OUR MAIN FOCUS IN THIS COURSE

1) RUNNING TIME
2) CORRECTNESS

OUR MAIN FOCUS IN THIS COURSE

1) RUNNING TIME
2) CORRECTNESS

Q: HOW TO PROVE THAT THE ALGORITM IS CORRECT?

OUR MAIN FOCUS IN THIS COURSE

1) RUNNING TIME
2) CORRECTNESS

Q: HOW TO PROVE THAT THE ALGORITM IS CORRECT?

A: FIND PATTERNS IN YOUR ALGORITM.

OUR MAIN FOCUS IN THIS COURSE

1) RUNNING TIME
2) CORRECTNESS

Q: HOW TO PROVE THAT THE ALGORITM IS CORRECT?

A: FIND PATTERNS IN YOUR ALGORITM.

- NON TRIVIAL
- NOVEL
- EXCITING

```
min ← A[1]
for i ← 2 to n
{
    if A[i] < min
        min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

```
min ← A[1]
for i ← 2 to n
{   if A[i] < min
        min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF A[1....i].

```
min ← A[1]
for i ← 2 to n
{
    if A[i] < min
        min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \ldots i]$.

PROOF: BY INDUCTION

```
min ← A[1]
for i ← 2 to n
{   if A[i] < min
        min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \ldots i]$.

PROOF: BY INDUCTION

1) $\square$

$i = 1$

BASE CASE

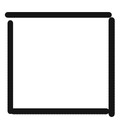
```
min ← A[1]
for i ← 2 to n
{
    if A[i] < min
        min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

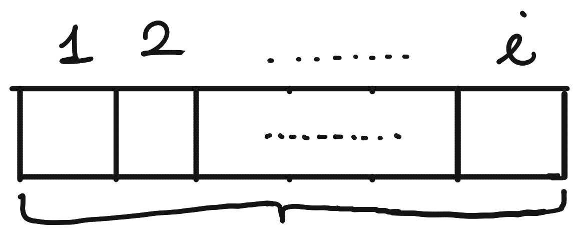
Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \ldots i]$.

PROOF: BY INDUCTION

1) 

$i = 1$

BASE CASE

2) 

ASSUME THAT THE STATEMENT IS TRUE FOR $[1 \ldots i]$

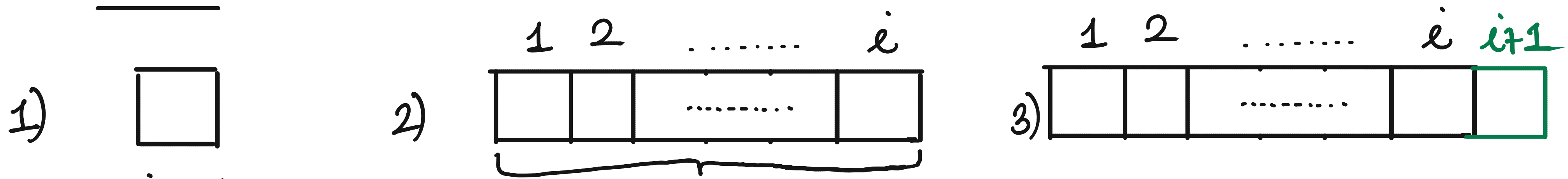
```
min ← A[1]
for  i ← 2 to n
{
     if  A[i] < min
         min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \dots i]$.

PROOF: BY INDUCTION



$i = 1$

BASE CASE

ASSUME THAT THE STATEMENT IS TRUE FOR $[1 \dots i]$ ⇒ PROVE THAT THE STATEMENT IS TRUE FOR $i+1$

```
min ← A[1]
for i ← 2 to n
{   if A[i] < min
        min ← A[i]
}
return min
```

Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \ldots i]$.

PROOF: BY INDUCTION

1) BASE CASE, $i = 1$. TRIVIALLY TRUE

```
min ← A[1]
for  i ← 2 to n
{
    if  A[i] < min
        min ← A[i]
}
return min
```

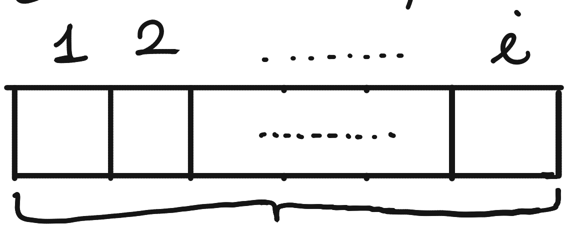
Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \ldots i]$.

PROOF: BY INDUCTION

1) BASE CASE, $i = 1$. TRIVIALLY TRUE

2) 

$\Rightarrow$ AFTER THE i^{th} ITERATION min CONTAINS MINIMUM OF $A[1 \ldots i]$

STATEMENT IS TRUE FOR $[1 \ldots i]$

```
min ← A[1]
for i ← 2 to n
{
    if A[i] < min
        min ← A[i]
}
return min
```

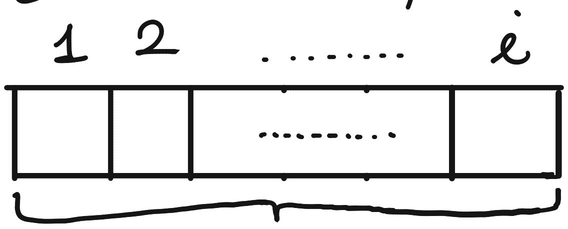
Q: PROVE THAT THIS ALGORITHM IS CORRECT

PATTERN OR PROPERTY OR FEATURE OR OBSERVATION

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF A[1....i].

PROOF: BY INDUCTION

1) BASE CASE, $i = 1$. TRIVIALLY TRUE

2) 

$\Rightarrow$ AFTER THE i^{th} ITERATION min CONTAINS MINIMUM OF A[1.... i]

STATEMENT IS TRUE FOR [1...i]

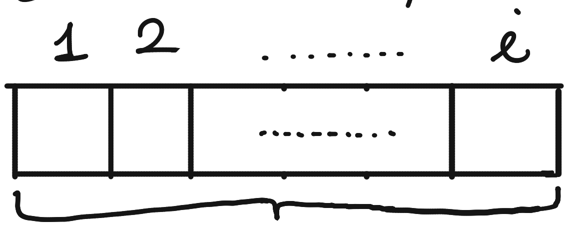
3) SHOW THAT AFTER THE $(i+1)^{th}$ ITERATION, min CONTAINS MINIMUM OF A[1.... i+1]

Lemma: After the i^{th} iteration, min contains the minimum of $A[1 \ldots i]$.

