# L7

**sep 17 2011**

abhi shelat

Median, Arbitrage, FFT

Mergesort

Closest pair

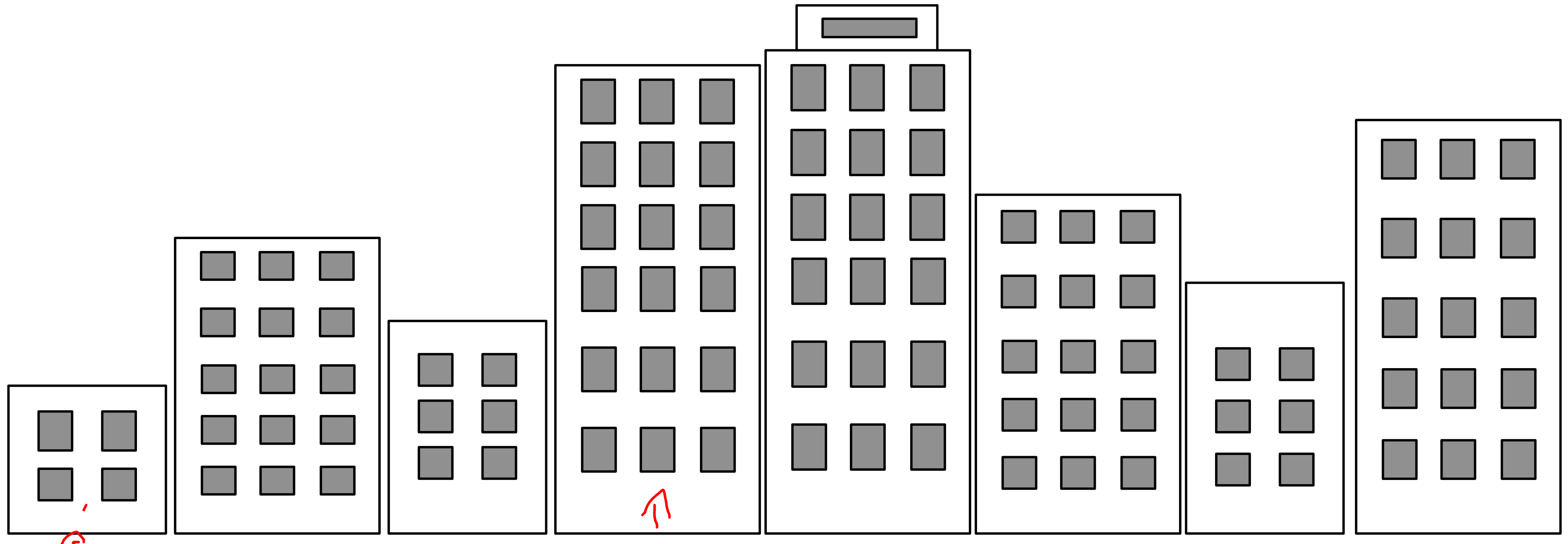
Karatsuba/Matrix

MEDIAN

arbitrage

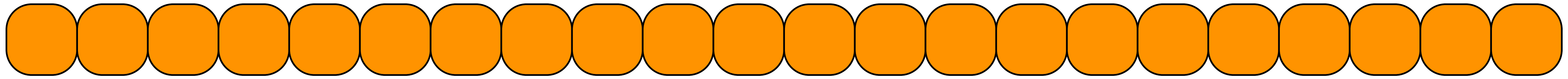
FFT

LeftNifty(A[1,...,n]) ← # of times that a building is taller than one of its left neighbors.



0 1 1 3 4 3 2 5 = 19

# MEDIAN

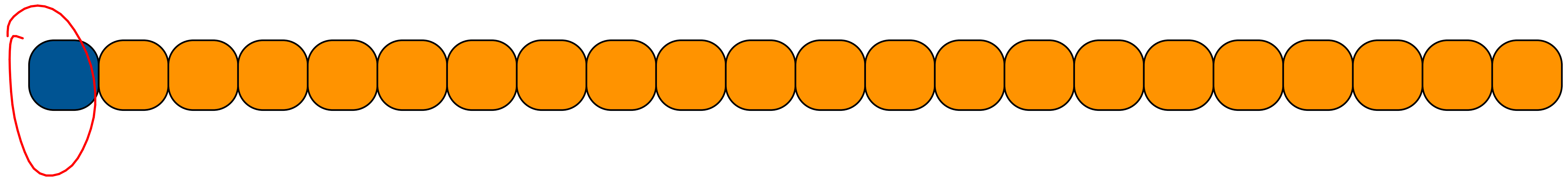


problem: given a list of $n$ elements, find the element of rank $i$.

**key insight:**

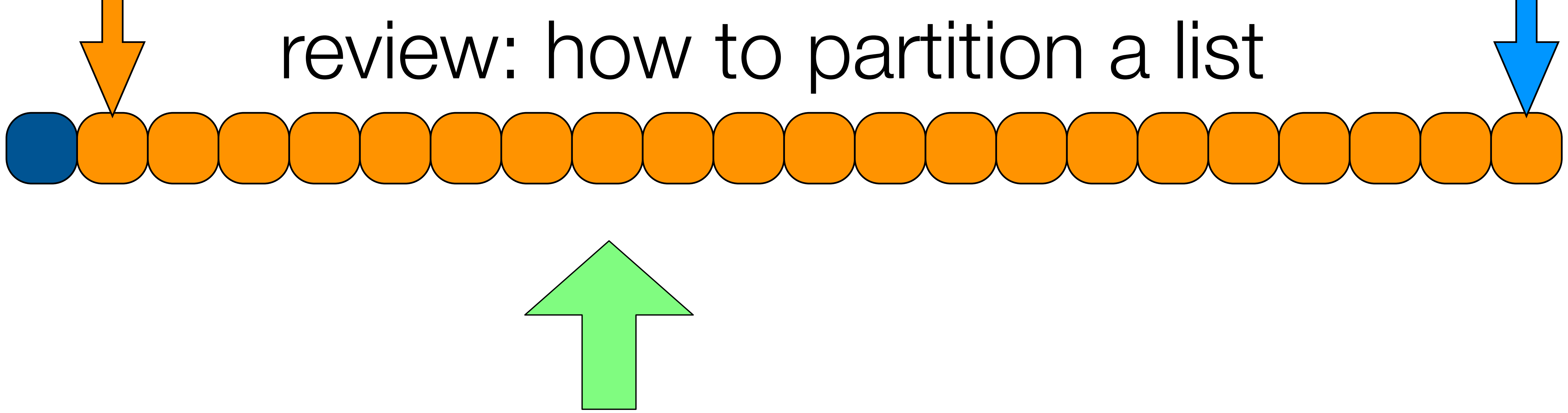
**we do not have to “fully” sort.**

**semi sort can suffice.**

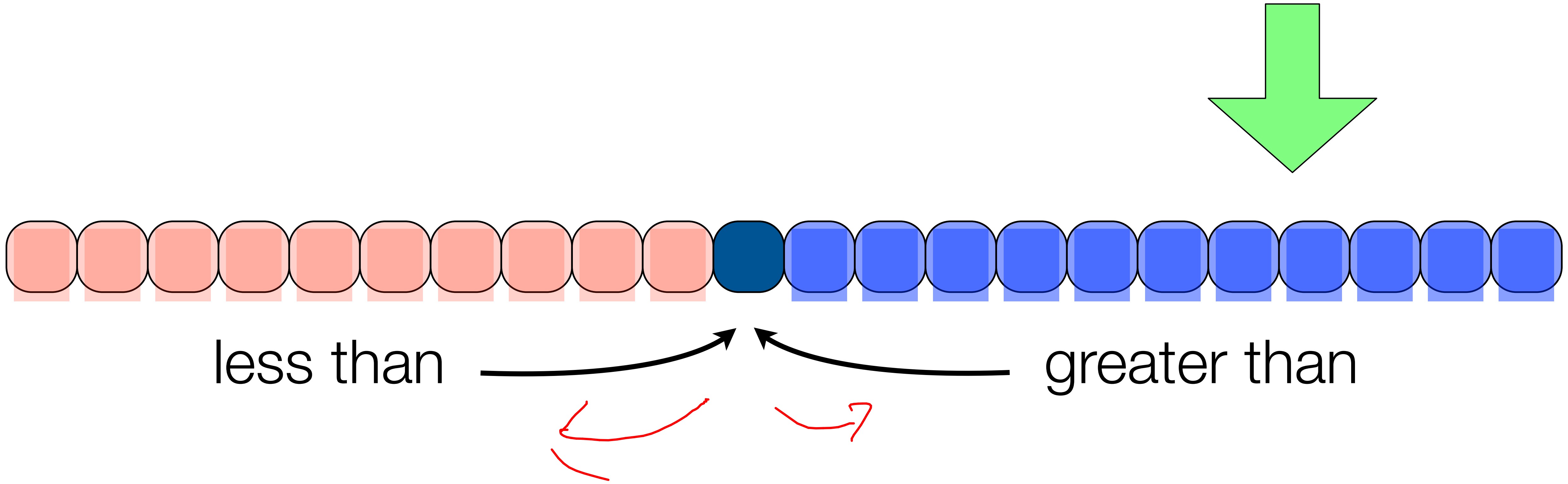


pick first element
partition list about this one
see where we stand

# review: how to partition a list

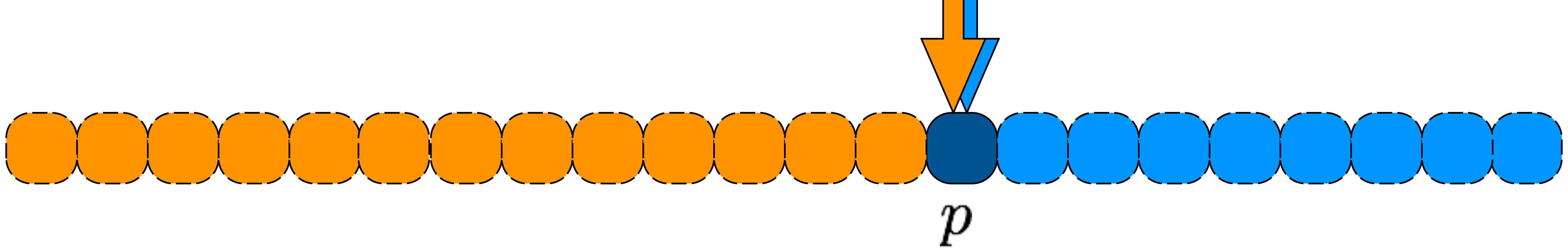


GOAL: start with THIS LIST and END with THAT LIST

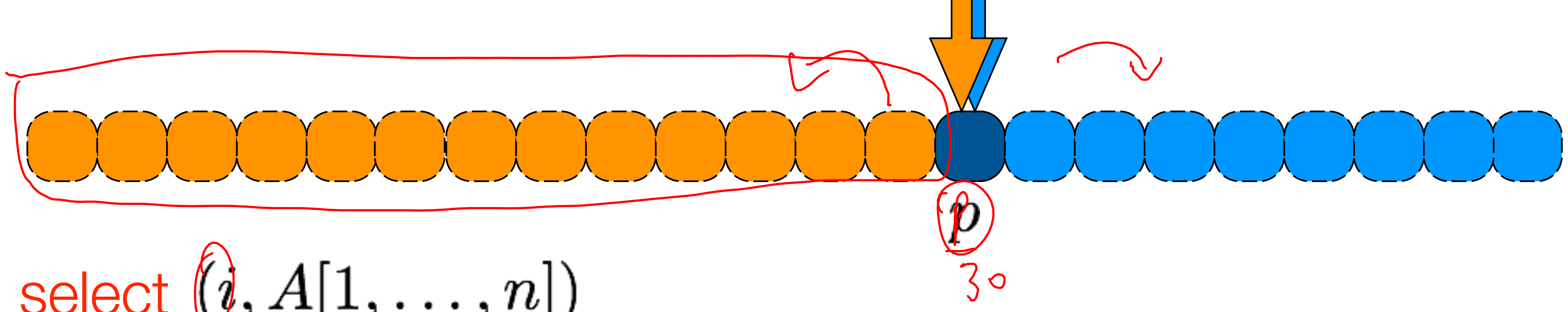


less than

greater than




select $(i, A[1, \ldots, n])$




select $(i, A[1, \ldots, n])$

handle base case.

partition list about first element

if pivot is position $i$, return pivot

else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$

else select $((i-p-1), A[p+1, \ldots, n])$

$\sim \frac{1}{2}$ of the input

$$T(n) = T\left(\frac{n}{2}\right) + \Theta(n) = \Theta(n)$$

select $(i, A[1, \ldots, n])$

Assume our partition always splits list into two eql parts

handle base case.

partition list about first element

if pivot is position $i$, return pivot

else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$

else select $((i-p-1), A[p+1, \ldots, n])$

select $(i, A[1, \ldots, n])$

Assume our partition always splits list into two eql parts

handle base case.

partition list about first element

if pivot is position $i$, return pivot

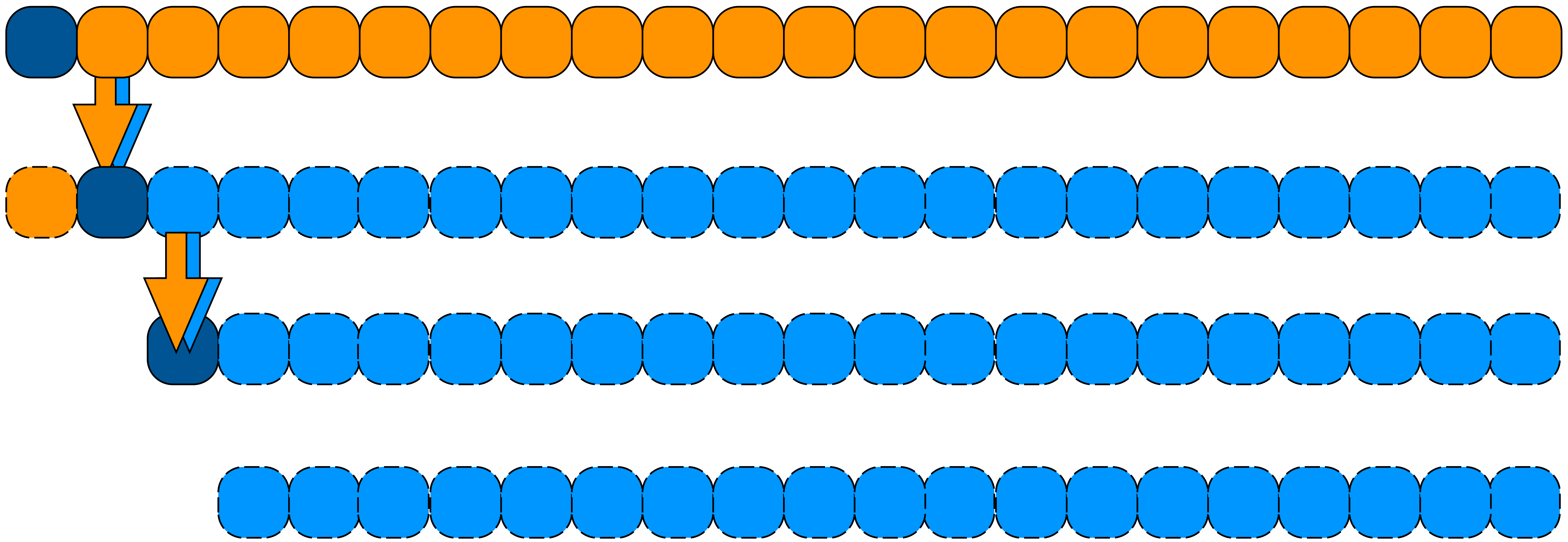
else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$

