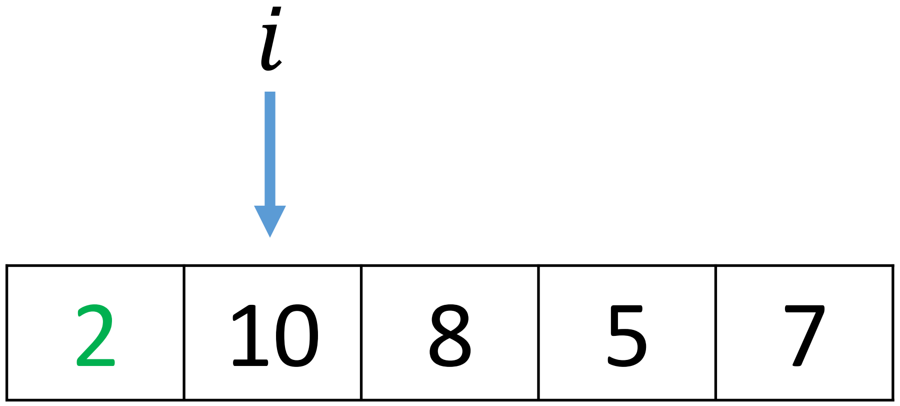
Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```



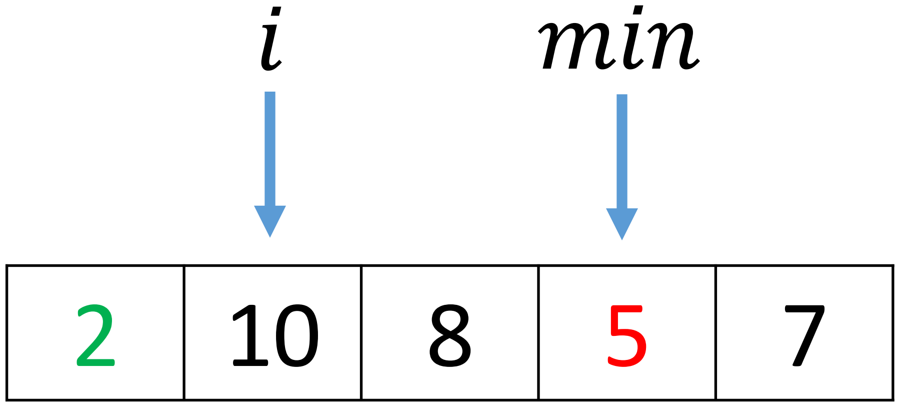
Sorting by selection

```
SELECTION-SORT(A)
for i = 1 to A.length – 1
    min = i
    for j = i + 1 to A.length
        if A[j] < A[min]
            min = j
    swap A[min] and A[i]
```



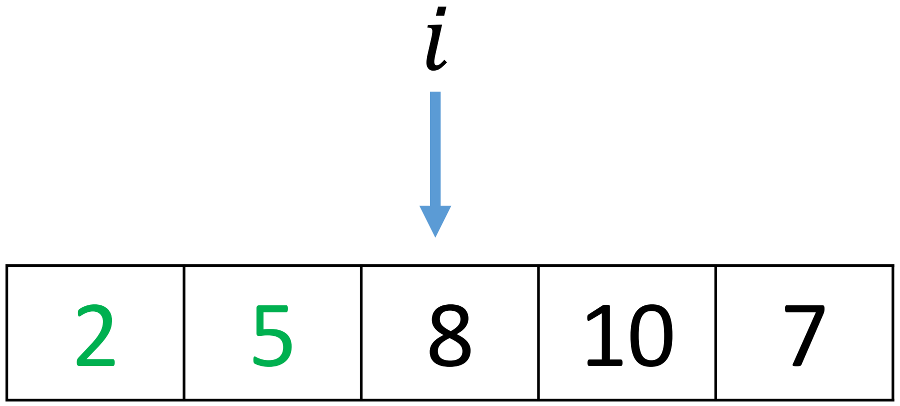
Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```