Proof: By induction

) Base case, $i = 1$. Trivially true

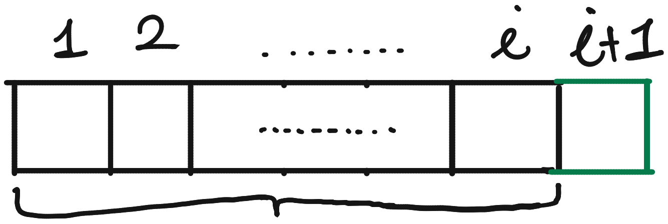
)



Statement is true for $[1 \ldots i]$

$\Rightarrow$ After the i^{th} iteration min contains minimum of $A[1 \ldots i]$

) Show that after the $(i+1)^{th}$ iteration, min contains minimum of $A[1 \ldots i+1]$



min_i

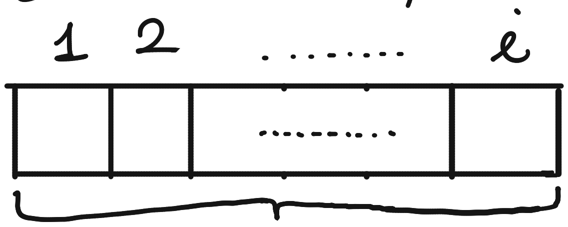
$$min_{i+1} = minimum\{ min_i , A[i+1] \}$$

Lemma: AFTER THE i^{th} ITERATION, min CONTAINS THE MINIMUM OF $A[1 \ldots i]$.

PROOF: BY INDUCTION

) BASE CASE, $i = 1$. TRIVIALLY TRUE

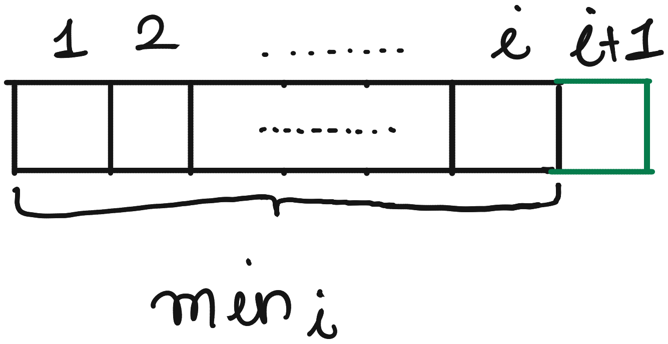
)



STATEMENT IS TRUE FOR $[1 \ldots i]$

$\Rightarrow$ AFTER THE i^{th} ITERATION min CONTAINS MINIMUM OF $A[1 \ldots i]$

) SHOW THAT AFTER THE $(i+1)^{th}$ ITERATION, min CONTAINS MINIMUM OF $A[1 \ldots i+1]$



$$min_{i+1} = \text{minimum}\{min_i, A[i+1]\}$$

$\Downarrow$ BY INDUCTION HYPOTHESIS

$$= \text{minimum}\{\text{minimum}\{A[1], A[2], \ldots, A[i]\}, A[i+1]\}$$

$$= \text{minimum}\{A[1], A[2], \ldots, A[i+1]\}$$

SORTING

10 2 8 7 6 1 3

Sorting

$\boxed{10}$ 2 8 7 6 1 3

SORTING

2

$\boxed{10}$ 8 7 6 1 3

SORTING

2

$\boxed{10}$ 8 7 6 1 3

Sorting

2	10

8 7 6 1 3

SORTING

8

2	10

7 6 1 3

SORTING

8

$\boxed{2}$ $\boxed{10}$ 7 6 1 3

SORTING

$\boxed{2}\,\boxed{8}\,\boxed{10}$ 7 6 1 3

Sorting

$\boxed{2}\boxed{8}\boxed{10}$ 7 6 1 3

$\boxed{2}\boxed{7}\boxed{8}\boxed{10}$ 6 1 3

SORTING

$\boxed{2}\boxed{8}\boxed{10}$ 7 6 1 3

$\boxed{2}\boxed{7}\boxed{8}\boxed{10}$ 6 1 3

$\boxed{2}\boxed{6}\boxed{7}\boxed{8}\boxed{10}$ 1 3

$\boxed{1}\boxed{2}\boxed{6}\boxed{7}\boxed{8}\boxed{10}$ 3

$\boxed{1}\boxed{2}\boxed{3}\boxed{6}\boxed{7}\boxed{8}\boxed{10}$

INSERTION SORT(A[1....n])
{

INSERTION SORT(A[1....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {   IF   A[j] < A[j-1]
                 SWAP ( A[j], A[j-1])

          ELSE
                 BREAK ;
      }
}
```

CORRECTNESS

INSERTION SORT (A[1....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {    IF   A[j] < A[j-1]
                  SWAP ( A[j], A[j-1])
           ELSE
                  BREAK ;
      }
}
```

CORRECTNESS

PATTERN OR INVARIANT : AFTER THE i^{th} ITERATION, A[1.... i] IS SORTED.

INSERTION SORT(A[1....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {   IF  A[j] < A[j-1]
                SWAP ( A[j], A[j-1])

          ELSE
                BREAK;
      }
}
```

CORRECTNESS

PATTERN OR INVARIANT: AFTER THE i^{th} ITERATION, A[1....i] IS SORTED.

PROOF: 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

INSERTION SORT (A[1.....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {     IF   A[j] < A[j-1]
                  SWAP ( A[j], A[j-1])

            ELSE
                  BREAK ;
      }
}
```

CORRECTNESS

<u>PATTERN OR INVARIANT</u> : AFTER THE i^{th} ITERATION, A[1.....i] IS SORTED.

<u>PROOF</u> : 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS :

1	2	....	$i-1$	i
a_1	a_2	....	a_{i-1}	a_i

SORTED

```
INSERTION SORT (A[1....n])
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {     IF   A[j] < A[j-1]
                    SWAP ( A[j], A[j-1])

            ELSE
                    BREAK ;
      }
}
```

CORRECTNESS

PATTERN OR INVARIANT : AFTER THE i^{th} ITERATION, $A[1 \ldots i]$ IS SORTED.

PROOF : 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS :

1	2	....	$i-1$	i
a_1	a_2	....	a_{i-1}	a_i

SORTED

$$a_1 < a_2 < \ldots\ldots < a_{i-1} < a_i$$

PATTERN OR INVARIANT : AFTER THE i^{th} ITERATION, $A[1 \ldots i]$ IS SORTED.

PROOF : 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS :

1	2	………	$i-1$	i
a_1	a_2	……..	a_{i-1}	a_i

SORTED

$$a_1 < a_2 < \ldots\ldots < a_{i-1} < a_i$$

(3)

1	2	………	$i-1$	i	$i+1$
a_1	a_2	……..	a_{i-1}	a_i	x

AFTER i^{th} ITERATION

1	2	….	j	….	$i-1$	i	$i+1$
a_1	a_2	….	x	…	a_{i-2}	a_{i-1}	a_i

PATTERN OR INVARIANT : AFTER THE i^{th} ITERATION, $A[1....i]$ IS SORTED.

PROOF : 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS :

1	2	....	$i-1$	i
a_1	a_2	....	a_{i-1}	a_i

SORTED

$$a_1 < a_2 < \dots\dots < a_{i-1} < a_i$$

(3)

1	2	....	$i-1$	i	$i+1$
a_1	a_2	....	a_{i-1}	a_i	x

AFTER i^{th} ITERATION

1	2	....	j	....	$i-1$	i	$i+1$
a_1	a_2	....	x	...	a_{i-2}	a_{i-1}	a_i

$$a_1 < a_2 < \dots < a_{j-1} \ ? \ x \ ? \ a_j < a_{j+1} < \dots < a_i$$

PATTERN OR INVARIANT: AFTER THE i^{th} ITERATION, $A[1....i]$ IS SORTED.