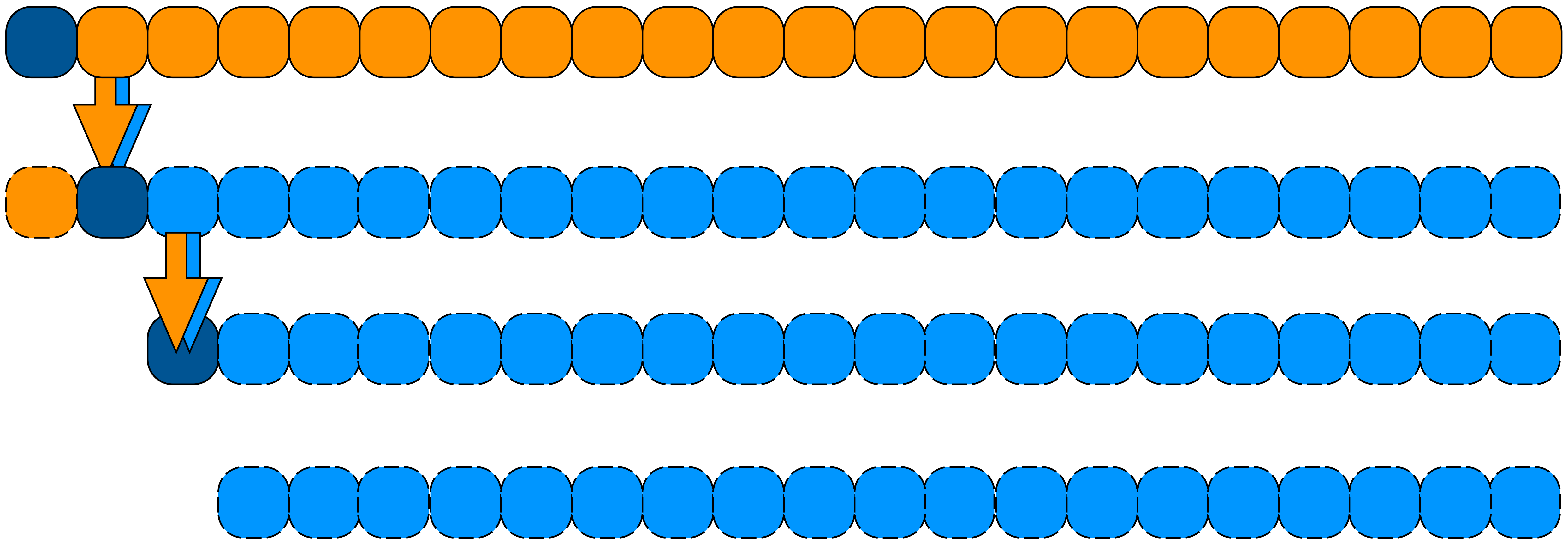
else select $((i-p-1), A[p+1, \ldots, n])$

$$T(n) = T(n/2) + O(n)$$

$$\Theta(n)$$

problem: what if we always pick bad partitions?





problem: what if we always pick bad partitions?

select $(i, A[1, \ldots, n])$

handle base case.

partition list about first element

if pivot is position $i$, return pivot

else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$

else select $((i-p-1), A[p+1, \ldots, n])$

select $(i, A[1, \ldots, n])$

handle base case.
partition list about first element ← When partition fails to split "properly"
if pivot is position $i$, return pivot
else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$
else select $((i-p-1), A[p+1, \ldots, n])$

then the running time follows

$$T(n) = T(n-1) + O(n)$$

→ $T(n-3)$

$$\Theta(n^2)$$

# Needed:

a good partition element

partition $(A[1, \ldots, n])$ → should find an element that has at least
30% of the array "on the left" smaller
30% of the array "on the right" larger

i.e., in the
30 – 70% percentiles.

# Needed:

a good partition element

partition $(A[1, \ldots, n])$

produce an element where
30% smaller, 30% larger

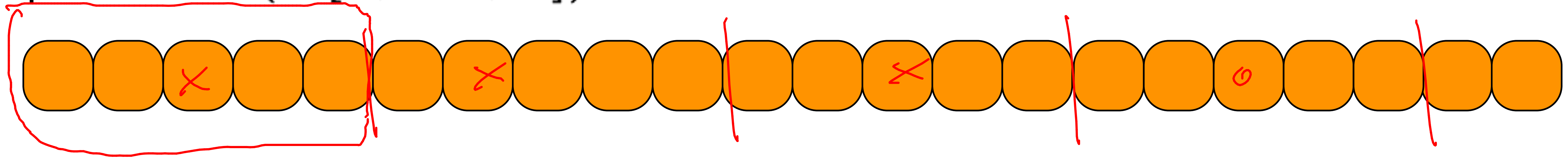
solution:
bootstrap


image: gucci

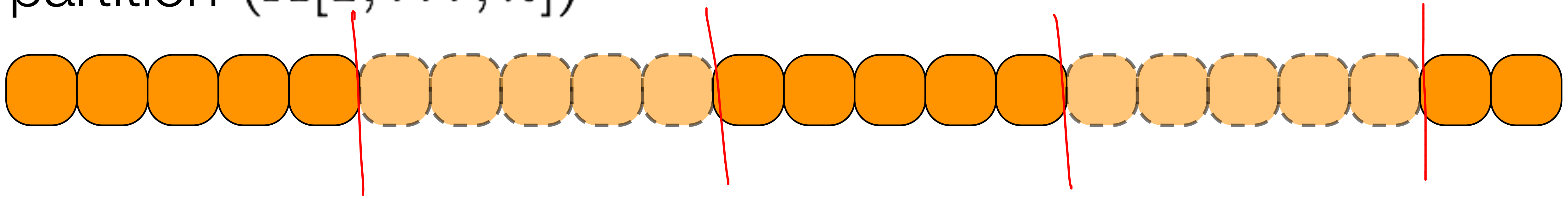

image: mark nason


image: d&g

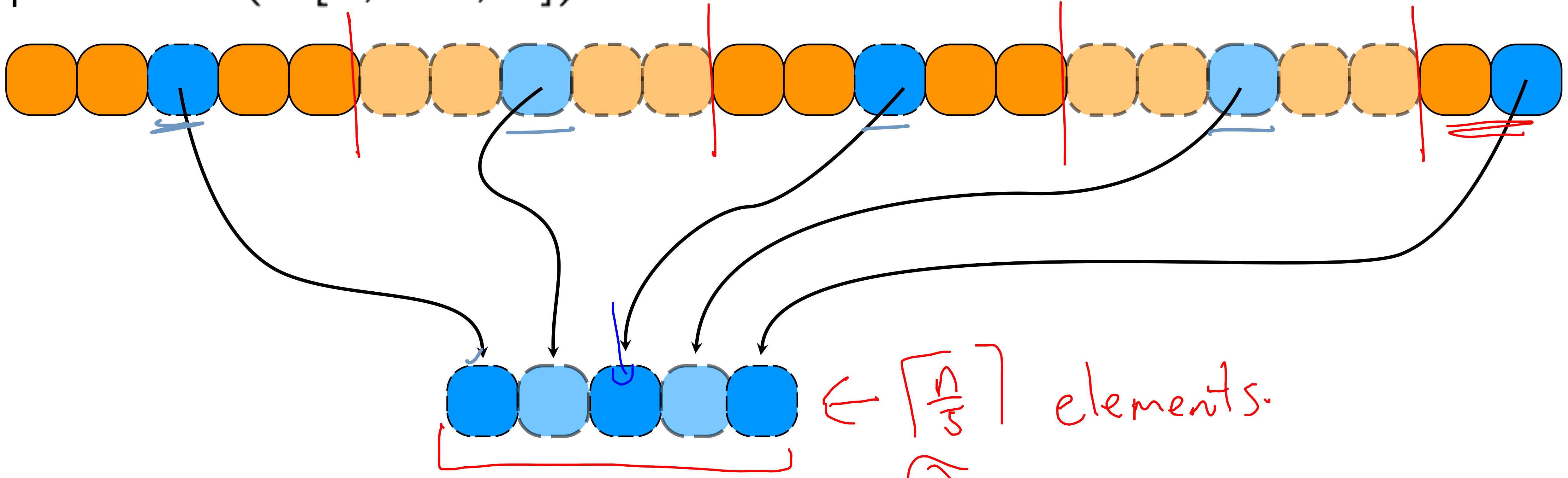
partition $(A[1, \ldots, n])$ — an element in the 30–70% percentile.



partition $(A[1, \ldots, n])$



partition $(A[1, \ldots, n])$

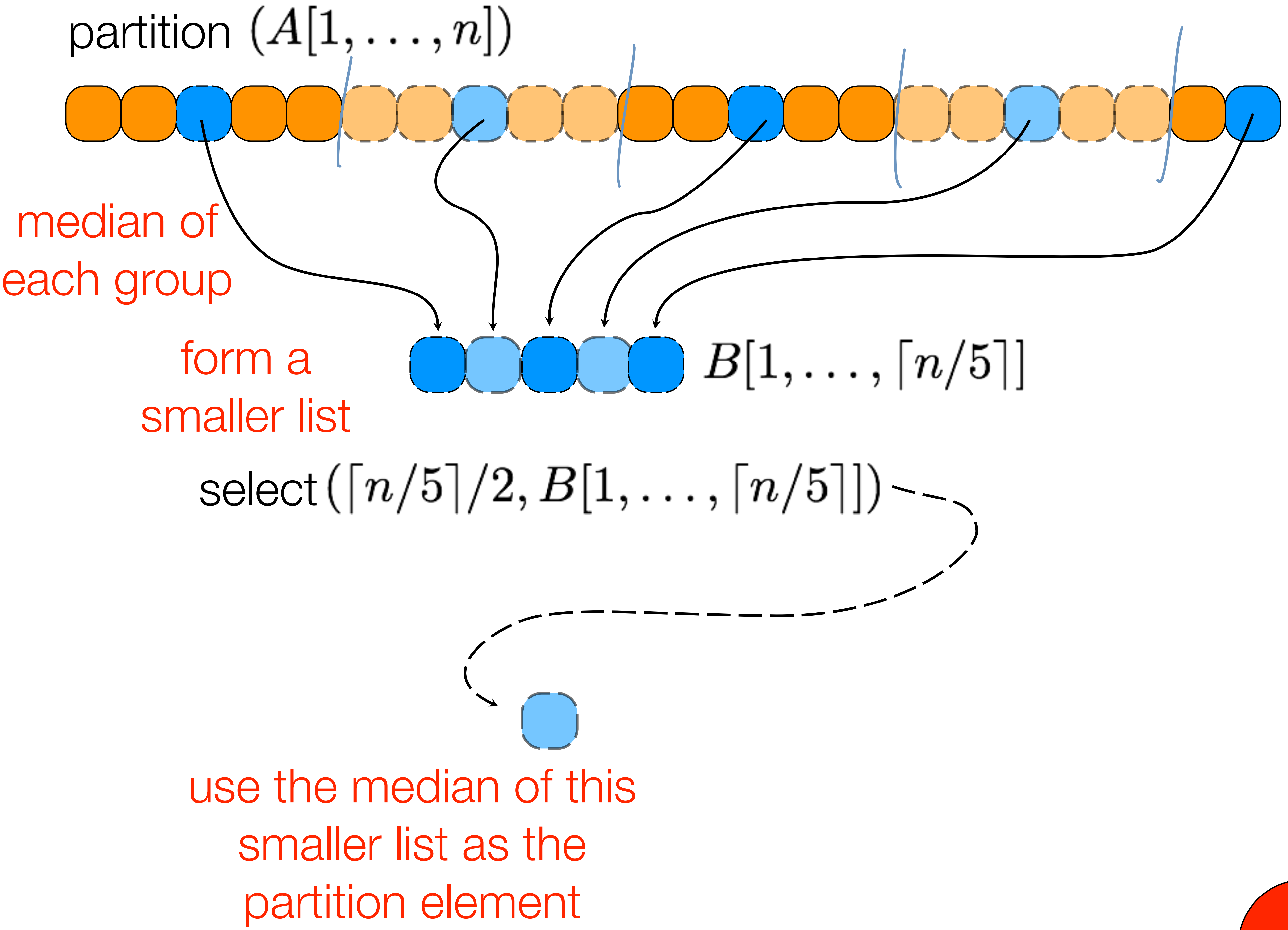


- divide the array into groups of 5.
  compute the median of each group.
- gather all of these medians into a new list of size $\lceil \frac{n}{5} \rceil$
- use select$(\lceil \frac{n}{10} \rceil$, blue array$)$ to compute the median of this smaller list.

$\leftarrow \lceil \frac{n}{5} \rceil$ elements.

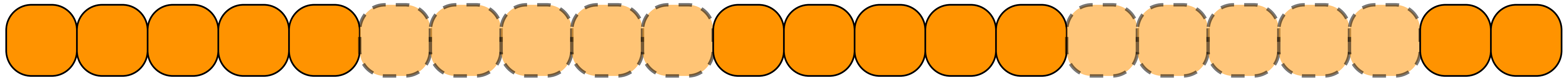
call select$(\lceil \frac{n}{10} \rceil$, blue array$)$
$i = ?$

will return the "median of medians"



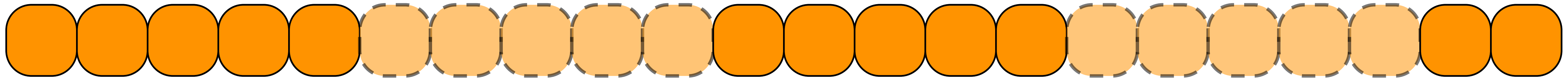
partition $(A[1,\ldots,n])$
median of each group
form a smaller list
$B[1,\ldots,\lceil n/5\rceil]$
select $(\lceil n/5\rceil/2, B[1,\ldots,\lceil n/5\rceil])$
use the median of this smaller list as the partition element

partition $(A[1, \ldots, n])$



1.
2.
3.
4.
5.