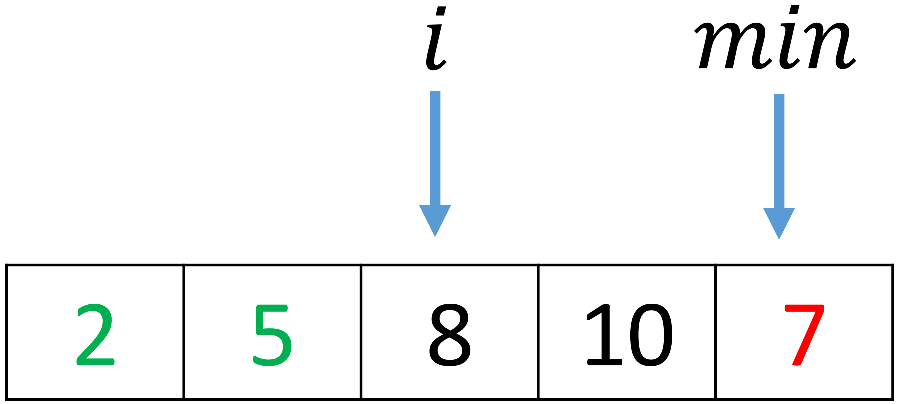
Sorting by selection

```
SELECTION-SORT(A)
for i = 1 to A.length – 1
    min = i
    for j = i + 1 to A.length
        if A[j] < A[min]
            min = j
    swap A[min] and A[i]
```



Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```



Sorting by selection

SELECTION-SORT(A)

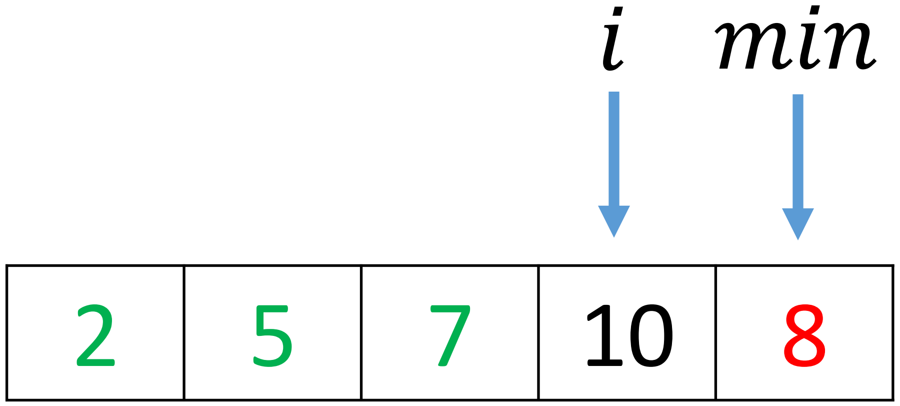
for $i = 1$ **to** $A.length - 1$
    $min = i$
    for $j = i + 1$ **to** $A.length$
        if $A[j] < A[min]$
            $min = j$
    swap $A[min]$ and $A[i]$

i

2	5	7	10	8

Sorting by selection

```
SELECTION-SORT(A)
for i = 1 to A.length – 1
    min = i
    for j = i + 1 to A.length
        if A[j] < A[min]
            min = j
    swap A[min] and A[i]
```



Sorting by selection

```
SELECTION-SORT(A)
for i = 1 to A.length − 1
    min = i
    for j = i + 1 to A.length
        if A[j] < A[min]
            min = j
    swap A[min] and A[i]
```

The loop terminates

				i
2	5	7	8	10

Proof of correctness

Loop invariant:

At the start of each iteration of the **outer** for loop (on i), $A[1..i-1]$ contains the smallest $i-1$ elements of A in sorted order.

Exercise: Prove initialization and maintenance, and conclude that when the for loop on i terminates, the array is sorted.

Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```

Cost	Number of times
c_1	n

Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```

Cost	Number of times
c_1	n
c_2	$n-1$

Sorting by selection

```
SELECTION-SORT(A)
for i = 1 to A.length – 1
    min = i
    for j = i + 1 to A.length
        if A[j] < A[min]
            min = j
    swap A[min] and A[i]
```

Cost	Number of times
c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$

Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length – 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```

Cost	Number of times
c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$

Sorting by selection

SELECTION-SORT(A)

```
for i = 1 to A.length – 1
    min = i
    for j = i + 1 to A.length
        if A[j] < A[min]
            min = j
    swap A[min] and A[i]
```

Cost	Number of times
c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$
c_5	$\sum_{i=1}^{n-1} t_i$

Sorting by selection

```
SELECTION-SORT(A)
    for i = 1 to A.length − 1
        min = i
        for j = i + 1 to A.length
            if A[j] < A[min]
                min = j
        swap A[min] and A[i]
```

Cost	Number of times
c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$
c_5	$\sum_{i=1}^{n-1} t_i$
c_6	$n-1$

Analysis

- So, the running time is **sum of the following**:

c_1	n

$\longrightarrow c_1 n$

Analysis

- So, the running time is **sum of the following**:

c_1	n
c_2	$n-1$

$c_1 n$

$\longrightarrow c_2 n + \cdots$

'...' means some lower order terms that we ignore

Analysis

- So, the running time is **sum of the following**:

c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$

$c_1 n$

$c_2 n + \cdots$

$\longrightarrow c'_3 n^2 + \cdots$

'…' means some lower order terms that we ignore

Analysis

- So, the running time is **sum of the following**:

c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$

$c_1 n$

$c_2 n + \cdots$

$c'_3 n^2 + \cdots$

$c'_4 n^2 + \cdots$

'…' means some lower order terms that we ignore

Analysis

- So, the running time is **sum of the following**:

c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$
c_5	$\sum_{i=1}^{n-1} t_i$

$c_1 n$

$c_2 n + \cdots$

$c'_3 n^2 + \cdots$

$c'_4 n^2 + \cdots$

$\rightarrow$ between 0 and $(c'_5 n^2 + \cdots)$

'…' means some lower order terms that we ignore

Analysis

- So, the running time is **sum of the following**:

c_1	n	$c_1 n$
c_2	$n-1$	$c_2 n + \cdots$
c_3	$\sum_{i=1}^{n-1}(n-i)$	$c'_3 n^2 + \cdots$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$	$c'_4 n^2 + \cdots$
c_5	$\sum_{i=1}^{n-1} t_i$	between 0 and $(c'_5 n^2 + \cdots)$
c_6	$n-1$	$c_6 n + \cdots$

'…' means some lower order terms that we ignore

Analysis

- So, the running time is sum of the following:

c_1	n
c_2	$n-1$
c_3	$\sum_{i=1}^{n-1}(n-i)$
c_4	$\sum_{i=1}^{n-1}(n-i-1)$
c_5	$\sum_{i=1}^{n-1} t_i$
c_6	$n-1$

$\cancel{c_1 n}$

$\cancel{c_2 n + \cdots}$

$c'_3 n^2 \cancel{+ \cdots}$

$c'_4 n^2 \cancel{+ \cdots}$

between 0 and $(c'_5 n^2 \cancel{+ \cdots})$

$\cancel{c_6 n + \cdots}$

Analysis

- By summing the following I get:

$c'_3 n^2$

$c'_4 n^2$

between 0 and $c'_5 n^2$

- worst-case:

$(c_3' + c_4' + c_5')n^2 = O(n^2)$

- best-case:

$(c_3' + c_4')n^2 = \Omega(n^2)$

Analysis

- worst-case is $O(n^2)$
- best-case is $\Omega(n^2)$

Selection-sort's time complexity is $\boldsymbol{\Theta(n^2)}$

Other $O(n^2)$ sorting algorithms

- **Bubble-sort** and **shell-sort** are two other sorting algorithms that run in $O(n^2)$ time.
- If interested, you can look it up on wikipedia.

- In practice, insertion-sort **is the most efficient $\boldsymbol{O(n^2)}$** sorting algorithm.

- In fact, if $\boldsymbol{n \sim 20}$ it is preferred over merge-sort, quick-sort and other optimal algorithms.