PROOF: 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS:

1	2	....	i-1	i
a_1	a_2	....	a_{i-1}	a_i

SORTED

$$a_1 < a_2 < \ldots\ldots < a_{i-1} < a_i$$

(3)

1	2	....	i-1	i	i+1
a_1	a_2	....	a_{i-1}	a_i	x

AFTER i^{th} ITERATION

1	2	....	j	....	i-1	i	i+1
a_1	a_2	....	x	...	a_{i-2}	a_{i-1}	a_i

$x<$ $x<$ $x<$

$$a_1 < a_2 < \ldots < a_{j-1}\ ?\ x\ ?\ a_j < a_{j+1} < \ldots < a_i$$

PATTERN OR INVARIANT : AFTER THE i^{th} ITERATION, $A[1 \ldots i]$ IS SORTED.

PROOF : 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS :

1	2	....	$i-1$	i
a_1	a_2	....	a_{i-1}	a_i

$\underbrace{\hspace{6em}}_{\text{SORTED}}$

$$a_1 < a_2 < \ldots\ldots < a_{i-1} < a_i$$

(3)

1	2	....	$i-1$	i	$i+1$
a_1	a_2	....	a_{i-1}	a_i	x

AFTER i^{th} ITERATION

1	2	.....	j	.....	$i-1$	i	$i+1$
a_1	a_2	....	x	...	a_{i-2}	a_{i-1}	a_i

$$a_1 < a_2 < \ldots < a_{j-1} \; ? \; x < \overset{x<}{a_j} < \overset{x<}{a_{j+1}} < \ldots < \overset{x<}{a_i}$$

PATTERN OR INVARIANT : AFTER THE i^{th} ITERATION, $A[1 \ldots i]$ IS SORTED.

PROOF : 1) BASE CASE, $i = 1$ TRIVIALLY TRUE.

2) INDUCTION HYPOTHESIS :

$$\underbrace{\begin{array}{|c|c|c|c|c|} \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2} & \multicolumn{1}{c}{\cdots\cdots} & \multicolumn{1}{c}{i-1} & \multicolumn{1}{c}{i} \\ \hline a_1 & a_2 & \cdots\cdots & a_{i-1} & a_i \\ \hline \end{array}}_{\text{SORTED}}$$

$$a_1 < a_2 < \ldots\ldots < a_{i-1} < a_i$$

(3)

$$\begin{array}{|c|c|c|c|c|c} \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2} & \multicolumn{1}{c}{\cdots\cdots} & \multicolumn{1}{c}{i-1} & \multicolumn{1}{c}{i} & i+1 \\ \hline a_1 & a_2 & \cdots\cdots & a_{i-1} & a_i & x \\ \hline \end{array}$$

AFTER i^{th} ITERATION

$$\begin{array}{|c|c|c|c|c|c|c|c|} \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2} & \multicolumn{1}{c}{\cdots} & \multicolumn{1}{c}{j} & \multicolumn{1}{c}{\cdots} & \multicolumn{1}{c}{i-1} & \multicolumn{1}{c}{i} & \multicolumn{1}{c}{i+1} \\ \hline a_1 & a_2 & \cdots & x & \cdots & a_{i-2} & a_{i-1} & a_i \\ \hline \end{array}$$

$$a_1 < a_2 < \ldots < a_{j-1} < x < \overset{x<}{a_j} < \overset{x<}{a_{j+1}} < \ldots < \overset{x<}{a_i}$$

SINCE OUR ALGORITHM STOPPED AT $j \Rightarrow a_{j-1} < x$.

INSERTION SORT(A[1....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {    IF   A[j] < A[j-1]
                  SWAP ( A[j], A[j-1])

           ELSE
                  BREAK;
      }
}
```

RUNNING TIME

```
INSERTION SORT (A[1.....n])
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {    IF  A[j] < A[j-1]
                 SWAP ( A[j], A[j-1])
           ELSE
                 BREAK ;
      }
}
```

c

RUNNING TIME $= \sum_{i=2}^{n} \sum_{j=i}^{2} c$

INSERTION SORT (A[1....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {    IF  A[j] < A[j-1]
                SWAP ( A[j], A[j-1])
           ELSE
                BREAK;
      }
}
```

c

RUNNING TIME $= \sum_{i=2}^{n} \sum_{j=i}^{2} c$

$= \sum_{i=2}^{n} (i-1)c$

$= \frac{(n-1) \cdot n}{2} \cdot c$

$= O(n^2)$

INSERTION SORT (A[1.....n])

```
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {    IF   A[j] < A[j-1]                 }
                 SWAP ( A[j], A[j-1])         }  c
           ELSE                               }
                 BREAK ;                      }
      }
}
```

$$\text{RUNNING TIME} = \sum_{i=2}^{n} \sum_{j=i}^{2} c$$

$$= \sum_{i=2}^{n} (i-1)c$$

$$= \frac{(n-1).n}{2} . c$$

$$= O(n^2)$$

WORST CASE INPUT

```
INSERTION SORT (A[1....n])
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {     IF  A[j] < A[j-1]
                  SWAP ( A[j], A[j-1])                  c
            ELSE
                  BREAK ;
      }
}
```

RUNNING TIME $= \sum_{i=2}^{n} \sum_{j=i}^{2} c$

$= \sum_{i=2}^{n} (i-1) c$

$= \frac{(n-1) \cdot n}{2} \cdot c$

$= O(n^2)$

WORST CASE INPUT

$n \quad n-1 \quad n-2 \quad \cdots\cdots \quad 3 \quad 2 \quad 1$

Best Case Input

BEST CASE INPUT

1 2 3 . . . . . . n $n-1$

RUNNING TIME $= O(n)$

BEST CASE INPUT

1 2 3 ...... n $n-1$

RUNNING TIME $= O(n)$

THE BEST CASE RUNNING TIME IS VERY FAST AND WORST CASE IS VERY BAD.

Q: DOES INSERTION SORT GOOD OR BAD ON OTHER INPUTS?

BEST CASE INPUT

1 2 3 ...... n n-1

RUNNING TIME $= O(n)$

THE BEST CASE RUNNING TIME IS VERY FAST AND WORST CASE IS VERY BAD.

Q: DOES INSERTION SORT GOOD OR BAD ON OTHER INPUTS?

$\Downarrow$

Q: ON AVERAGE, WHAT IS THE RUNNING TIME OF INSERTION SORT

BEST CASE INPUT

1 2 3 ...... n n-1

RUNNING TIME = $O(n)$

THE BEST CASE RUNNING TIME IS VERY FAST AND WORST CASE IS VERY BAD.

Q: DOES INSERTION SORT GOOD OR BAD ON OTHER INPUTS?