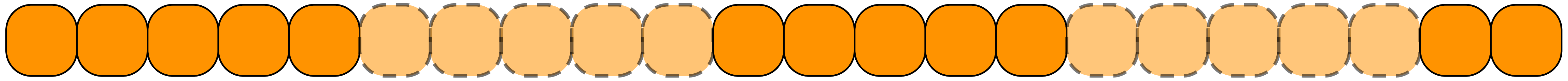
partition $(A[1, \ldots, n])$



- divide list into groups of 5 elements → $\Theta(n)$
- find median of each small list → $\Theta(n)$ ← there are $\lceil \frac{n}{5} \rceil$ small lists → finding median of each small list takes $\Theta(1)$ time
- gather all medians → $\Theta(n)$
- ↳ call select(...) on this sublist to find median → $T(\frac{n}{5})$
- return the result → $\Theta(1)$.

$$P(n) = T(\lceil \tfrac{n}{5} \rceil) + \Theta(n)$$

partition $(A[1,\ldots,n])$



divide list into groups of 5 elements

find median of each small list

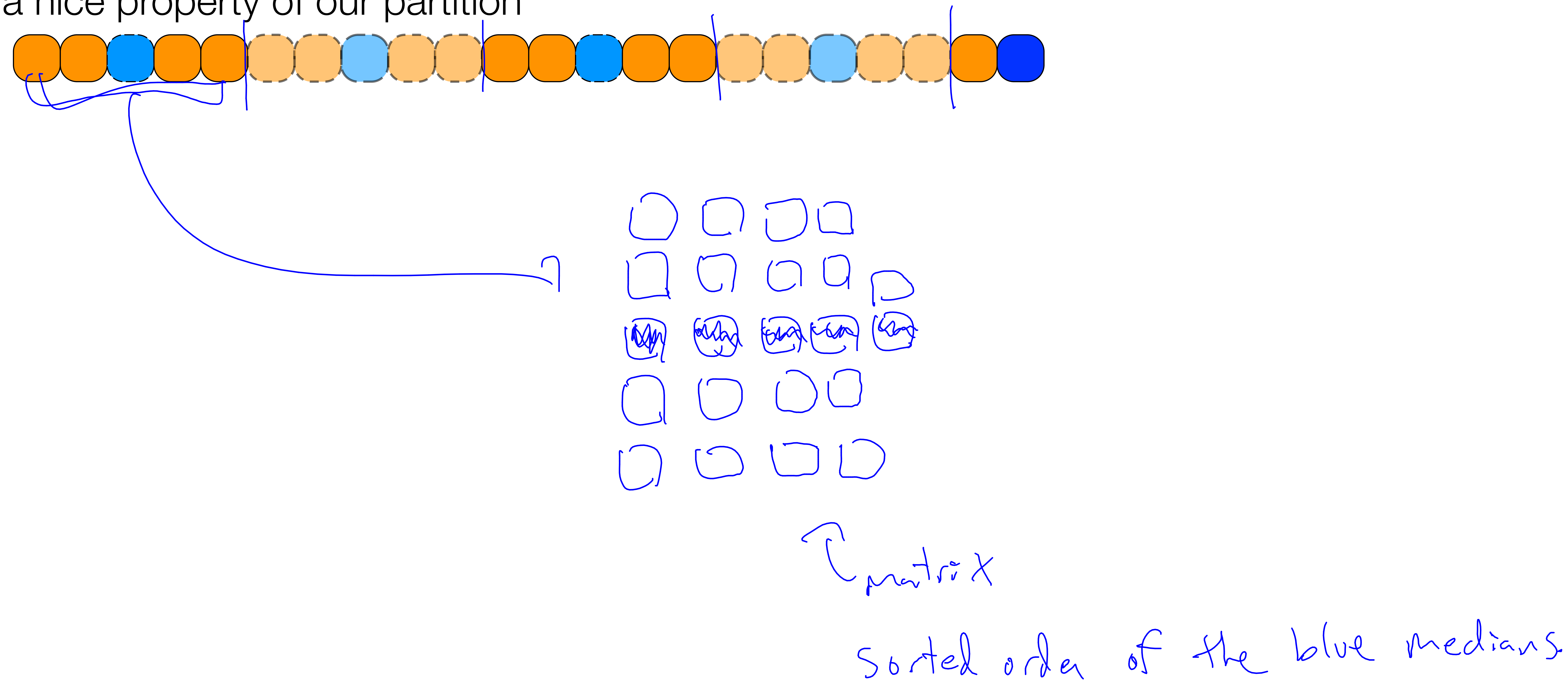
gather all medians

call select(...) on this sublist to find median

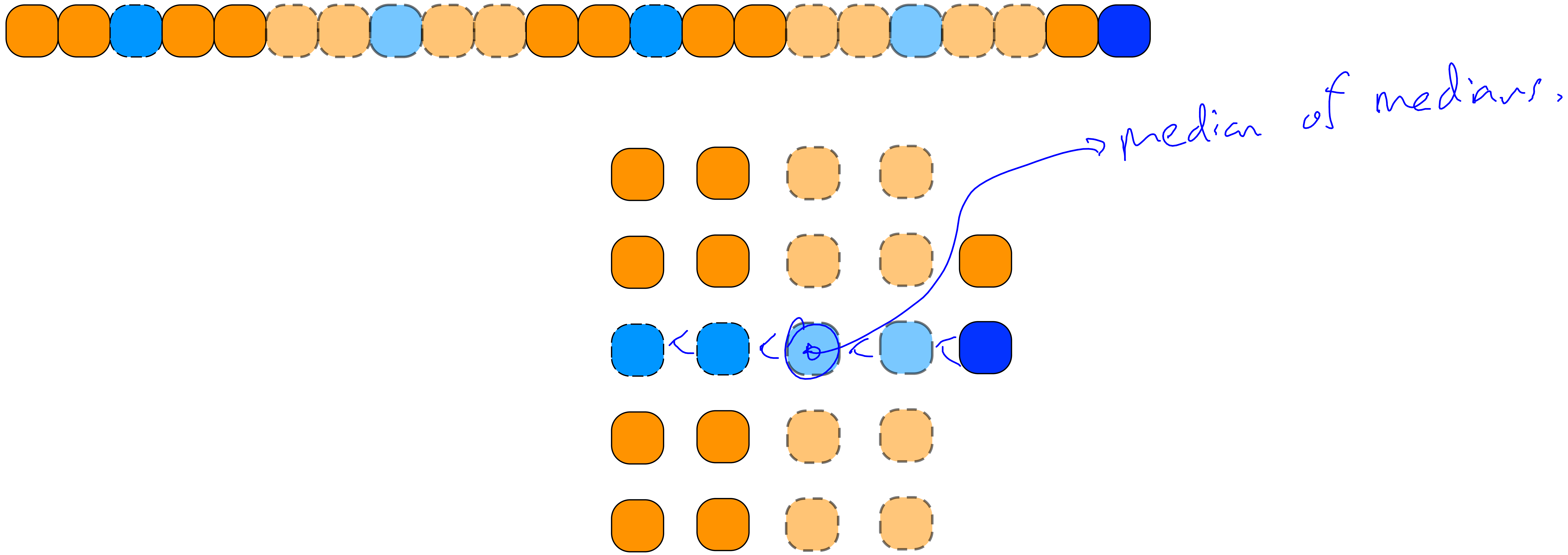
return the result

$$P(n) = S(\lceil n/5 \rceil) + O(n)$$

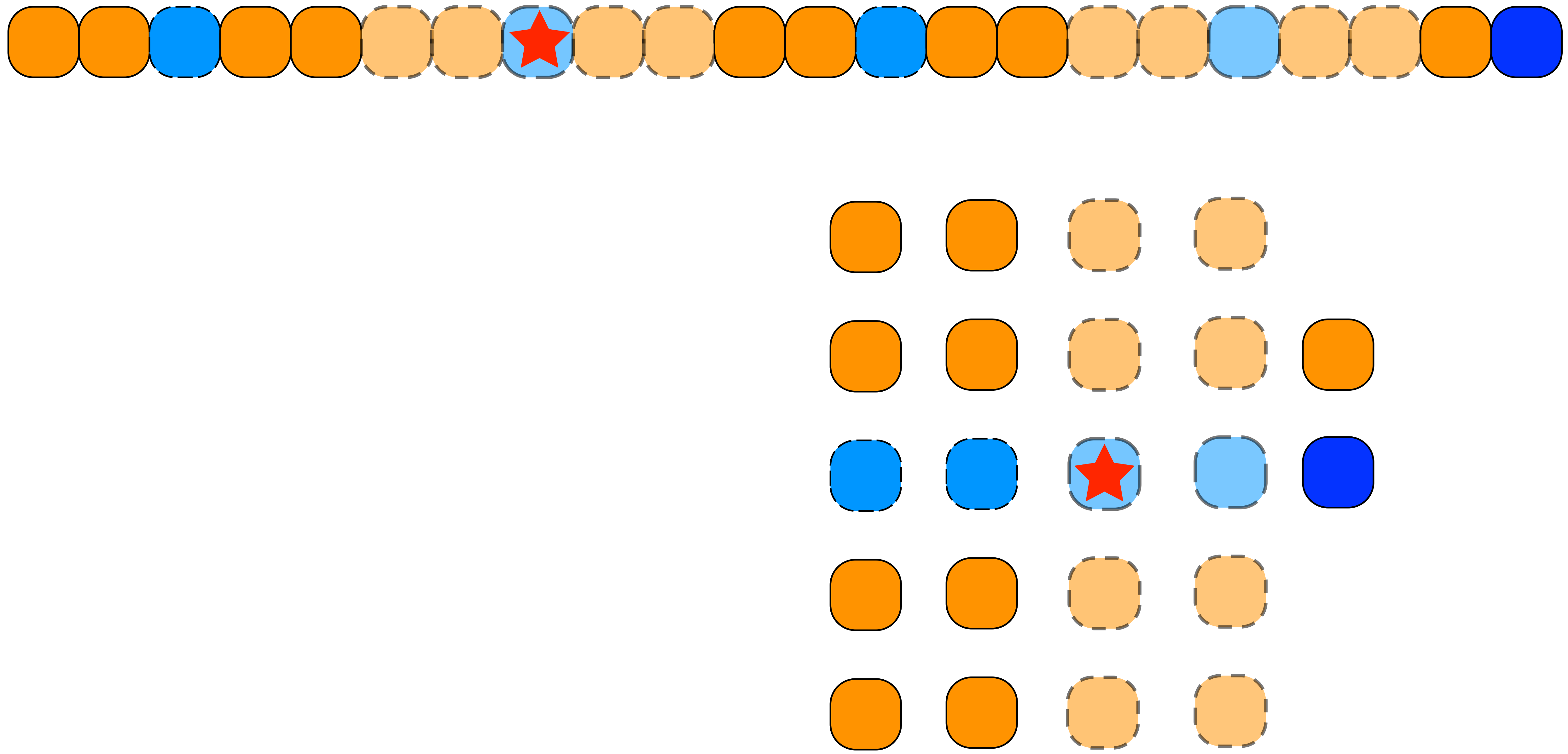
# a nice property of our partition



# a nice property of our partition

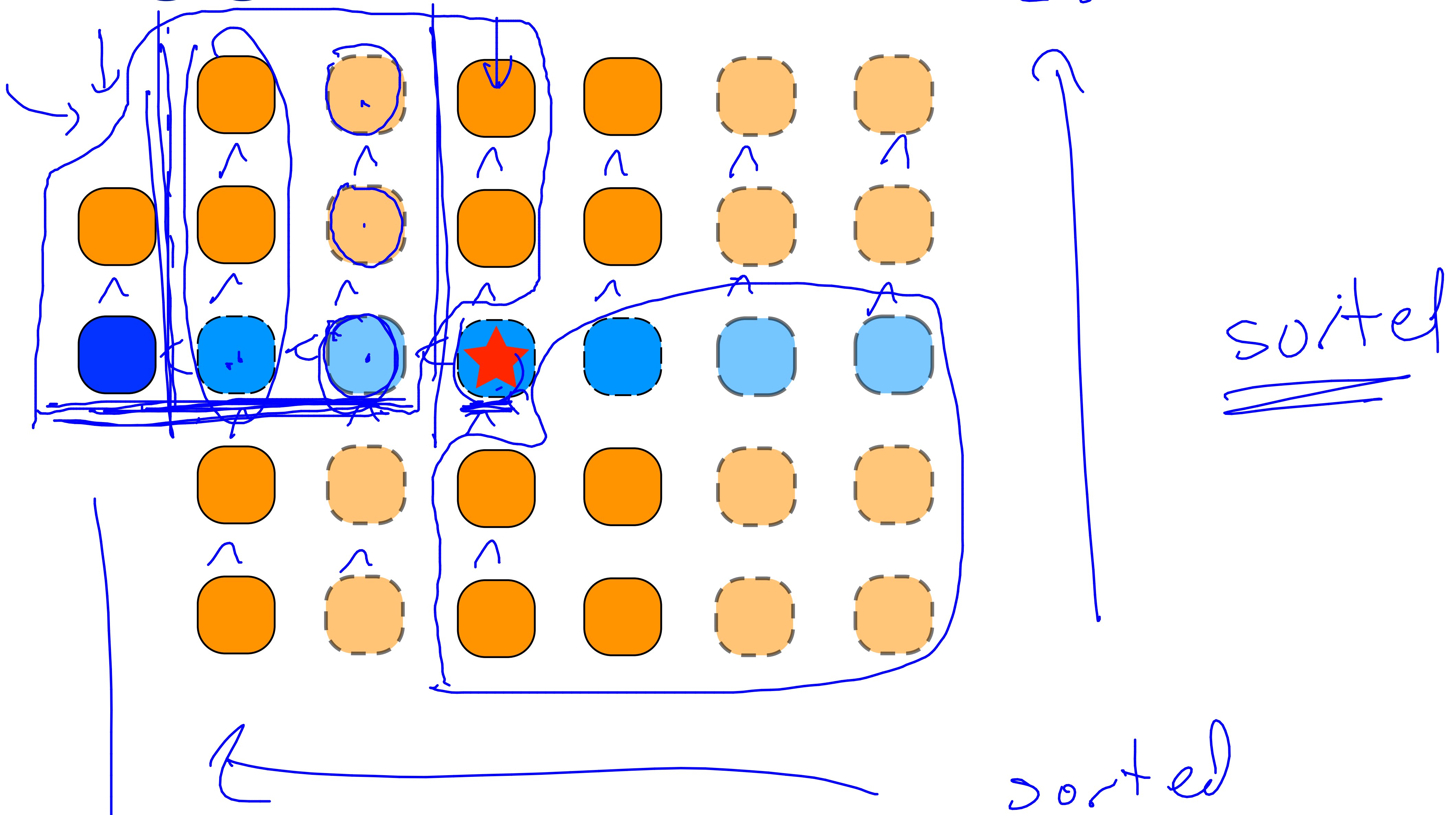


# a nice property of our partition



# SWITCH TO A BIGGER EXAMPLE

$3\left(\left\lceil \frac{1}{2}\left\lceil \frac{n}{5} \right\rceil \right\rceil - 2\right)$



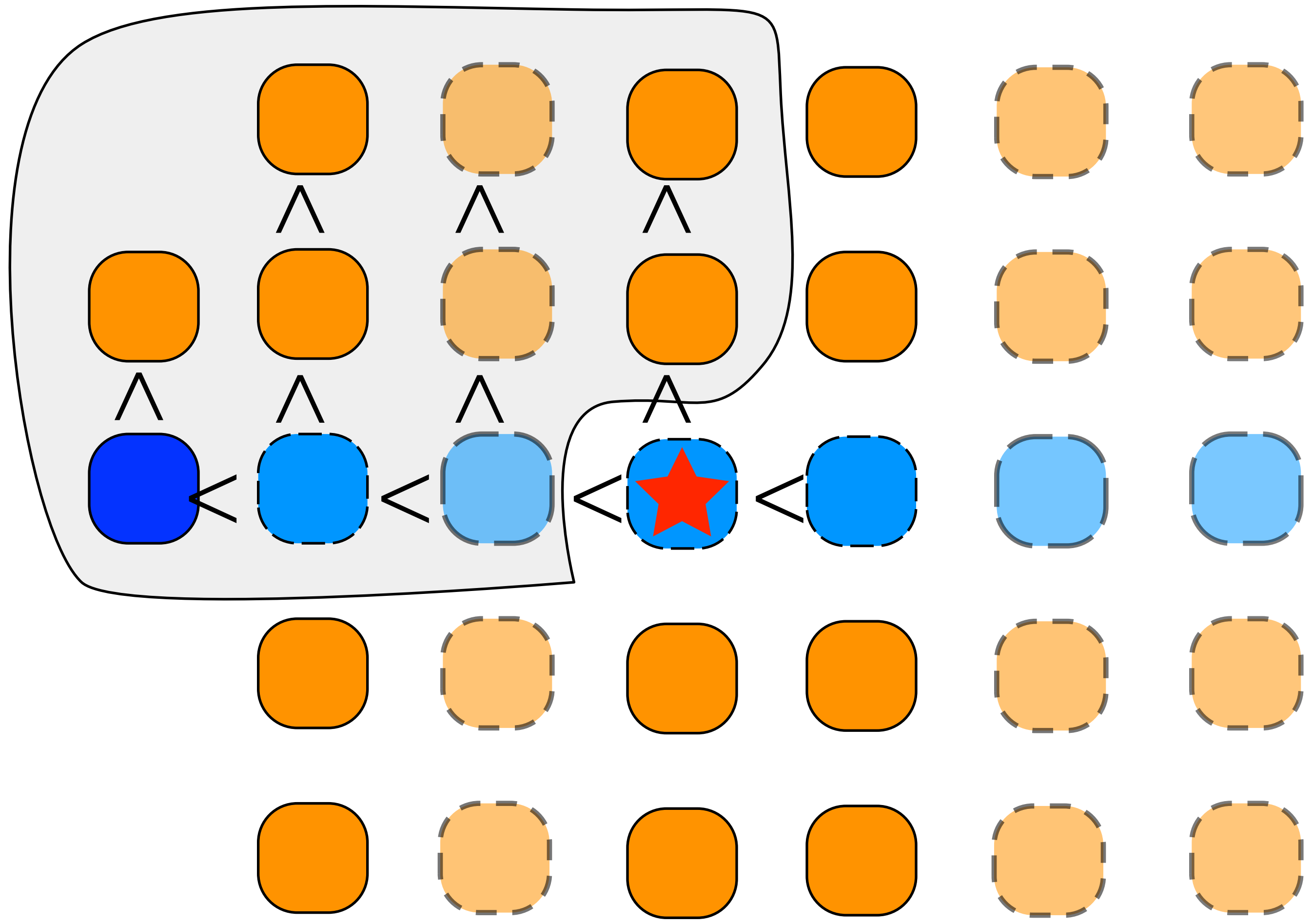

discard the first and last columns in this set

→ lets reason about the size of this set.