$\Downarrow$

Q: ON AVERAGE, WHAT IS THE RUNNING TIME OF INSERTION SORT

$n=3$	RUNNING TIME $i=2$	RUNNING TIME $i=3$	TOTAL RUNNING TIME
1 2 3	c	c	2c
1 3 2	c	2c	3c
2 1 3	2c	c	3.c
2 3 1	c	3c	4c
3 1 2	2c	2c	4c
3 2 1	2c	3c	5c

BEST CASE INPUT

1 2 3 ...... n n-1

RUNNING TIME = $O(n)$

THE BEST CASE RUNNING TIME IS VERY FAST AND WORST CASE IS VERY BAD.

Q: DOES INSERTION SORT GOOD OR BAD ON OTHER INPUTS?

⇓

Q: ON AVERAGE, WHAT IS THE RUNNING TIME OF INSERTION SORT

$n=3$	RUNNING TIME ITERATION 2	 ITERATION 3	TOTAL RUNNING TIME
123	c	c	2c
132	c	2c	3c
213	2c	c	3.c
231	c	3c	4c
312	2c	2c	4c
321	2c	3c	5c
			+ ______
			21c

AVERAGE RUNNING TIME = $\frac{21c}{6}$

BEST CASE INPUT

1 2 3 ...... n n-1

RUNNING TIME $= O(n)$

THE BEST CASE RUNNING TIME IS VERY FAST AND WORST CASE IS VERY BAD.

Q: DOES INSERTION SORT GOOD OR BAD ON OTHER INPUTS?

$\Downarrow$

Q: ON AVERAGE, WHAT IS THE RUNNING TIME OF INSERTION SORT

$n=3$	RUNNING TIME ITERATION 2	RUNNING TIME ITERATION 3	TOTAL RUNNING TIME
1 2 3	c	c	2c
1 3 2	c	2c	3 c
2 1 3	2C	c	3.c
2 3 1	c	3c	4 c
3 1 2	2c	2c	4 c
3 2 1	2c	3c	5 c

AVERAGE RUNNING TIME $= \frac{9c}{6} + \frac{12c}{6}$

$+$ ______

$\frac{21c}{6}$

BEST CASE INPUT

1 2 3 ...... n n-1

RUNNING TIME = $O(n)$

THE BEST CASE RUNNING TIME IS VERY FAST AND WORST CASE IS VERY BAD.

Q: DOES INSERTION SORT GOOD OR BAD ON OTHER INPUTS?

⇓

Q: ON AVERAGE, WHAT IS THE RUNNING TIME OF INSERTION SORT

$n=3$

	RUNNING TIME		TOTAL RUNNING TIME
	ITERATION 2	ITERATION 3	
1 2 3	c	c	2c
1 3 2	c	2c	3c
2 1 3	c	c	2c
2 3 1	c	2c	3c
3 1 2	c	2c	3c
3 2 1	c	2c	3c

$$\text{AVERAGE RUNNING TIME} = \frac{9c}{6} + \frac{12c}{6}$$

AVERAGE RUNNING TIME AT ITERATION 2

$$\frac{21c}{6}$$

$$\text{AVERAGE RUNNING TIME} = \sum_{i=2}^{n} \text{AVERAGE RUNNING TIME IN ITERATION } i$$

AVERAGE RUNNING TIME IN ITERATION 2

$$\text{AVERAGE RUNNING TIME} = \sum_{i=2}^{n} \text{AVERAGE RUNNING TIME IN ITERATION } i$$

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \;\ldots\; a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2 a_3 \ldots\ldots a_n$	c
CASE 2 : $a_2 a_1 a_3 \ldots\ldots a_n$	$2c$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \; \dots \; a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2 a_3 \dots\dots a_n$	c
CASE 2 : $a_2 a_1 a_3 \dots\dots a_n$	$2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

e.g. $1\; 2\; 3\; 4 \;\dots\dots\; n\text{-}1\; n$

$3\; 8\; 1\; n \;\dots\; 7\; 4$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\ a_2\ a_3\ \ldots\ a_n$

AFTER 2^{nd} ITERATION RUNNING TIME

CASE 1 : $a_1 a_2\ a_3\ \ldots\ a_n$ $\quad c$

CASE 2 : $a_2 a_1\ a_3\ \ldots\ a_n$ $\quad 2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

e.g. $1\ 2\ 3\ 4\ \ldots\ n\text{-}1\ n$

$3\ 8\ 1\ n\ \ldots\ 7\ 4$

1) CHOOSE ANY TWO NUMBERS FROM n NUMBERS

$a < b$ $\quad$ # ways $= \binom{n}{2}$

2)

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \,\dots\, a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2 a_3 \dots\dots a_n$	c
CASE 2 : $a_2 a_1 a_3 \dots\dots a_n$	$2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

e.g. $1\ 2\ 3\ 4 \dots\dots n\text{-}1\ n$

$3\ 8\ 1\ n \dots\dots 7\ 4$

1) CHOOSE ANY TWO NUMBERS FROM n NUMBERS

$a < b$ $\qquad$ # ways $= \binom{n}{2}$

2) $\underbrace{a\ b}_{\text{FIXED}}$ $\boxed{\text{REST } n\text{-}2 \text{ NUMBERS}}$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \ldots a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2 a_3 \ldots\ldots a_n$	c
CASE 2 : $a_2 a_1 a_3 \ldots\ldots a_n$	$2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

e.g. $1\ 2\ 3\ 4 \ldots\ldots n\text{-}1\ n$

$3\ 8\ 1\ n \ldots\ 7\ 4$

1) CHOOSE ANY TWO NUMBERS FROM n NUMBERS

$a < b$ $\qquad$ # ways $= \binom{n}{2}$

2) $\underbrace{a\ b}_{\text{FIXED}}$ $\boxed{\text{REST } n\text{-}2 \text{ NUMBERS}}$

3) IN HOW MANY WAYS CAN YOU ARRANGE REST $n-2$ NUMBERS $\Rightarrow$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \ldots\ldots a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2\, a_3 \ldots\ldots a_n$	c
CASE 2 : $a_2 a_1\, a_3 \ldots\ldots a_n$	$2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

e.g. $1\ 2\ 3\ 4 \ldots\ldots n\text{-}1\ n$

$3\ 8\ 1\ n \ldots\ldots 7\ 4$

1) CHOOSE ANY TWO NUMBERS FROM n NUMBERS

$a < b$ # ways $= \binom{n}{2}$

2) $\underbrace{a\ b}_{\text{FIXED}}$ $\boxed{\text{REST } n\text{-}2 \text{ NUMBERS}}$

3) IN HOW MANY WAYS CAN YOU ARRANGE REST $n-2$ NUMBERS $\Rightarrow (n-2)!$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3\, \ldots\, a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2\, a_3 \ldots\ldots a_n$	c
CASE 2 : $a_2 a_1\, a_3 \ldots\ldots a_n$	$2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

A $\binom{n}{2}(n-2)!$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \,\ldots\, a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1 : $a_1 a_2 a_3 \ldots\ldots a_n$	c
CASE 2 : $a_2 a_1 a_3 \ldots\ldots a_n$	$2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

A $\binom{n}{2}(n-2)!$

Q: IN HOW MANY PERMUTATIONS, THERE IS A SWAP IN ITERATION 2.