"set of elements in the input that are smaller than our median-of-medians"

→ How many columns? $\left\lceil \frac{n}{5} \right\rceil$ columns.

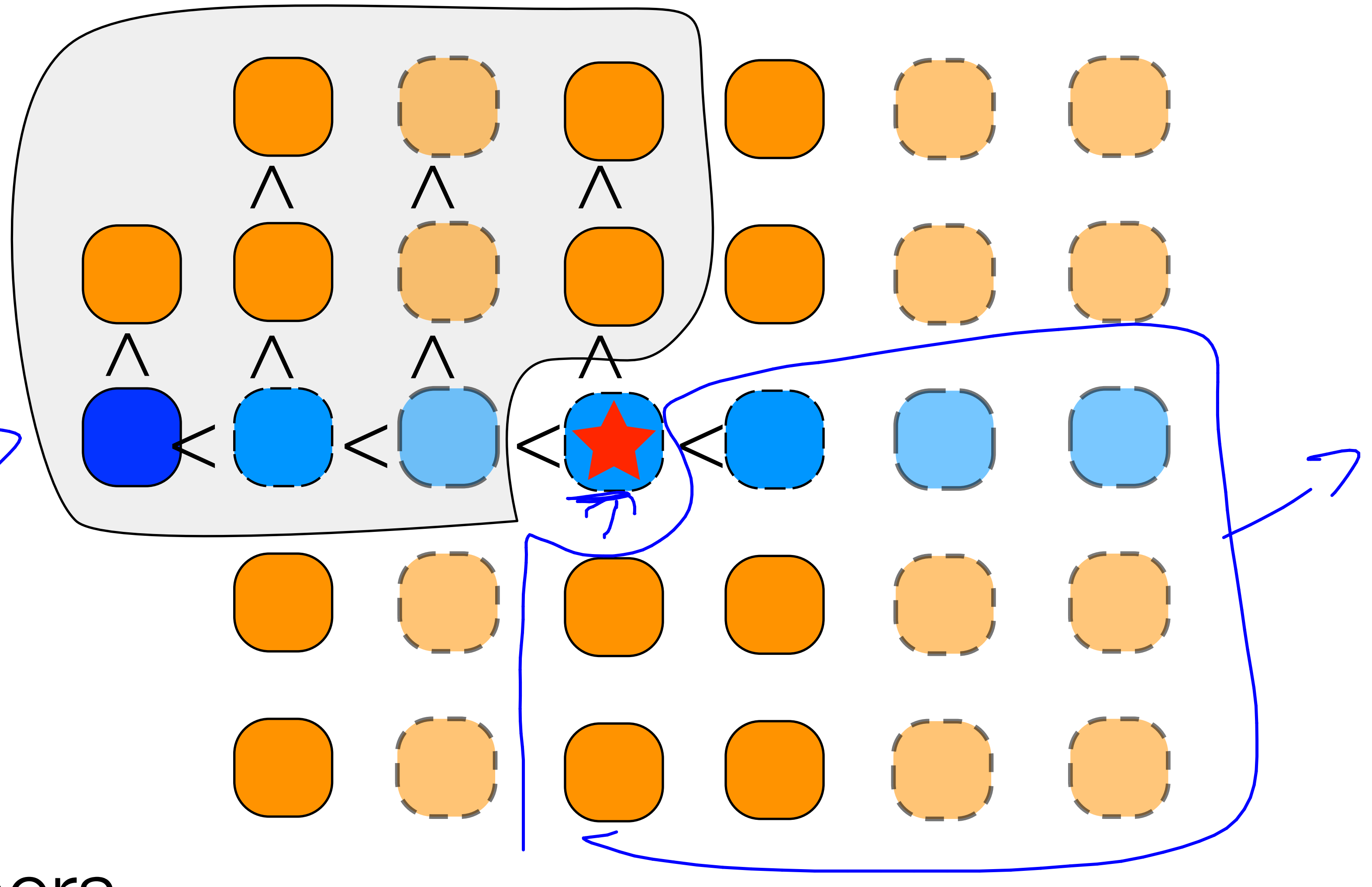
a nice property of our partition



# a nice property of our partition

$$3\left(\left\lceil \frac{1}{2}\lceil n/5\rceil \right\rceil - 2\right)$$

$$\geq \frac{3n}{10} - 6$$



$> \frac{3n}{10} - 6$

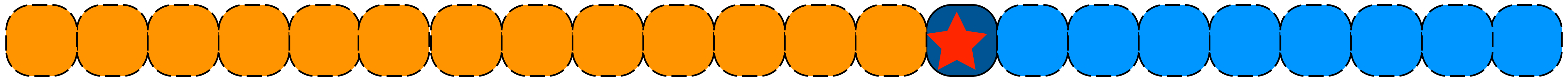
so at most

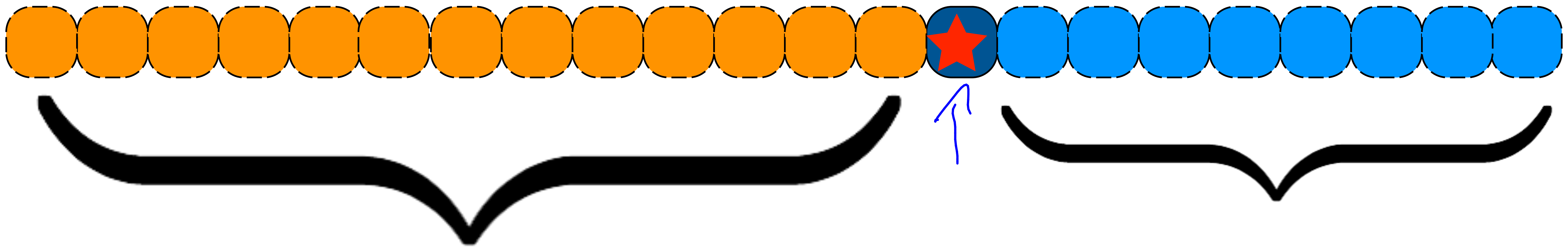
$\frac{7n}{10} + 6$ elements are smaller than *

this implies there are at most $\frac{7n}{10} + 6$ numbers

larger than ★
/smaller

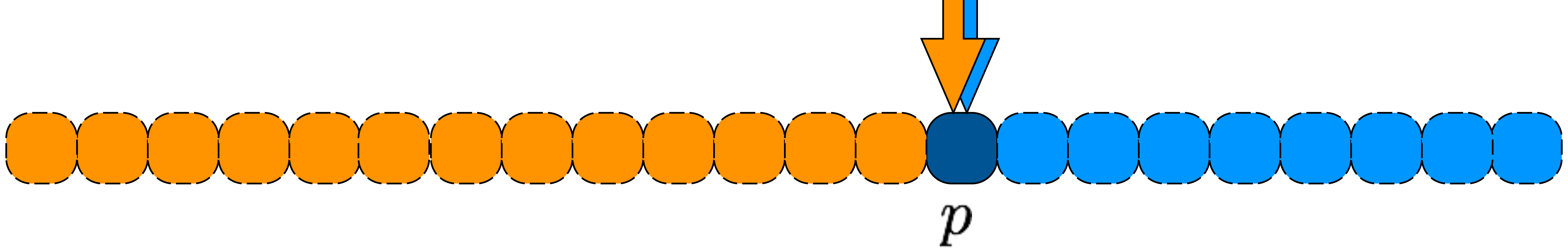
# a nice property of our partition





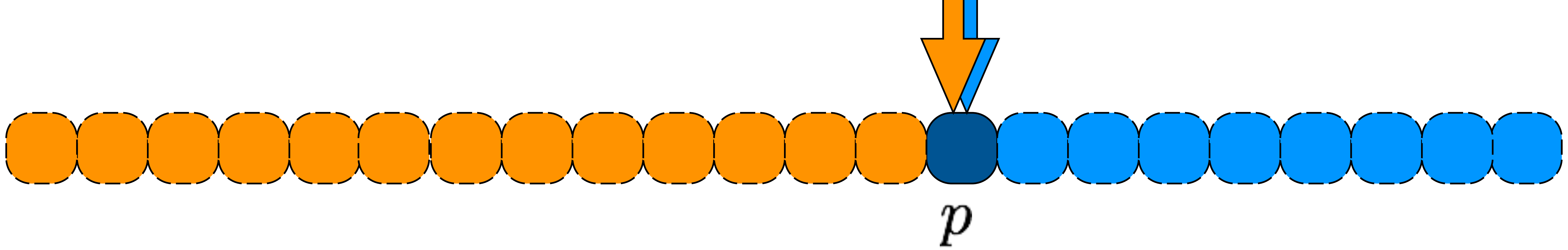
$$\leq \frac{7n}{10} + 6$$

$$\leq \frac{7n}{10} + 6$$




select $(i, A[1, \ldots, n])$

- Base case
- p = FINDPartition(A)
- partition A using p
- if index i = index p return p.
  else if i > p then select(p-i, A(p, ... n)]
  else select(i, A(1, ... p)]




select $(i, A[1, \ldots, n])$

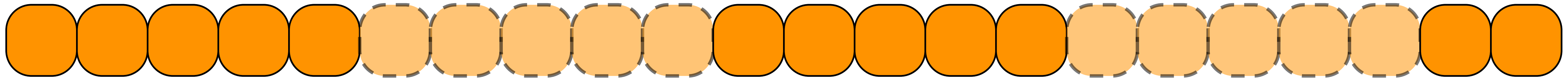
handle base case for small list
else pivot = FindPartitionValue(A,n) ⟷ $P(n) = T(\frac{n}{5}) + \Theta(n)$
partition list about pivot ⟶ $\Theta(n)$
if pivot is position $i$, return pivot ⟶ $\Theta(1)$
else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$
else select $((i-p-1), A[p+1, \ldots, n])$

$\leq T(\frac{7n}{10} + 6)$

$$T(n) = T(\frac{n}{5}) + T(\frac{7n}{10} + 6) + 2\Theta(n) = \Theta(n)$$

by guess & check.

FindPartition $(A[1,\ldots,n])$



divide list into groups of 5 elements

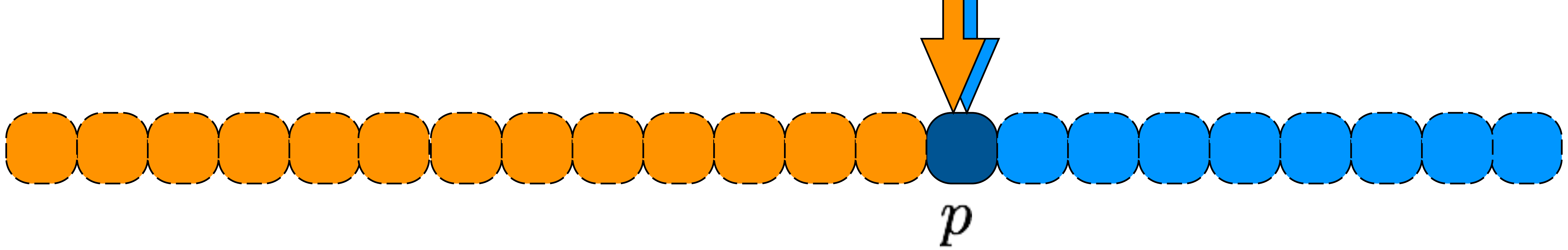
find median of each small list

gather all medians

call select(...) on this sublist to find median

return the result

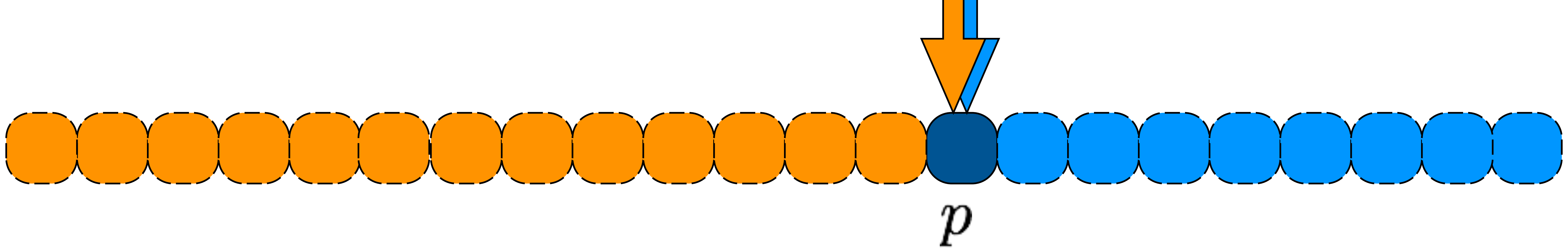
$$P(n) = S(\lceil n/5 \rceil) + O(n)$$




select $(i, A[1, \ldots, n])$

handle base case for small list
else pivot = FindPartitionValue(A,n)
partition list about pivot
if pivot is position $i$, return pivot
else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$
else select $((i-p-1), A[p+1, \ldots, n])$

$$S(n) = S(\lceil n/5 \rceil) + O(n) + S(7n/10 + 6)$$



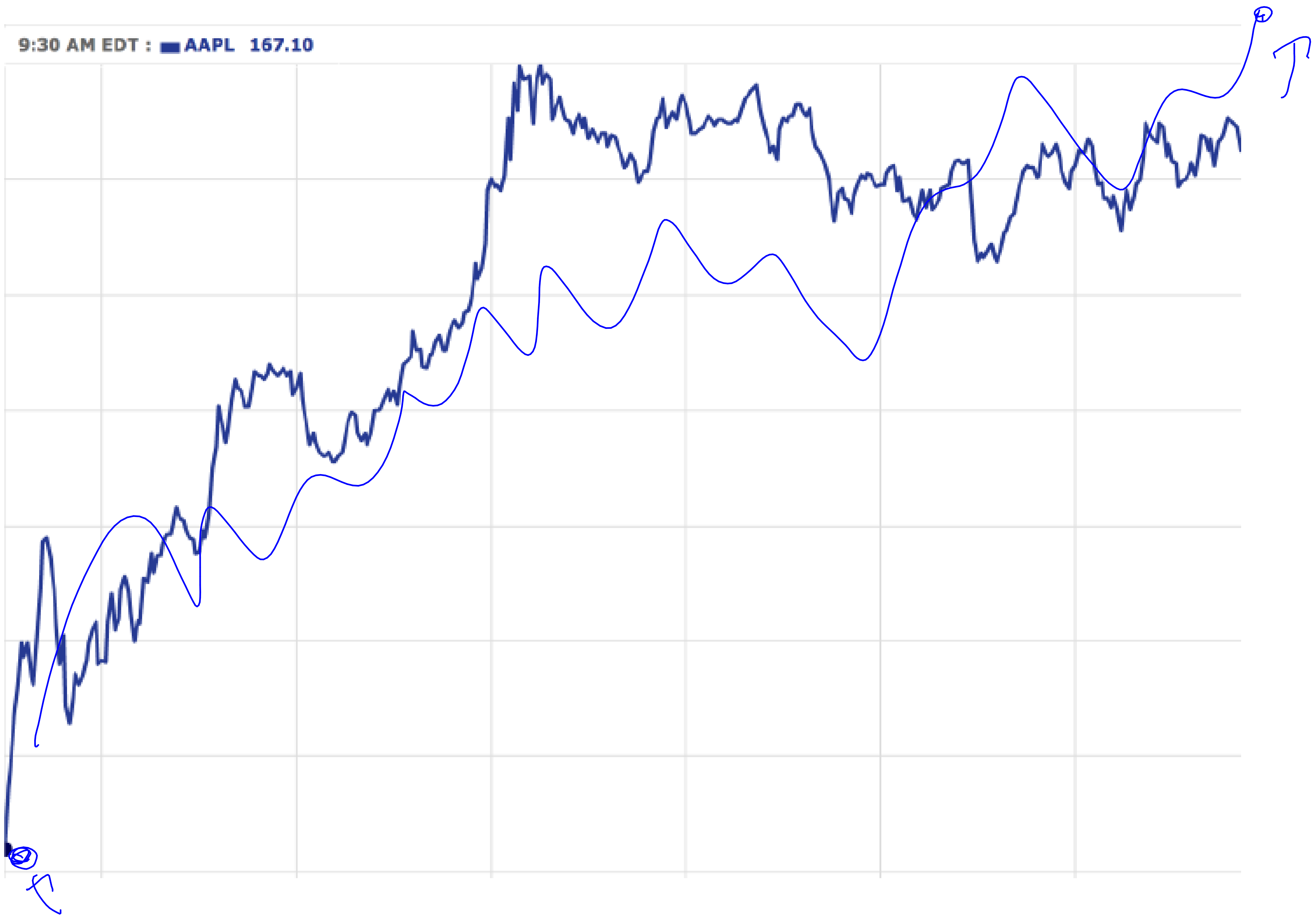

select $(i, A[1, \ldots, n])$

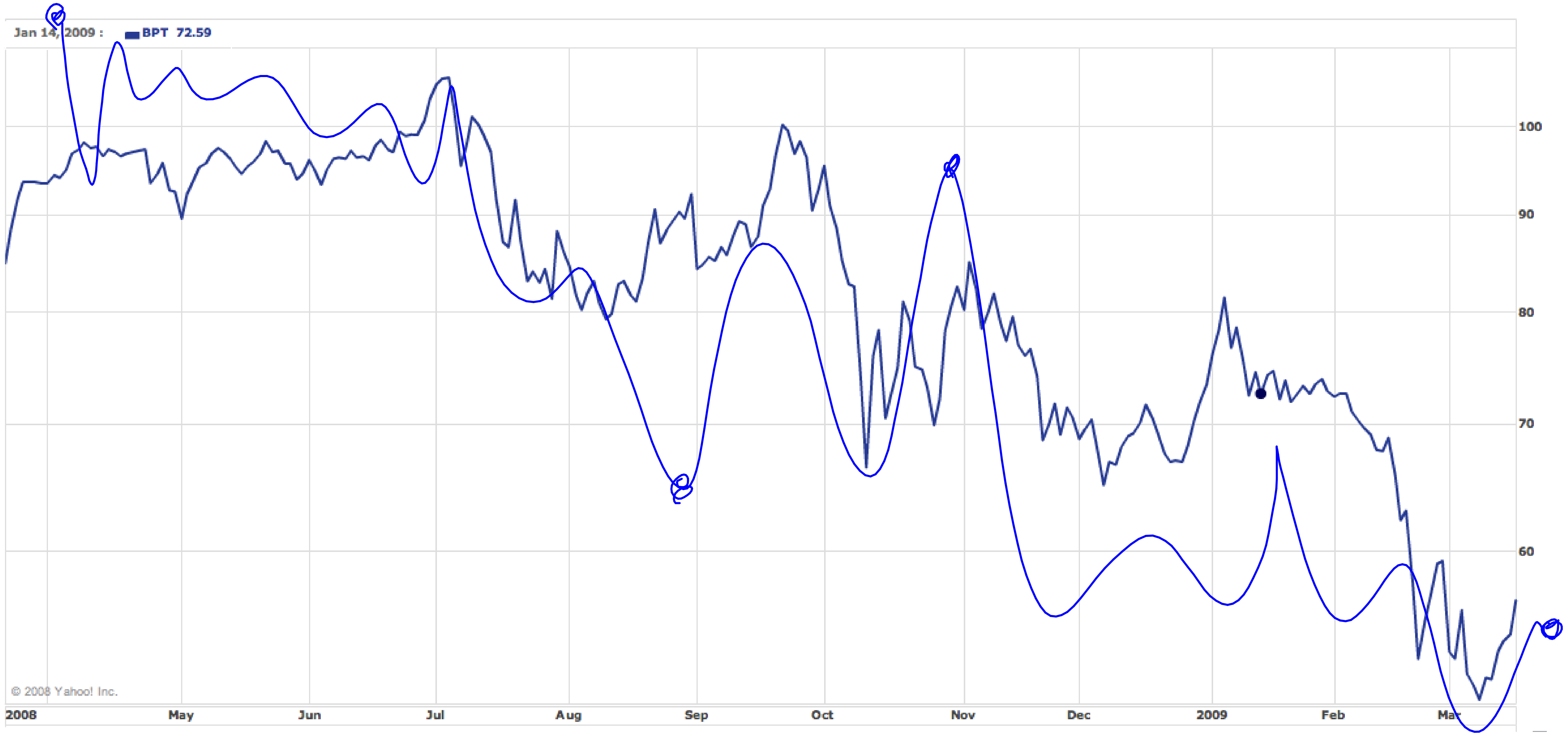
handle base case for small list
else pivot = FindPartitionValue(A,n)
partition list about pivot
if pivot is position $i$, return pivot
else if pivot is in position $> i$ select $(i, A[1, \ldots, p-1])$
else select $((i-p-1), A[p+1, \ldots, n])$

$$S(n) = S(\lceil n/5 \rceil) + O(n) + S(7n/10 + 6)$$

$$\Theta(n)$$

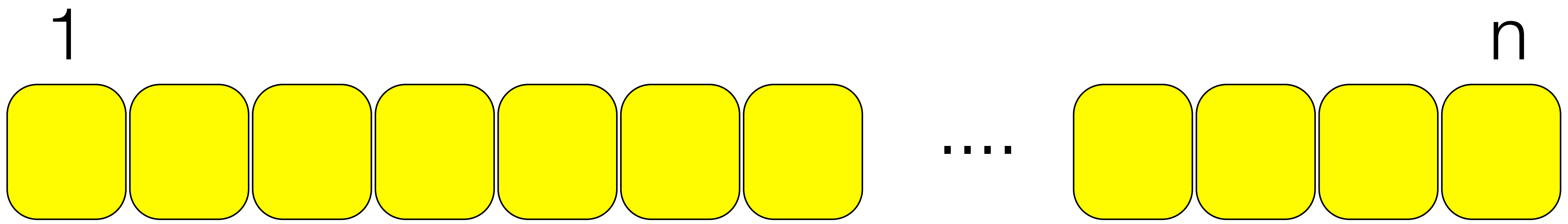
# arbitrage


9:30 AM EDT : AAPL 167.10


Jan 14, 2009 :
BPT 72.59
100
90
80
70
60
© 2008 Yahoo! Inc.
2008
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
2009
Feb
Mar

12:38 PM EDT : AIG 40.58
AIG.
© 2008 Yahoo! Inc.
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00

input: array of n numbers - closing tickers for some stock.

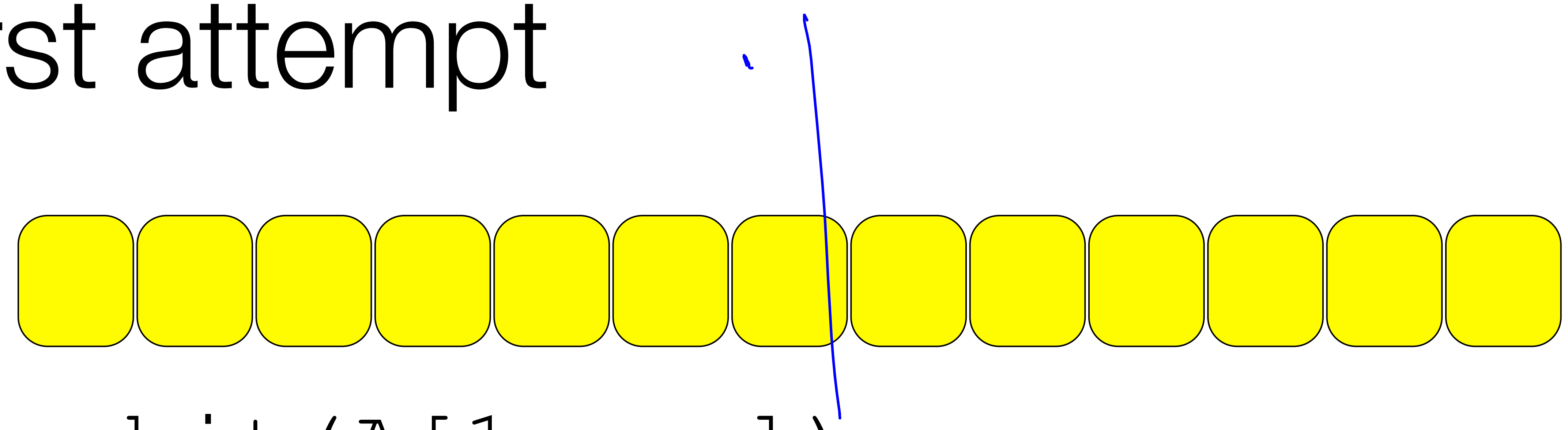



**goal:** output a pair of days $i, j$ such that $i < j$

$$\max (A[j] - A[i])$$

clearly $\Theta(n^2)$ solution

# first attempt



`arbit(A[1...n])`

$max_L \leftarrow Arbit(A[1, \frac{n}{2}])$

$max_R \leftarrow Arbit(A[\frac{n}{2}, n])$

$m_L :=$ find min on left

$m_R :=$ find max on right

return $max(max_L, max_R, m_R - m_L)$

# first attempt

```
arbit(A[1...n])
    base case if |A|<=2
    lg = arbit(left(A))
    rg = arbit(right(A))
    minl = min(left(A))
    maxr = max(right(A))
    return max{maxr-minl,lg,rg}
```

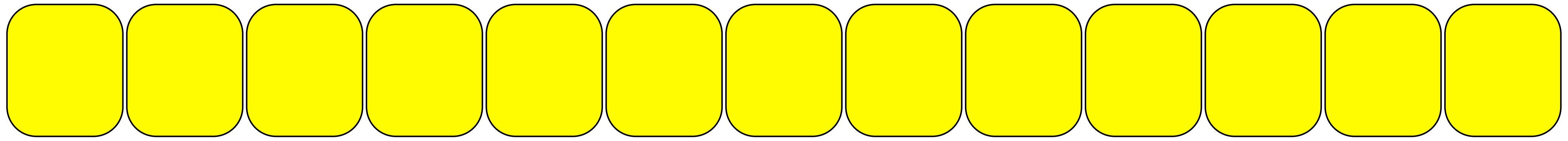
lg, rg → $2T(n/2)$

minl → $\Theta(n)$; maxr → $\Theta(n)$ → remove these steps??

return → $\Theta(1)$

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n) = \Theta(n\log n)$$

# first attempt: time $\Theta(n \log n)$



```
arbit(A[1...n])
    base case if |A|<=2
    lg = arbit(left(A))
    rg = arbit(right(A))
    minl = min(left(A))
    maxr = max(right(A))

    return max{maxr-minl,lg,rg}
```

# better approach

$T(n) = 2T(\frac{n}{2}) + \Theta(1) = \Theta(n)$

goal ↑

idea: arbit returns both the i, j pair

AND the max, min elements in the array

# better approach

Can we find a solution that has $T(n) = 2T(n/2) + O(1)$ ?

# better approach

Can we find a solution that has $T(n) = 2T(n/2) + O(1)$ ?

```
minl = min(left(A))
maxr = max(right(A))
return max{maxr-minl,lg,rg}
```

# second attempt

```
arbit+(A[1...n])
   base case if |A|<=2
```

$(i_L, j_L, max_L, min_L) = \text{Arbit}(\text{Left half})$

$(i_R, j_R, max_R, min_R) = \text{Arbit}(\text{Right})$

return $\max\left( (i_L, j_L), (i_R, j_R), (max_R - min_L) \right)$

# second attempt

```
arbit+(A[1...n])
   base case if |A|=1
  (lg,minl,max) = arbit(left(A))
  (rg,mi,maxr) = arbit(right(A))
  return max{maxr-minl,lg,rg}
```