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \,\ldots\, a_n$

AFTER 2^{nd} ITERATION — RUNNING TIME

CASE 1 : $a_1 a_2 a_3 \ldots\ldots a_n$ — c

CASE 2 : $a_2 a_1 a_3 \ldots\ldots a_n$ — $2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

A $\binom{n}{2}(n-2)!$

Q: IN HOW MANY PERMUTATIONS, THERE IS A SWAP IN ITERATION 2.

A SAME ANALYSIS, $\binom{n}{2}(n-2)!$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \ldots a_n$

AFTER 2^{nd} ITERATION — RUNNING TIME

CASE 1 : $a_1 a_2 a_3 \ldots a_n$ — c

CASE 2 : $a_2 a_1 a_3 \ldots a_n$ — $2c$

Q: IN HOW MANY PERMUTATIONS, THERE IS NO SWAP IN ITERATION 2.

A $\binom{n}{2}(n-2)! = \dfrac{n(n-1)\cdot(n-2)!}{2\cdot 1} = \dfrac{n!}{2}$

Q: IN HOW MANY PERMUTATIONS, THERE IS A SWAP IN ITERATION 2.

A SAME ANALYSIS, $\binom{n}{2}(n-2)! = \dfrac{n!}{2}$

$$\text{AVERAGE RUNNING TIME} = \sum_{i=2}^{n} \text{AVERAGE RUNNING TIME IN ITERATION } i$$

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \,\ldots\, a_n$

AFTER 2^{nd} ITERATION — RUNNING TIME

CASE 1 : $a_1 a_2 a_3 \ldots\ldots a_n$ — c

CASE 2 : $a_2 a_1 a_3 \ldots\ldots a_n$ — $2c$

AVERAGE RUNNING TIME IN ITERATION 2

$$= \frac{\left(\begin{array}{c}\text{\# PERMUTATION OF}\\ \text{CASE 1}\end{array}\right) \times c + \left(\begin{array}{c}\text{\# PERMUTATIONS OF}\\ \text{CASE 2}\end{array}\right) \times 2c}{n!}$$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 2

PERMUTATION : $a_1\, a_2\, a_3 \ldots\ldots a_n$

AFTER 2^{nd} ITERATION	RUNNING TIME
CASE 1: $a_1 a_2 a_3 \ldots\ldots a_n$	c
CASE 2: $a_2 a_1 a_3 \ldots\ldots a_n$	$2c$

AVERAGE RUNNING TIME IN ITERATION 2

$$= \frac{\left(\#\text{ PERMUTATION OF CASE 1}\right) \times c + \left(\#\text{ PERMUTATIONS OF CASE 2}\right) \times 2c}{n!}$$

$$= \frac{\left(\frac{n!}{2}\right) \times c + \left(\frac{n!}{2}\right) \times 2c}{n!}$$

AVERAGE RUNNING TIME IN ITERATION 3

$$= \frac{\left(\frac{n!}{3}\right) \times c + \left(\frac{n!}{3}\right) \times 2c + \left(\frac{n!}{3}\right) 3c}{n!}$$

AVERAGE RUNNING TIME IN ITERATION 3

$$= \frac{\left(\frac{n!}{3}\right) \times c + \left(\frac{n!}{3}\right) \times 2c + \left(\frac{n!}{3}\right) 3c}{n!}$$

AVERAGE RUNNING TIME IN ITERATION i

$$= \frac{\left(\frac{n!}{i}\right) \times c + \left(\frac{n!}{i}\right) \times 2c + \cdots + \left(\frac{n!}{i}\right) i}{n!}$$

AVERAGE RUNNING TIME IN ITERATION 3

$$= \frac{\left(\frac{n!}{3}\right) \times c + \left(\frac{n!}{3}\right) \times 2c + \left(\frac{n!}{3}\right) 3c}{n!}$$

AVERAGE RUNNING TIME IN ITERATION i

$$= \frac{\left(\frac{n!}{i}\right) \times c + \left(\frac{n!}{i}\right) \times 2c + \cdots + \left(\frac{n!}{i}\right) i}{n!}$$

$$= \frac{c}{i}(1 + 2 + \ldots + i)$$

$$= \frac{c}{i} \frac{i(i+1)}{2}$$

$$= \frac{c(i+1)}{2}$$

AVERAGE RUNNING TIME IN ITERATION 3

$$= \frac{\left(\frac{n!}{3}\right) \times c + \left(\frac{n!}{3}\right) \times 2c + \left(\frac{n!}{3}\right) 3c}{n!}$$

AVERAGE RUNNING TIME IN ITERATION i

$$= \frac{\left(\frac{n!}{i}\right) \times c + \left(\frac{n!}{i}\right) \times 2c + \cdots + \left(\frac{n!}{i}\right) i}{n!}$$

$$= \frac{c}{i} (1 + 2 + \ldots + i)$$

$$= \frac{c}{i} \frac{i(i+1)}{2}$$

$$= \frac{c(i+1)}{2}$$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

AVERAGE RUNNING TIME IN ITERATION 3

$$= \frac{\left(\frac{n!}{3}\right) \times c + \left(\frac{n!}{3}\right) \times 2c + \left(\frac{n!}{3}\right) 3c}{n!}$$

AVERAGE RUNNING TIME IN ITERATION i

$$= \frac{\left(\frac{n!}{i}\right) \times c + \left(\frac{n!}{i}\right) \times 2c + \cdots + \left(\frac{n!}{i}\right) i}{n!}$$

$$= \frac{c}{i}(1 + 2 + \ldots + i)$$

$$= \frac{c}{i} \frac{i(i+1)}{2}$$

$$= \frac{c(i+1)}{2}$$

AVERAGE RUNNING TIME $= \sum_{i=2}^{n}$ AVERAGE RUNNING TIME IN ITERATION i

$$= \sum_{i=2}^{n} c \frac{(i+1)}{2}$$

$$\leq c \frac{(n+1)(n+2)}{4}$$

$$= O(n^2)$$

When Should We use Insertion Sort

WHEN SHOULD WE USE INSERTION SORT

1) THE INPUT SIZE IS SMALL
2) THE INPUT IS NEARLY SORTED.

PROBLEM 1: IF $i < j$ & $A[i] > A[j]$, THEN PAIR (i, j) IS CALLED AN INVERSION.

1 3 8 5 4

PROBLEM 1: IF $i < j$ & $A[i] > A[j]$, THEN PAIR (i, j) IS CALLED AN INVERSION.

1 3 8 5 4

1 2

\# INVERSIONS = 3

PROBLEM 1: IF $i<j$ & $A[i] > A[j]$, THEN PAIR (i,j) IS CALLED AN INVERSION.

1 3 8 5 4

1 2

INVERSIONS = 3

Q: FIND THE PERMUTATION WITH MAXIMUM NUMBER OF INVERSIONS

PROBLEM 1 : IF $i < j$ & $A[i] > A[j]$, THEN PAIR (i, j) IS CALLED AN INVERSION.

$$1 \quad 3 \quad 8 \quad 5 \quad 4$$

$$1 \quad 2$$

INVERSIONS = 3

Q: FIND THE PERMUTATION WITH MAXIMUM NUMBER OF INVERSIONS

$$n \quad n-1 \quad \ldots\ldots \quad 1$$

INVERSIONS =

PROBLEM 1: IF $i < j$ & $A[i] > A[j]$, THEN PAIR (i, j) IS CALLED AN INVERSION.