, min(minl, mi), max(max, maxr)

$$T(n) = 2T(\tfrac{n}{2}) + \Theta(1) = \Theta(n)$$

# Fast Fourier Transform

# big ideas:

$$A(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1}$$

↰ polynomial

output FFT: ⇓

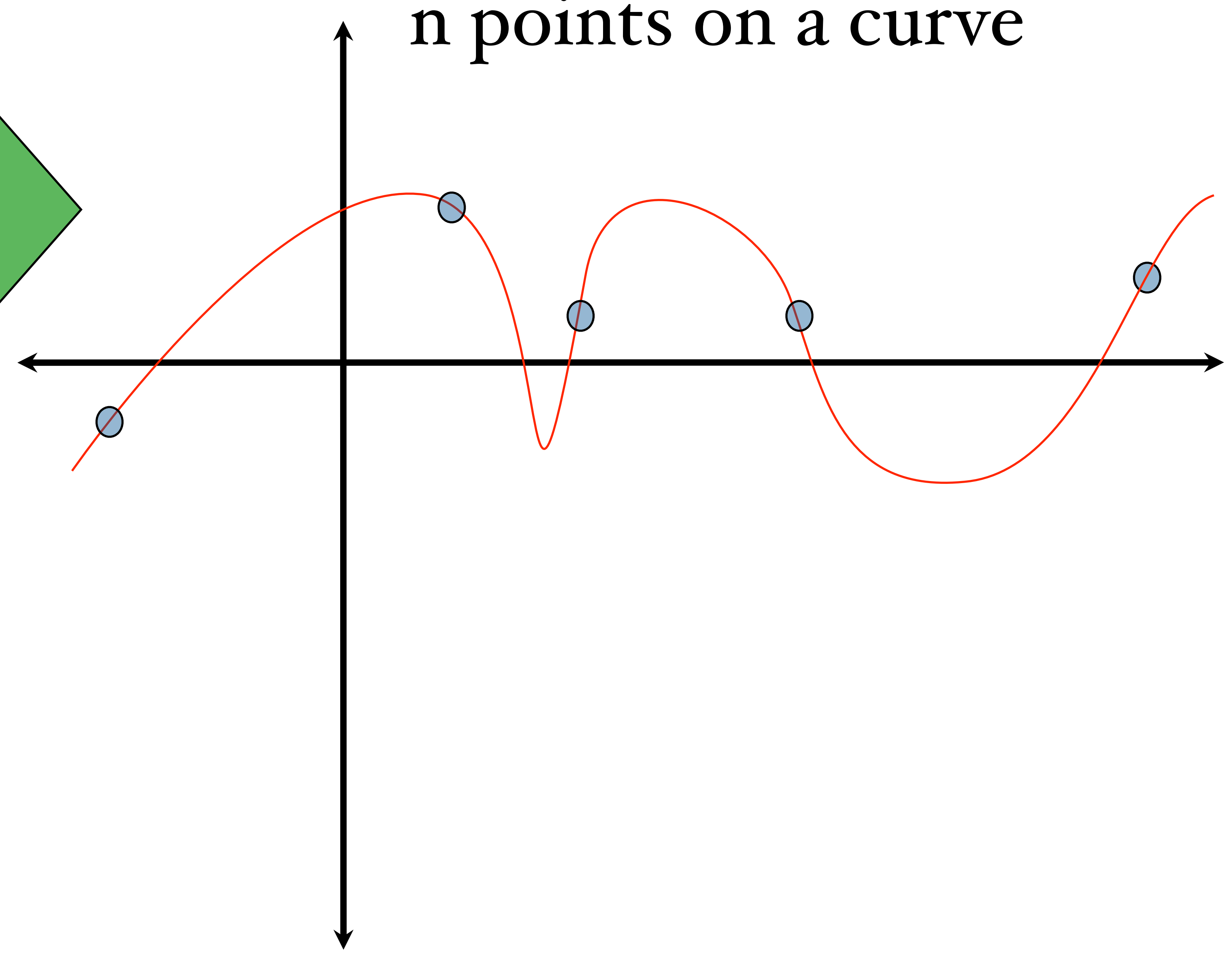
$A(\omega_0)$ ... $A(\omega_n)$

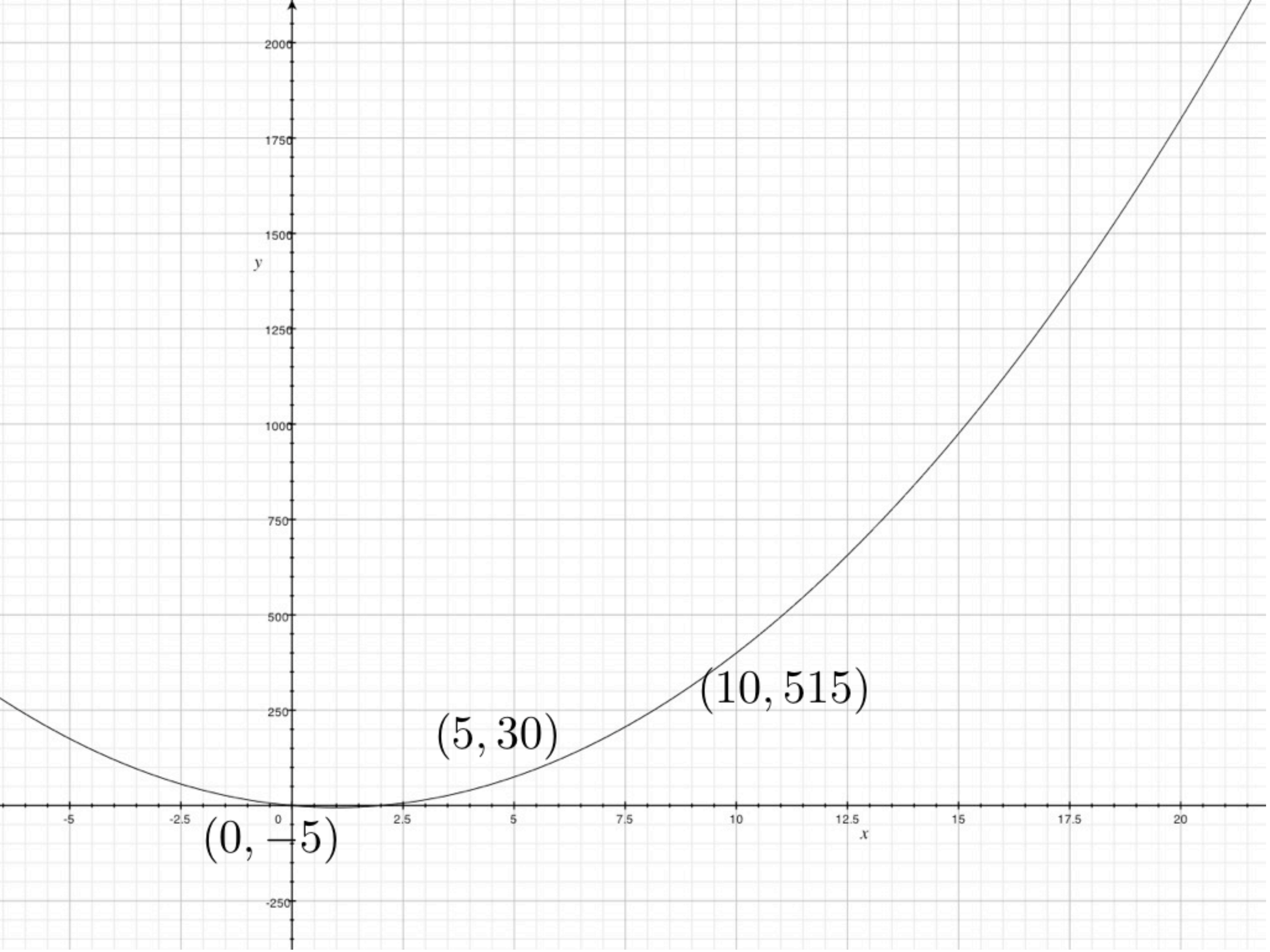
eval of polynomial A on n points

degree $n-1$ polynomial

$$A(x)$$

n points on a curve





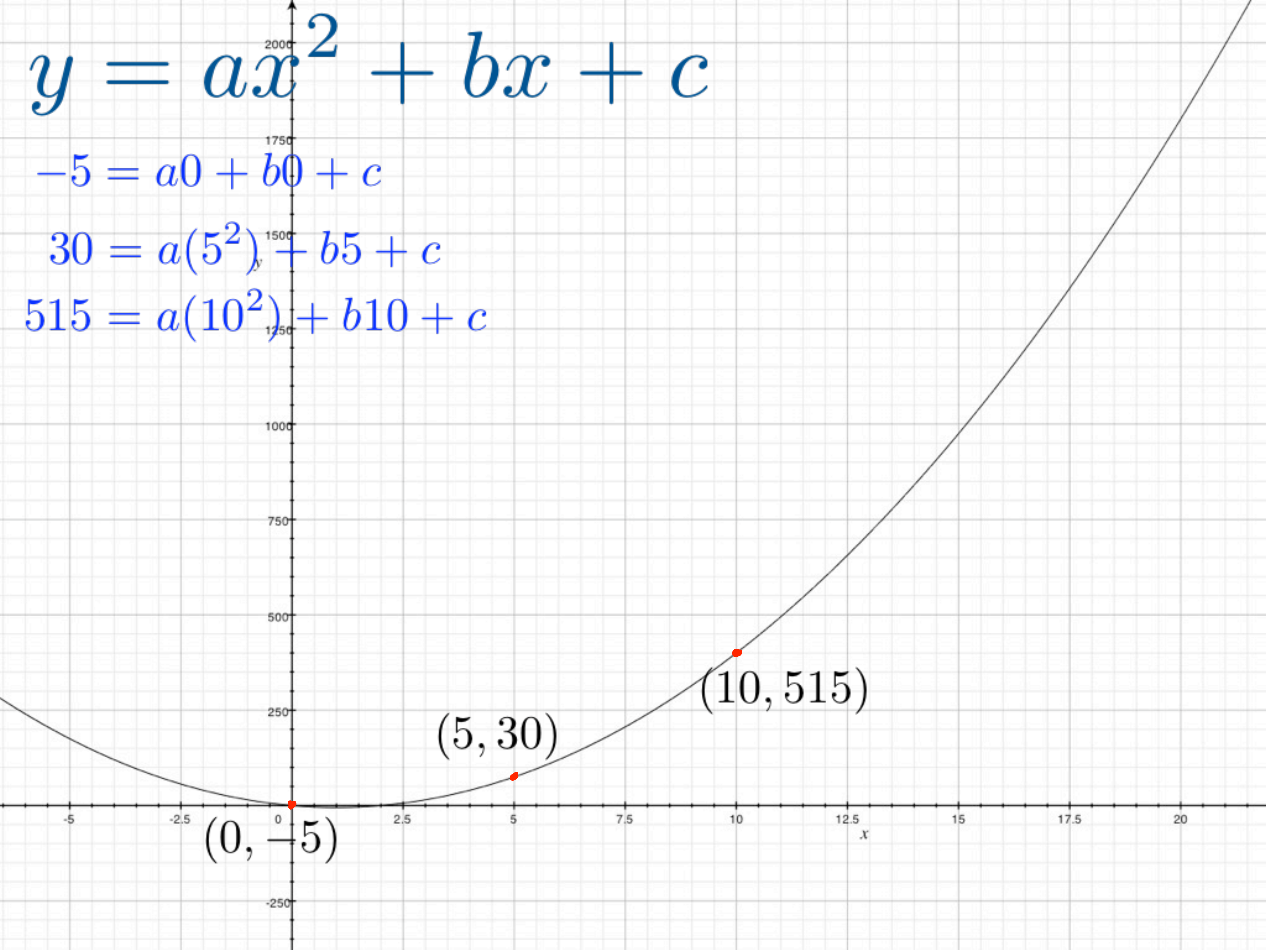
y
x
2000
1750
1500
1250
1000
750
500
250
-250
-5
-2.5
0
2.5
5
7.5
10
12.5
15
17.5
20
(10, 515)
(5, 30)
(0, −5)

$$y = ax^2 + bx + c$$

$$-5 = a0 + b0 + c$$

$$30 = a(5^2) + b5 + c$$

$$515 = a(10^2) + b10 + c$$



# FFT

input: $a_0, a_1, a_2, \ldots, a_{n-1}$

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

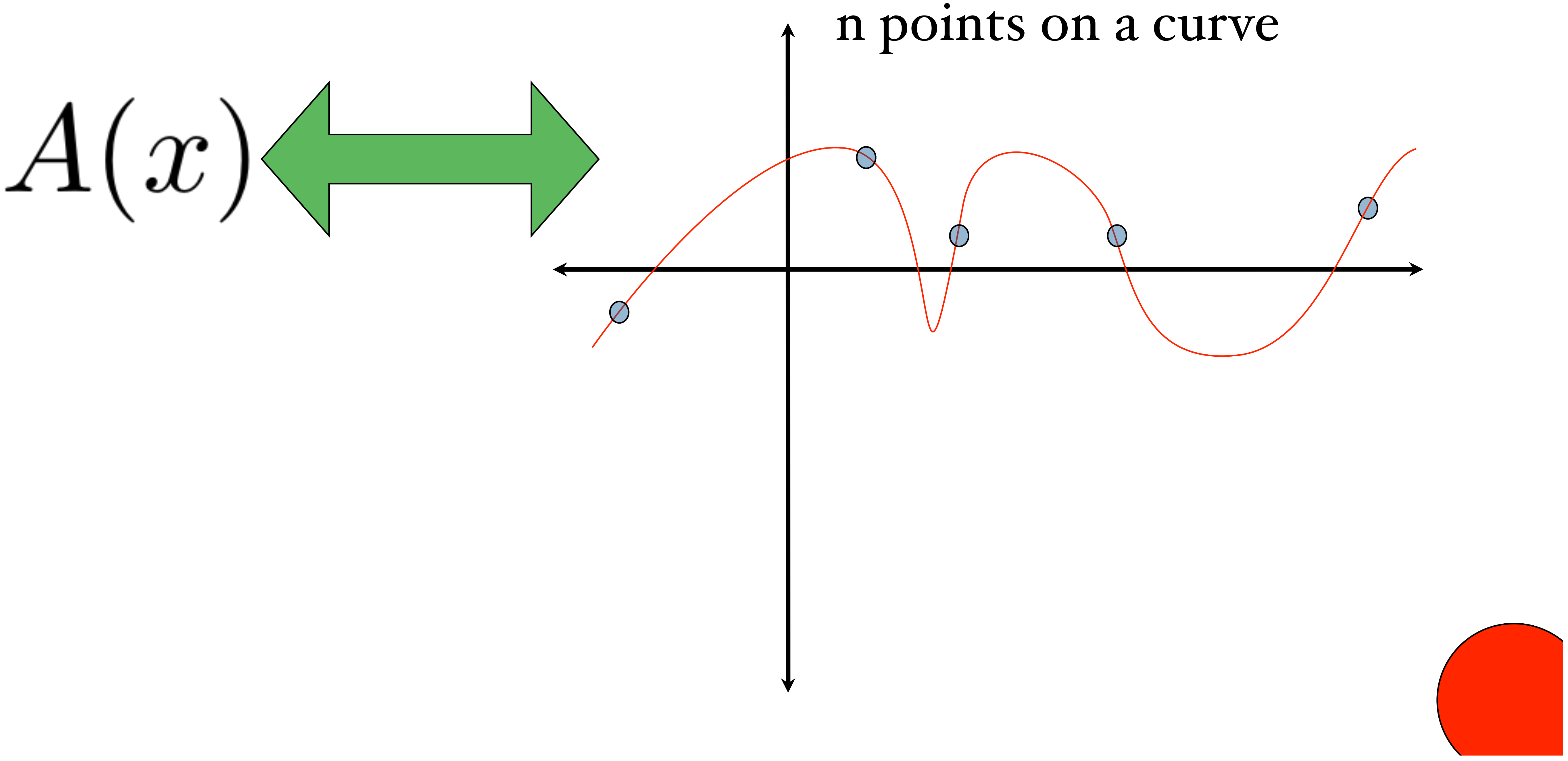
output:

# FFT

input: $a_0, a_1, a_2, \ldots, a_{n-1}$

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

output: evaluate polynomial A at (any) n different points.



$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

Brute force method to evaluate A at n points:

solve the large problem by

solving smaller problems

and combining solutions

T(n)=

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

$$
\begin{aligned}
A(x) &= a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} \\
&= a_0 + a_2 x^2 + a_4 x^4 + \cdots + a_{n-2} x^{n-2} \\
&\quad + a_1 x + a_3 x^3 + a_5 x^5 + \cdots + a_{n-1} x^{n-1}
\end{aligned}
$$

$$
\begin{aligned}
A_e(x) &= a_0 + a_2 x + a_4 x^2 + \cdots + a_n x^{(n-2)/2} \\
A_o(x) &= a_1 + a_3 x + a_5 x^2 + \cdots + a_{n-1} x^{(n-2)/2}
\end{aligned}
$$

$$
A(x) = A_e(x^2) + x A_o(x^2)
$$

$$A(x) = A_e(x^2) + xA_o(x^2)$$

suppose we had eval of Ae,Ao on {4,9,16,25}

$$A(x) = A_e(x^2) + xA_o(x^2)$$

suppose we had eval of Ae,Ao on {4,9,16,25}

$$A(2) = A_e(4) + 2A_o(4)$$

$$A(-2) = A_e(4) + (-2)A_o(4)$$

$$A(3) = A_e(9) + 3A_o(9)$$

$$A(-3) = A_e(9) + (-3)A_o(9)$$

Last remaining issue:

# roots of unity

$$x^n = 1$$

should have n solutions

what are they?

$$e^{2\pi i} = 1$$

consider $e^{2\pi i j/n}$ for j=0,1,2,3,...,n-1

$$e^{2\pi i} = 1$$

consider $e^{2\pi i j/n}$

$$\left[e^{(2\pi i/n)j}\right]^n = \left[e^{(2\pi i/n)n}\right]^j = \left[e^{2\pi i}\right]^j = 1^j$$

$e^{2\pi i j/n}$ = $\omega_{j,n}$ is an $n^{th}$ root of unity

$$\omega_{0,n}, \omega_{2,n}, \ldots, \omega_{n-1,n}$$

# What is this number?

$e^{2\pi ij/n}$ = $\omega_{j,n}$ is an $n^{th}$ root of unity

$$e^{2\pi ij/n} = \cos(2\pi j/n) + i\sin(2\pi j/n)$$

# Why is this true?

$e^{2\pi ij/n}$ = $\omega_{j,n}$ is an $n^{th}$ root of unity

# Taylor series expansion

$$f(y) = f(a) + \frac{f'(a)}{1!}(y-a) + \frac{f''(a)}{2!}(y-a)^2 + \frac{f'''(a)}{3!}(y-a)^2 +$$

$$e^y$$

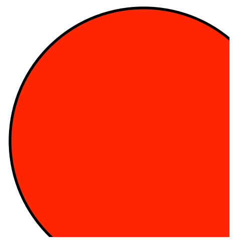
$$e^y = 1 + y + \frac{y^2}{2!} + \frac{y^3}{3!} + \frac{y^4}{4!} + \cdots$$

$$e^{ix} = \cos(x) + i\sin(x)$$

$$e^{ix} = \cos(x) + i\sin(x)$$

$$e^{2\pi i} = 1$$

$$e^{2\pi ij/n} = \cos(2\pi j/n) + i\sin(2\pi j/n)$$



$$e^{2\pi i} = 1$$

consider $e^{2\pi i j/n}$

$$\left[e^{(2\pi i/n)j}\right]^n = \left[e^{(2\pi i/n)n}\right]^j = \left[e^{2\pi i}\right]^j = 1^j$$

$e^{2\pi i j/n}$ = $\omega_{j,n}$ is an n^th root of unity

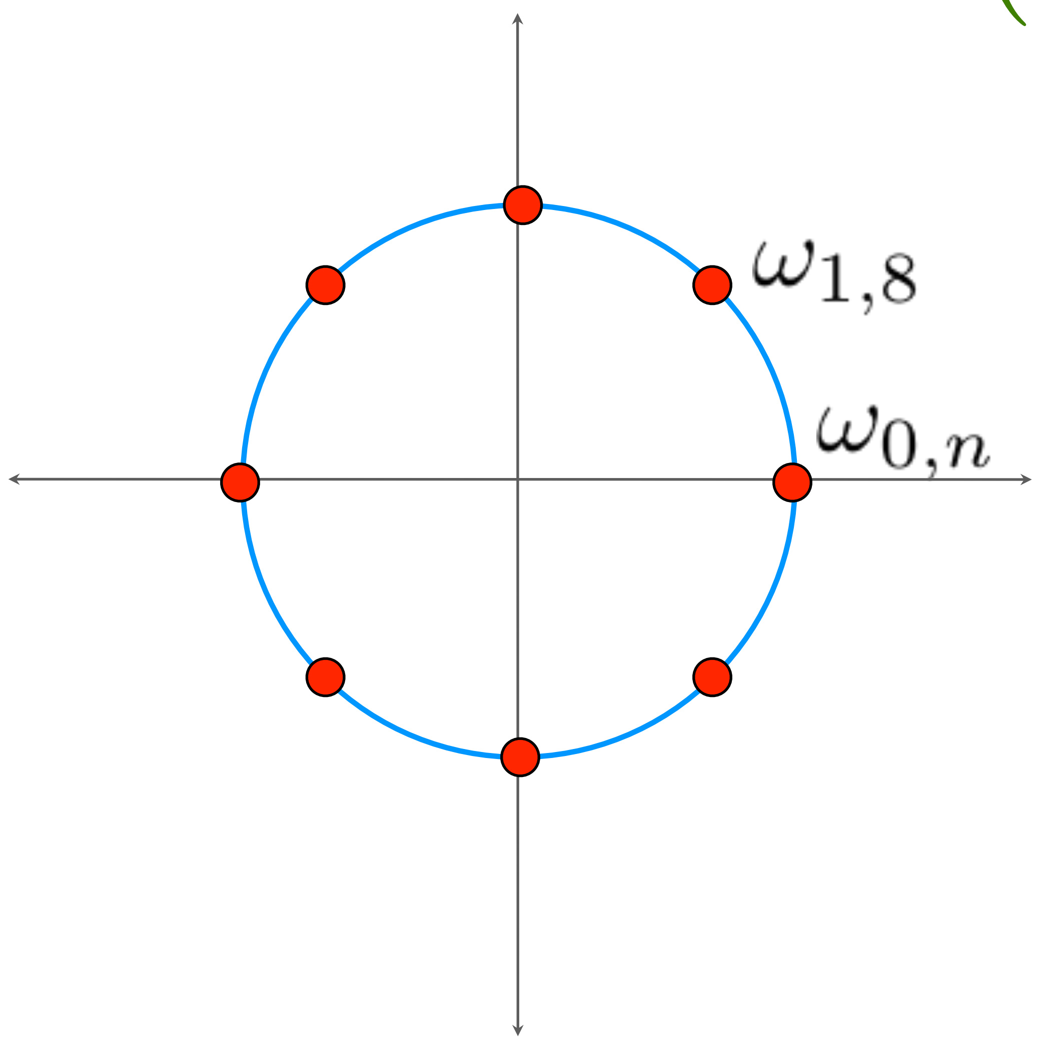
$$\omega_{0,n}, \omega_{2,n}, \ldots, \omega_{n-1,n}$$

# roots of unity

$$x^n = 1$$

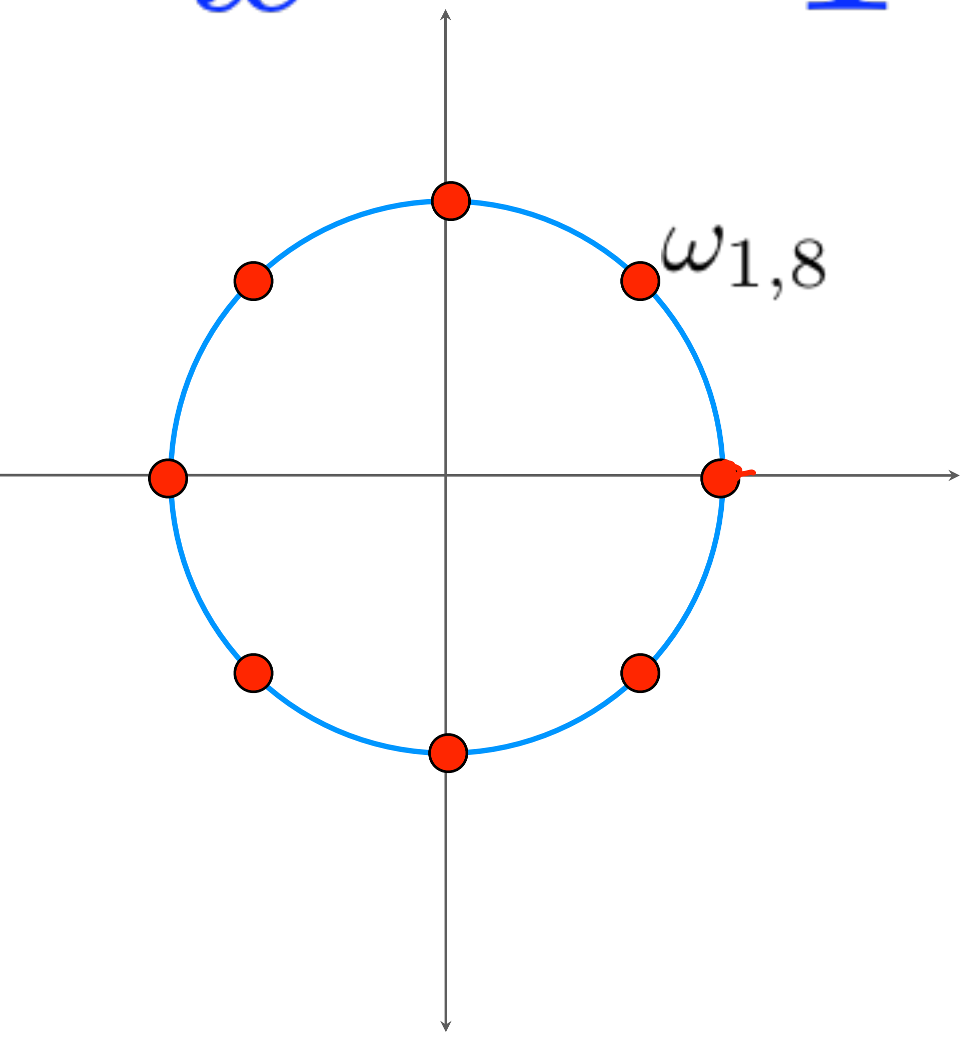
should have n solutions

$$e^{2\pi ij/n} = \cos(2\pi j/n) + i\sin(2\pi j/n)$$

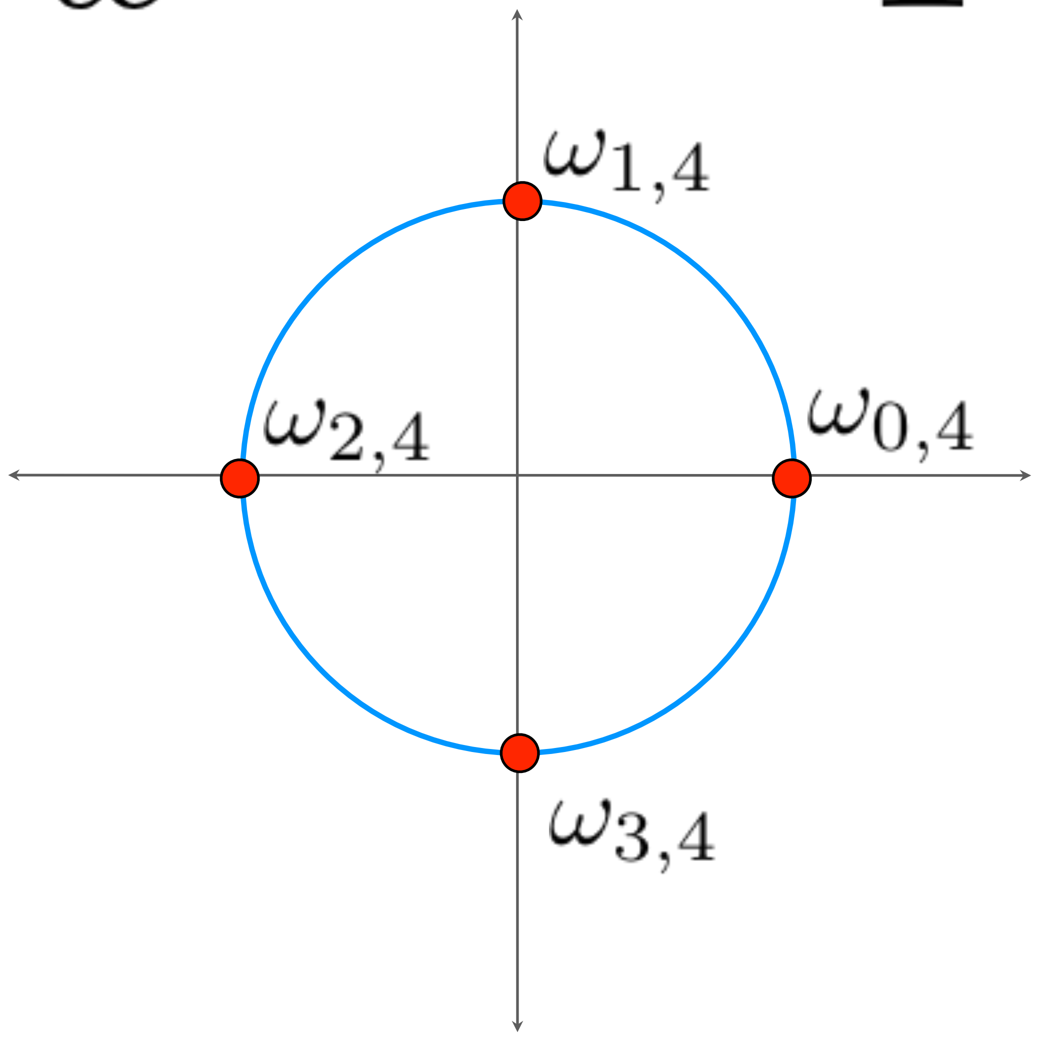


# squaring the n$^{th}$ roots of unity

$$x^n = 1$$




$$x^{n/2} = 1$$



Fact: squaring an $n^{th}$ root produces an $n/2^{th}$ root

example: $\omega_{1,8} =$

Fact: squaring an n[th] root produces an n/2[th] root

example: $$\omega_{1,8} = \left(\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)$$