1 3 8 5 4

1 2

INVERSIONS = 3

Q: FIND THE PERMUTATION WITH MAXIMUM NUMBER OF INVERSIONS

$n \quad n-1 \quad \ldots\ldots \quad 1$

$$\# \text{ INVERSIONS} = \frac{n(n-1)}{2}$$

PROBLEM 2: PROVE THAT THE RUNNING TIME OF INSERTION SORT ON A PERMUTATION P IS $O(n+I)$ WHERE I IS THE NUMBER OF PERMUTATION P

PROBLEM 2: PROVE THAT THE RUNNING TIME OF INSERTION SORT ON A PERMUTATION P IS $O(n + I)$ WHERE I IS THE NUMBER OF PERMUTATION P.

```
INSERTION SORT (A[1....n])
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {   IF  A[j] < A[j-1]
                SWAP ( A[j], A[j-1])

          ELSE
                BREAK ;
      }
}
```

PROBLEM 2: PROVE THAT THE RUNNING TIME OF INSERTION SORT ON A PERMUTATION P IS $O(n + I)$ WHERE I IS THE NUMBER OF PERMUTATION P.

```
INSERTION SORT (A[1....n])
{
    FOR i <- 2 to n
      FOR j <- i to 2
      {    IF  A[j] < A[j-1]          <- An Inversion Exists
               SWAP ( A[j], A[j-1])
                                      -> • FIXES INVERSION
           ELSE                          • DOES NOT CREATE
               BREAK ;                     NEW INVERSION
      }
}
```

PROBLEM 2: PROVE THAT THE RUNNING TIME OF INSERTION SORT ON A PERMUTATION P IS $O(n + I)$ WHERE I IS THE NUMBER OF PERMUTATION P.

```
INSERTION SORT (A[1....n])
{
    FOR i ← 2 to n
      FOR j ← i to 2
      {     IF A[j] < A[j-1]        ← AN INVERSION EXISTS
                SWAP ( A[j], A[j-1])
                                    ↳ • FIXES ONE INVERSION
            ELSE                      • DOES NOT CREATE
                BREAK ;                 NEW INVERSION
      }
}
```

→ NO INVERSION AT THE END SINCE THE ARRAY IS SORTED

PROBLEM 2 : PROVE THAT THE RUNNING TIME OF INSERTION SORT ON A PERMUTATION P IS $O(n+I)$ WHERE I IS THE NUMBER OF PERMUTATION P.

```
INSERTION SORT (A[1....n])
{
    FOR i <- 2 to n
      FOR j <- i to 2
      {     IF   A[j] < A[j-1]
                 SWAP ( A[j], A[j-1])
            ELSE
                 BREAK ;
      }
}
```

AN INVERSION EXISTS

↳ • FIXES ONE INVERSION
• DOES NOT CREATE NEW INVERSION

→ NO INVERSION AT THE END SINCE THE ARRAY IS SORTED

$\Rightarrow$ RUNNING TIME $= O(n+I)$.

PROBLEM 3: ASSUME THAT YOU ARE GIVEN AN ARRAY IN WHICH EACH ELEMENT IS K AWAY FROM ITS PROPER POSITION. SHOW THAT INSERTION SORT TAKES O(nk) TIME TO SORT SUCH AN ARRAY.

4 5 6 1 2 3 8

1 2 3 4 5 6 8

PROBLEM 3: ASSUME THAT YOU ARE GIVEN AN ARRAY IN WHICH EACH ELEMENT IS K AWAY FROM ITS PROPER POSITION. SHOW THAT INSERTION SORT TAKES O(nk) TIME TO SORT SUCH AN ARRAY.

4 5 6 1 2 3 8

1 2 3 4 5 6 8

A i $i+7k$

CAN $A[i] > A[i+7k]$?

A $\quad i \quad$ $i+7K$

CAN $A[i] > A[i+7K]$?

$i-6K$ $\quad$ $i+8K$

A $\quad i \quad$ $i+7K$

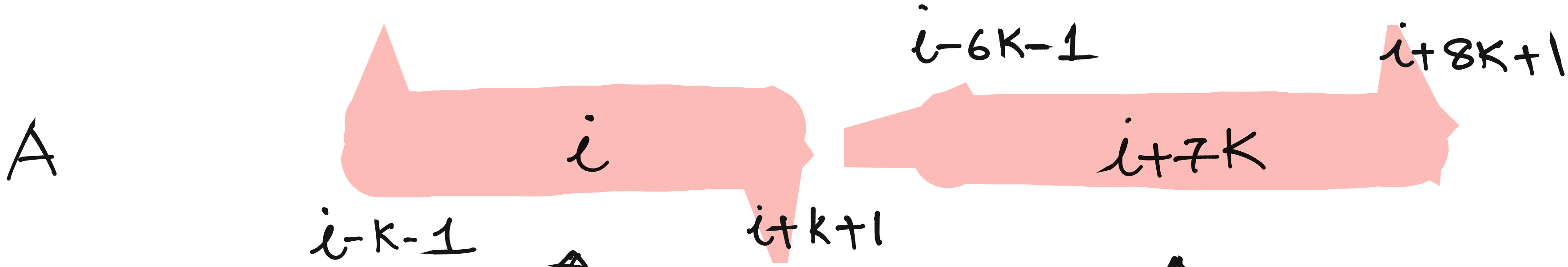
$i-K$ $\quad$ $i+K$

FINAL POSITION IN WHICH THE NUMBER AT $A[i]$ WILL LIE

FINAL POSITION IN WHICH THE NUMBER AT $A[i+7K]$ WILL LIE

A $\quad i \quad i+7K$

CAN $A[i] > A[i+7K]$?



FINAL POSITION IN WHICH THE NUMBER AT $A[i]$ WILL LIE

FINAL POSITION IN WHICH THE NUMBER AT $A[i+7K]$ WILL LIE

$\Rightarrow A[i] < A[i+7K]$

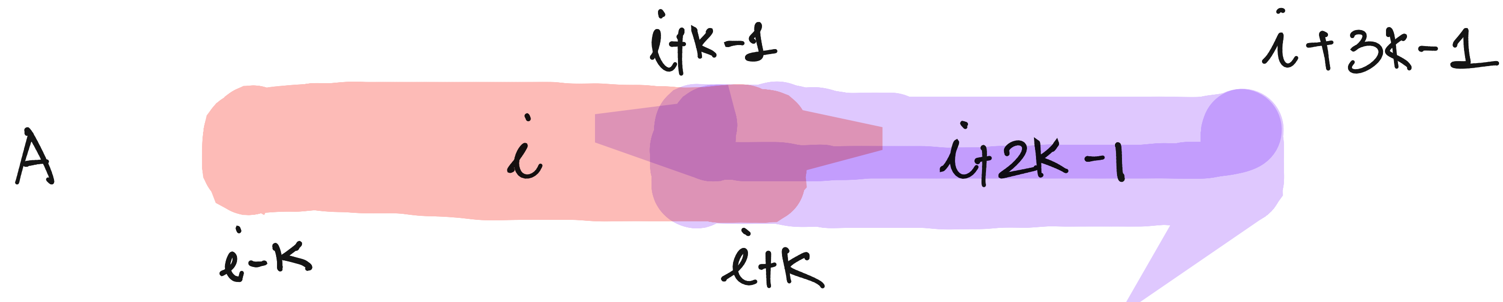
$\Rightarrow (i, i+7K)$ DONOT FORM AN INVERSION PAIR

A $\quad\quad i \quad\quad i+2K-1$

CAN $A[i] > A[i+2k-1]$?