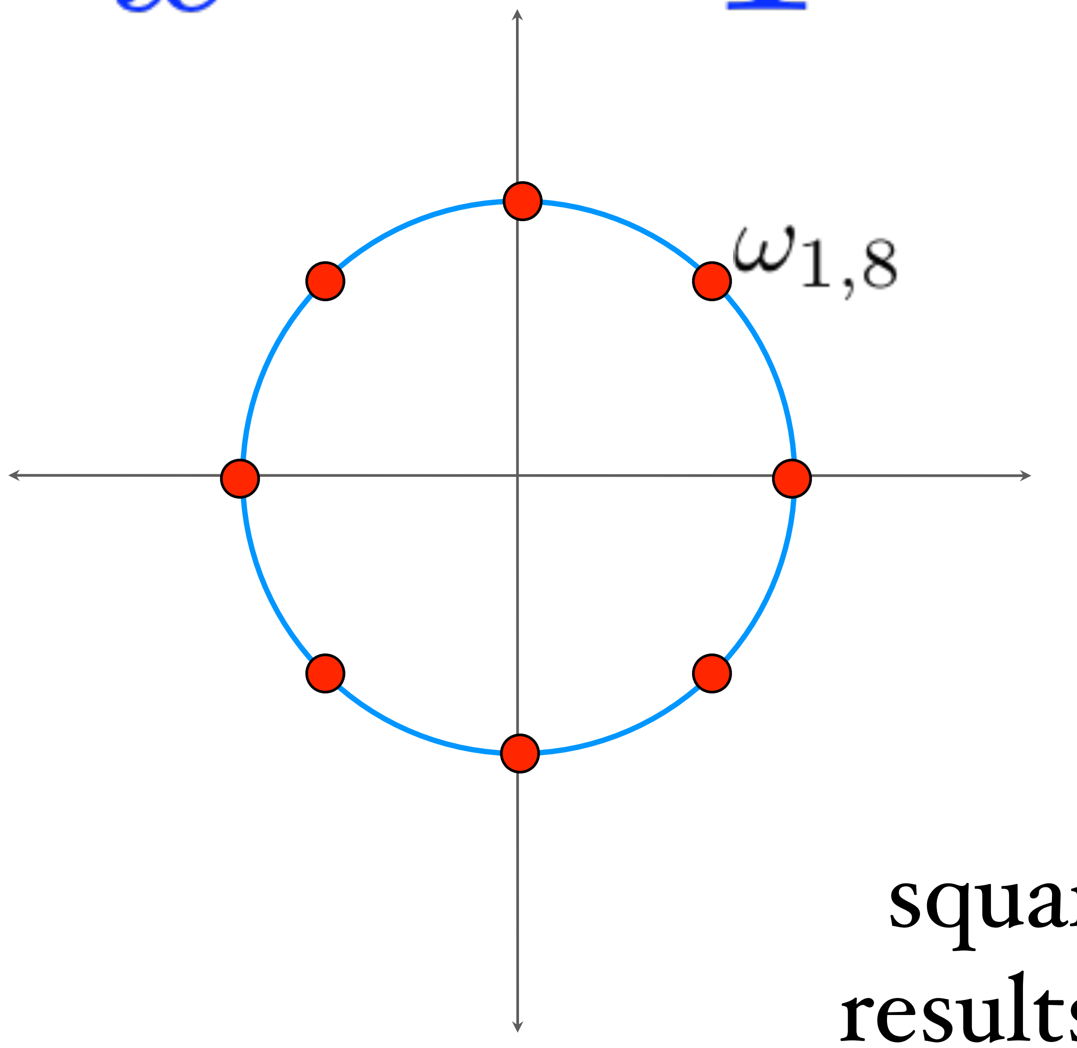
$$\omega_{1,8}^2 = \left(\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)^2 = \left(\frac{1}{\sqrt{2}}\right)^2 + 2\left(\frac{1}{\sqrt{2}}\frac{i}{\sqrt{2}}\right) + \left(\frac{i}{\sqrt{2}}\right)^2$$

$$= 1/2 + i - 1/2$$

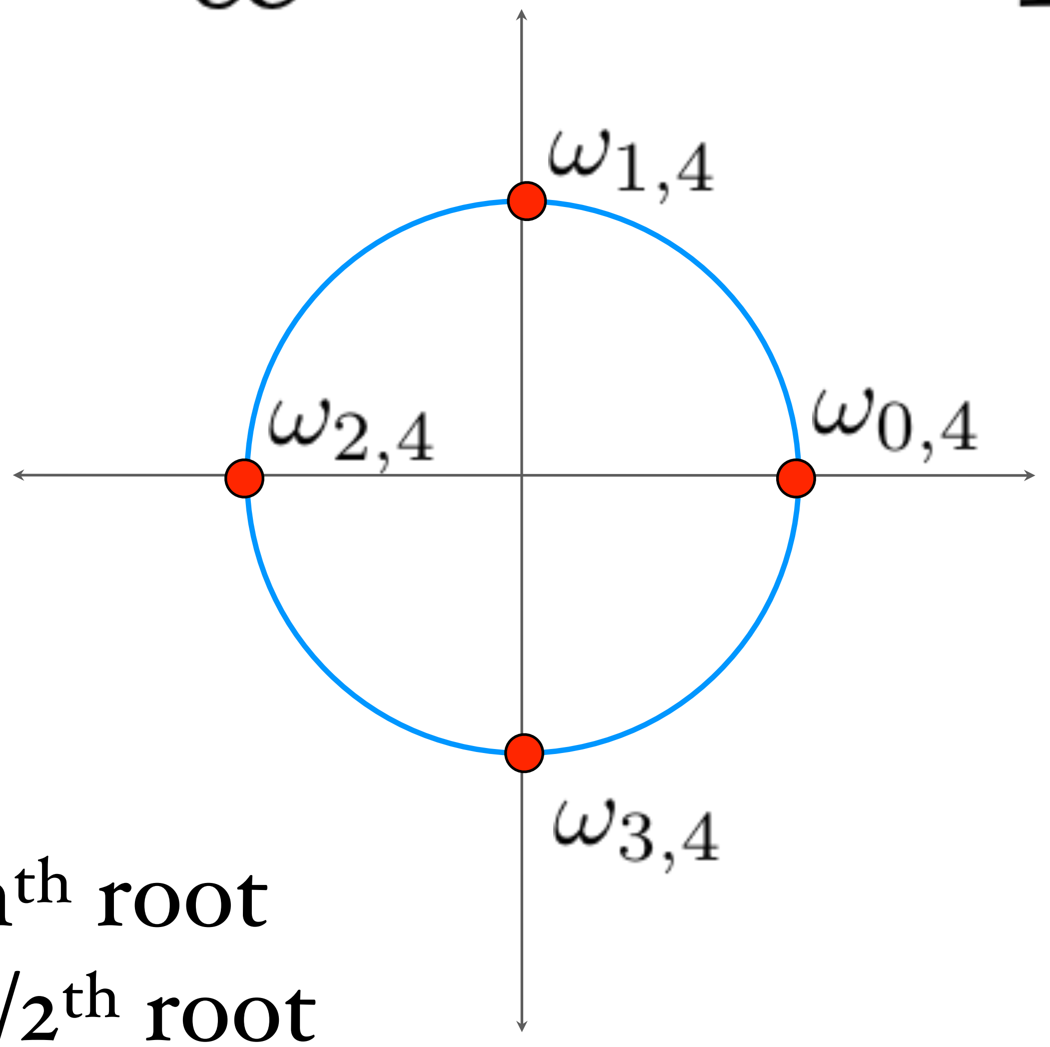
$$= i$$

# roots of unity

$x^n = 1$




$x^{n/2} = 1$




squaring an n$^{th}$ root
results in an n/2$^{th}$ root

$$A(x) = A_e(x^2) + xA_o(x^2)$$

evaluate at a root of unity

$$A(x) = A_e(x^2) + xA_o(x^2)$$

evaluate at a root of unity

$$A(\omega_{i,n}) = A_e(\omega_{i,n}^2) + \omega_{i,n}A_o(\omega_{i,n}^2)$$

$n^{th}$ root of unity

$n/2^{th}$ root of unity

$n/2^{th}$ root of unity

# FFT(f=a[1,...,n])

Evaluates degree n poly on the $n^{th}$ roots of unity

# FFT(f=a[1,...,n])

E[...] <- FFT($A_e$) // eval Ae on n/2 roots of unity

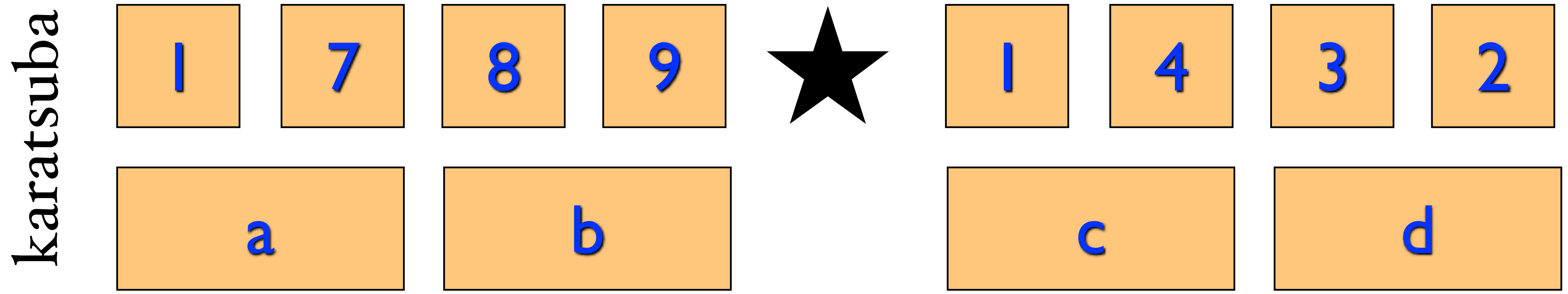
O[...] <- FFT($A_o$) // eval Ao on n/2 roots of unity

combine results using equation

$$A(\omega_{i,n}) = A_e(\omega_{i,n}^2) + \omega_{i,n} A_o(\omega_{i,n}^2)$$

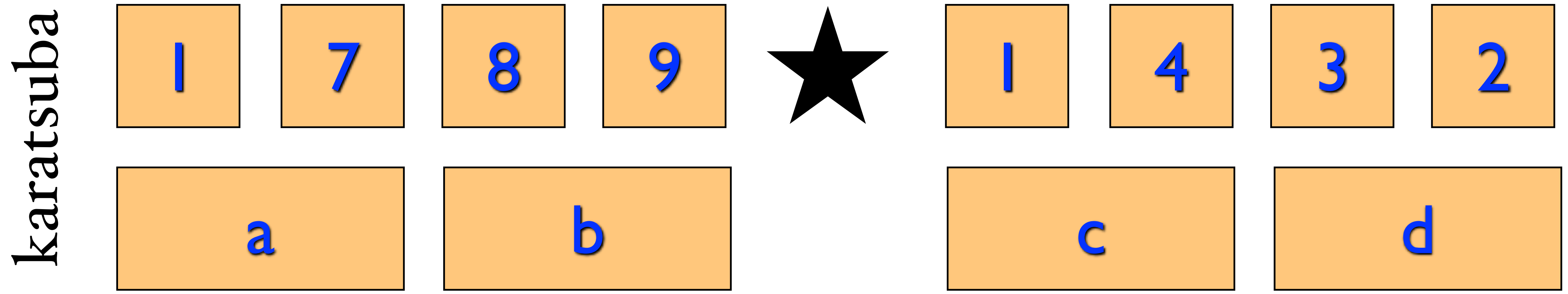
$$A(\omega_{i,n}) = A_e(\omega_{i \bmod n/2, \frac{n}{2}}) + \omega_{i,n} A_o(\omega_{i \bmod n/2, \frac{n}{2}})$$

# application to mult




$$\Theta(n^{\log_2 3})$$

# application to mult




$$T(n) = 3T(n/2) + 6O(n)$$

$$\Theta(n^{\log_2 3})$$

$a_3$ $a_2$ $a_1$ $a_0$ ★ $b_3$ $b_2$ $b_1$ $b_0$

$a_3$ $a_2$ $a_1$ $a_0$ ★ $b_3$ $b_2$ $b_1$ $b_0$

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots + 0x^7$$

$$B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + \cdots + 0x^7$$

| | | |
|---|---|---|
| $A(\omega_1)$ | $B(\omega_1)$ | $C(\omega_1)$ |
| $A(\omega_4)$ | $B(\omega_4)$ | $C(\omega_4)$ |
| $A(\omega_8)$ | $B(\omega_8)$ | $C(\omega_8)$ |

$$C(x) = A(x)B(x)$$

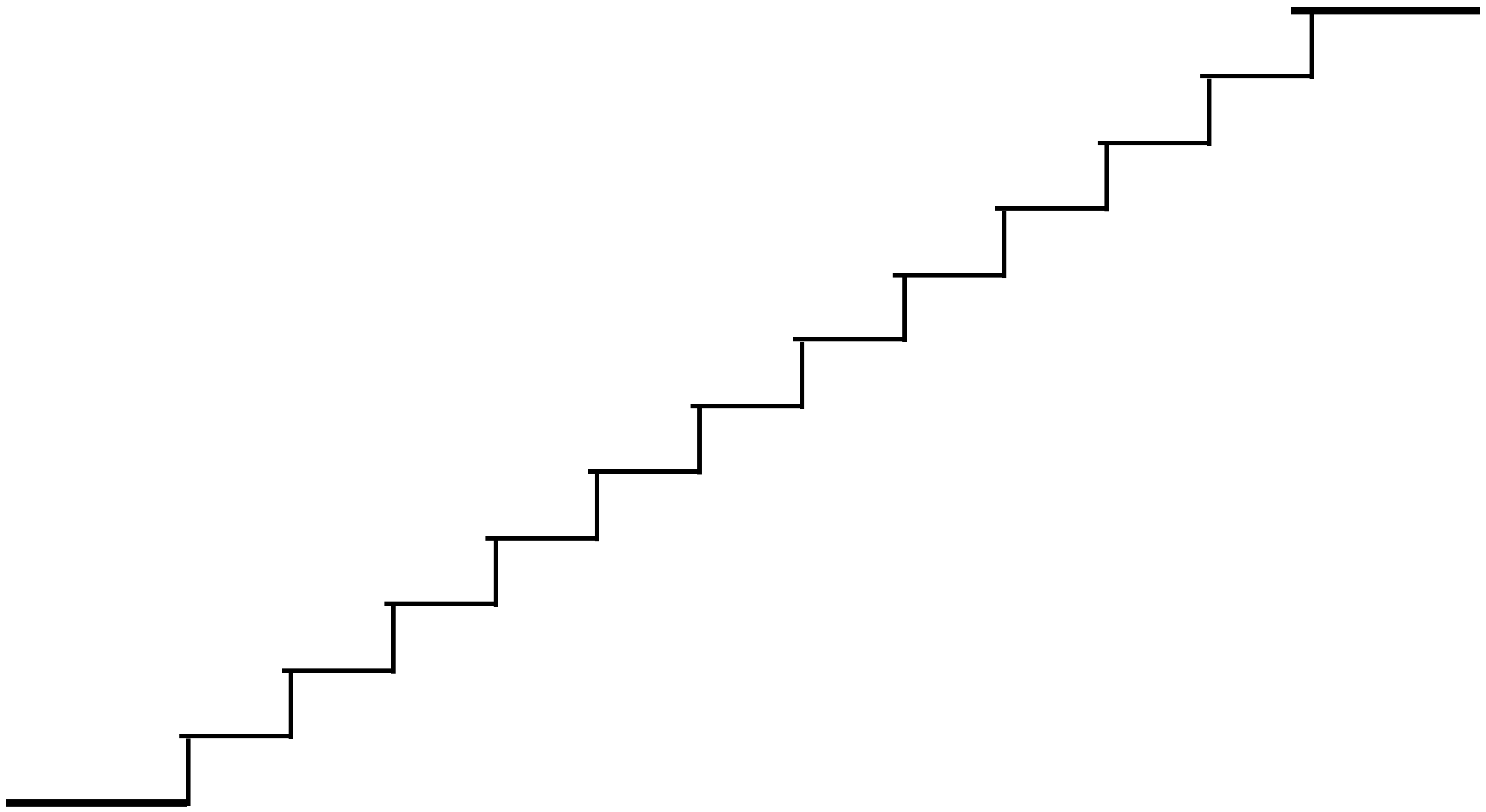
$a_3$ $a_2$ $a_1$ $a_0$ ★ $b_3$ $b_2$ $b_1$ $b_0$

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots + 0x^7$$

$$B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + \cdots + 0x^7$$

| | | |
|---|---|---|
| $A(\omega_1)$ | $B(\omega_1)$ | $C(\omega_1)$ |
| $A(\omega_8)$ | $B(\omega_8)$ | $C(\omega_8)$ |

$$C(x) = A(x)B(x)$$



```
Stairs(n)
    if n<=1 return 1
    return Stairs(n-1) + Stairs(n-2)
```

Stairs(5)

Stairs(4) Stairs(3)

Stairs(3) Stairs(2) Stairs(2) Stairs(1)

Stairs(2) Stairs(1) Stairs(1) Stairs(0) Stairs(1) Stairs(0)

```
    initialize memory M

Stairs(n)
    if n<=1 then return 1
    if n is in M, return M[n]
    answer = Stairs(i−1)+ Stairs(i−2)
    M[n] = answer
    return answer
```

```
Stairs(n)

    if n<=1 then return 1
    if n is in M, return M[n]
    answer = Stairs(i-1)+ Stairs(i-2)
    M[n] = answer
    return answer
```

Stairs(5)

Stairs(n)

```
stair[0]=1
stair[1]=1
for i=2 to n
    stair[i] = stair[i-1]+stair[i-2]
return stair[i]
```