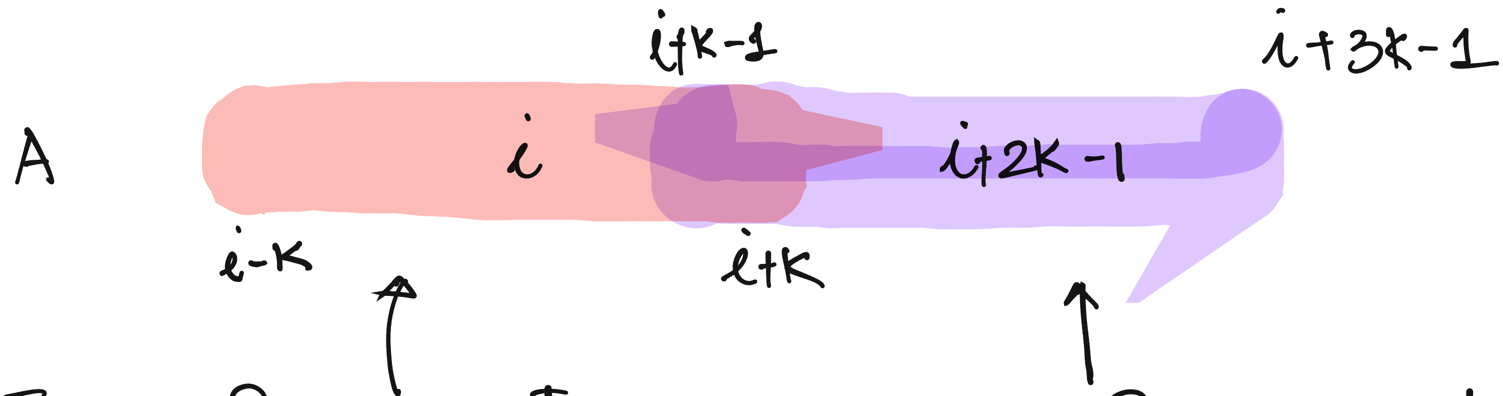
$A \qquad\qquad i \qquad\qquad i+2K-1$

CAN $A[i] > A[i+2k-1]$?



A $\quad i \quad i+2K-1$

CAN $A[i] > A[i+2k-1]$?



FINAL POSITION IN WHICH THE NUMBER AT $A[i]$ WILL LIE

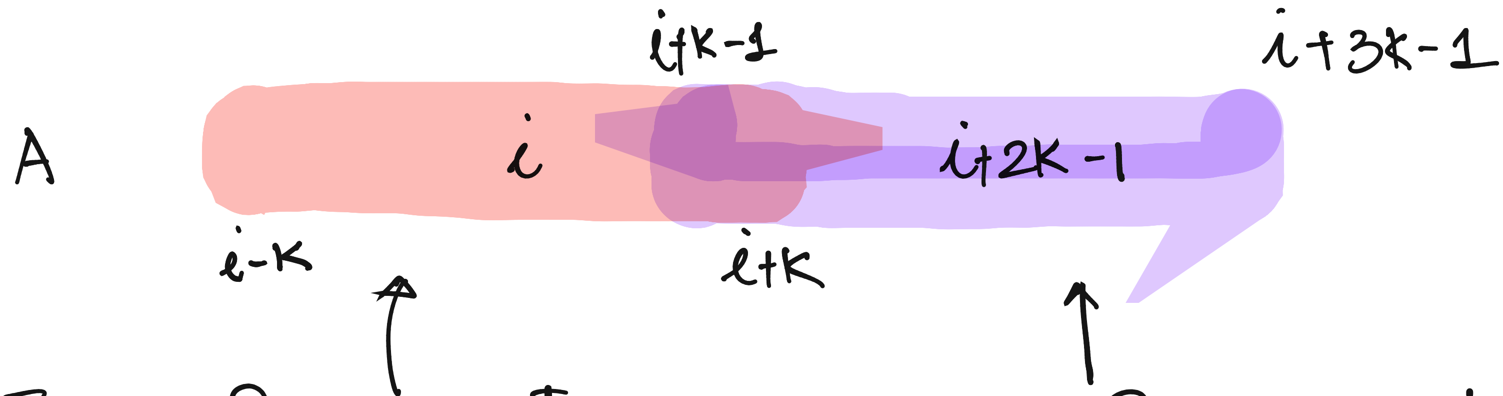
FINAL POSITION IN WHICH THE NUMBER AT $A[i+2K-1]$ WILL LIE

MAY BE $A[i+K]$

MAY BE $A[i+k-1]$

A $\quad i \quad i+2K-1$

CAN $A[i] > A[i+2K-1]$?



FINAL POSITION IN WHICH THE NUMBER AT $A[i]$ WILL LIE

MAY BE $A[i+K]$

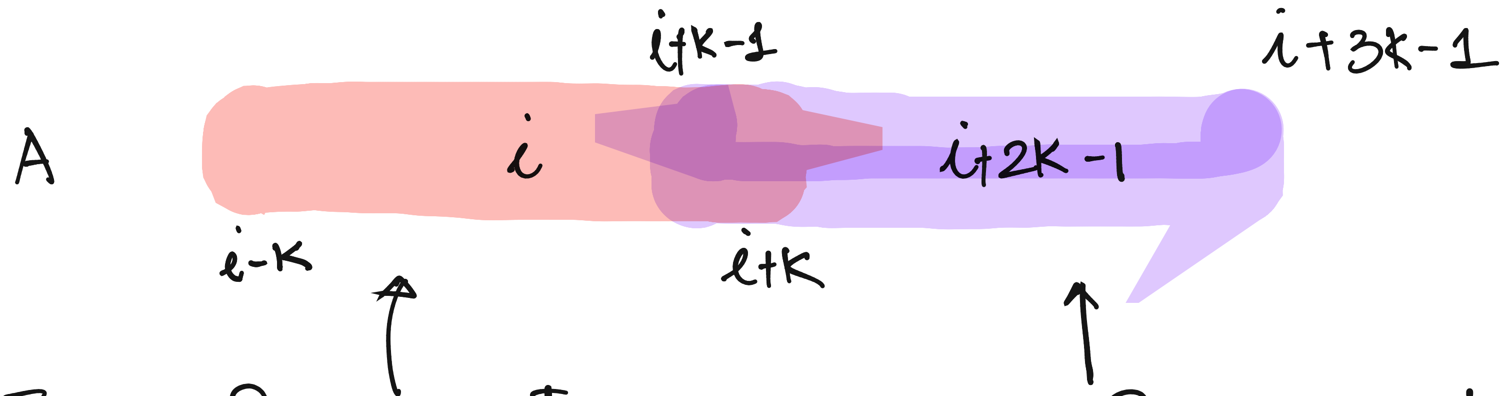
FINAL POSITION IN WHICH THE NUMBER AT $A[i+2K-1]$ WILL LIE

MAY BE $A[i+K-1]$

$\Rightarrow$ $(i, i+2K-1)$ MAY FORM AN INVERSION PAIR

A $\quad i \quad i+2k-1$

CAN $A[i] > A[i+2k-1]$?



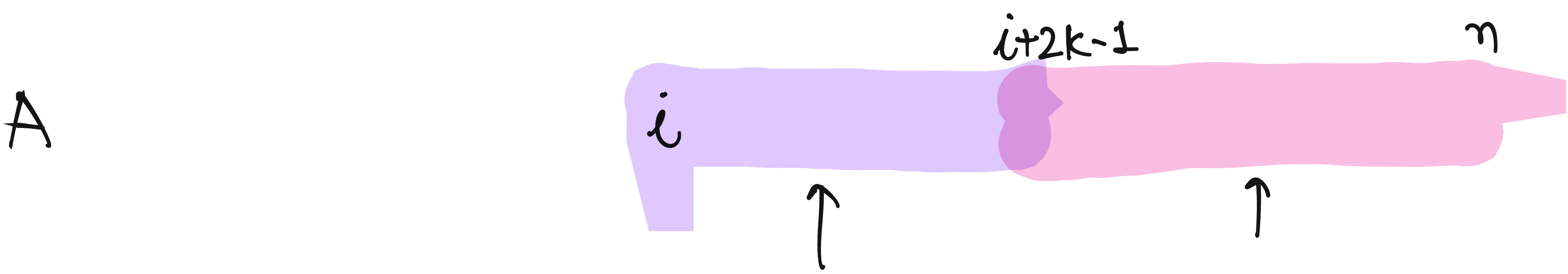
FINAL POSITION IN WHICH THE NUMBER AT $A[i]$ WILL LIE

FINAL POSITION IN WHICH THE NUMBER AT $A[i+2k-1]$ WILL LIE

MAY BE $A[i+k]$

MAY BE $A[i+k-1]$

$\Rightarrow (i, i+2k-1)$ MAY FORM AN INVERSION PAIR

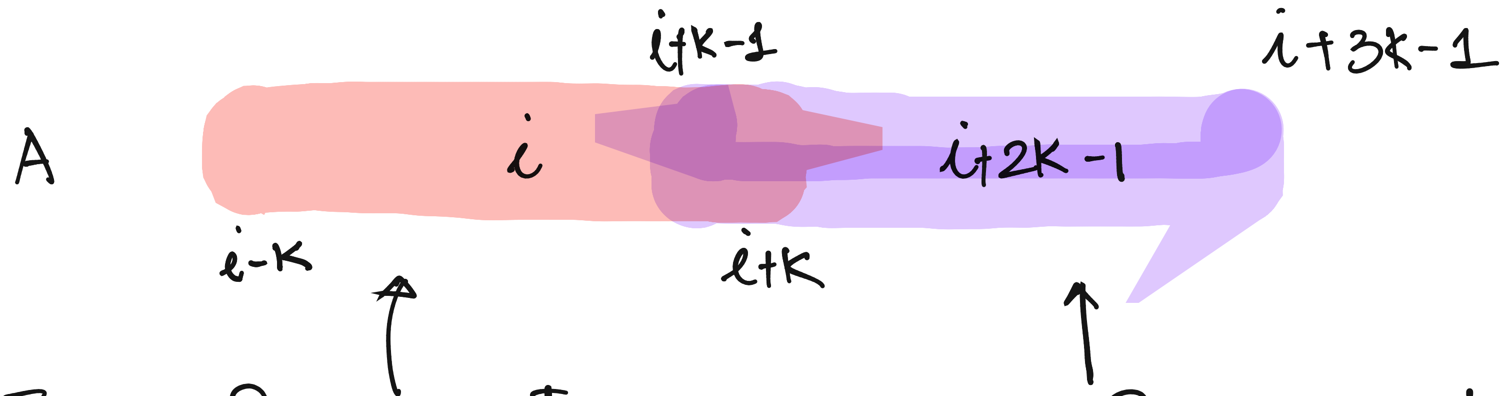


CAN FORM AN INVERSION PAIR

CANNOT FORM AN INVERSION PAIR

A $\quad i \quad i+2K-1$

CAN $A[i] > A[i+2k-1]$?



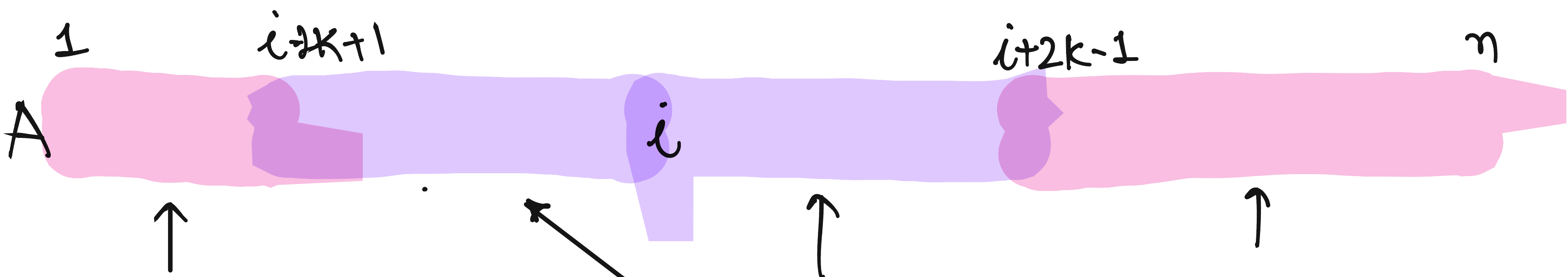
FINAL POSITION IN WHICH THE NUMBER AT A[i] WILL LIE

MAY BE A[i+K]

FINAL POSITION IN WHICH THE NUMBER AT A[i+2K-1] WILL LIE

MAY BE A[i+K-1]

$\Rightarrow$ $(i, i+2K-1)$ MAY FORM AN INVERSION PAIR



CANNOT FORM AN INVERSION PAIR

CAN FORM AN INVERSION PAIR

CANNOT FORM AN INVERSION PAIR

$\Rightarrow$ # INVERSION PAIR INVOLVING $i \le 4k$

$\Rightarrow$ # INVERSION $= \sum_{i=1}^{n} 4k$

$= O(nk)$

$\Rightarrow$ # INVERSION PAIR INVOLVING $i \leq 4k$

$\Rightarrow$ # INVERSION $= \sum_{i=1}^{n} 4k$

$= O(nk)$

$\Rightarrow$ RUNNING TIME OF INSERTION SORT

$= O(n +$ # INVERSIONS$)$

$= O(n + nk)$

$= O(nk)$