



MAKE  
SCHOOL

# LINKED LISTS

*Like a freight train*

# CHOOSE THE RIGHT TOOL FOR THE JOB

It's all about the context.

What is the shape of the data?

What are the constraints?

Which operations need to be fast?

# LINKED LISTS

# QUICK REVIEW - ARRAYS

Contiguous piece of memory

Same size storage space at each index

Static - Memory allocated once, size can't change

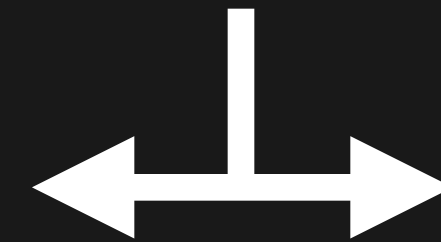
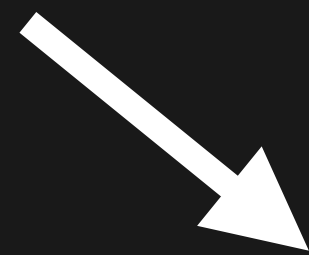
Dynamic - New memory allocated, array copied to grow

Address

Size

$A[0] = 2000$

$S = 6$



0

1

2

3

4

5

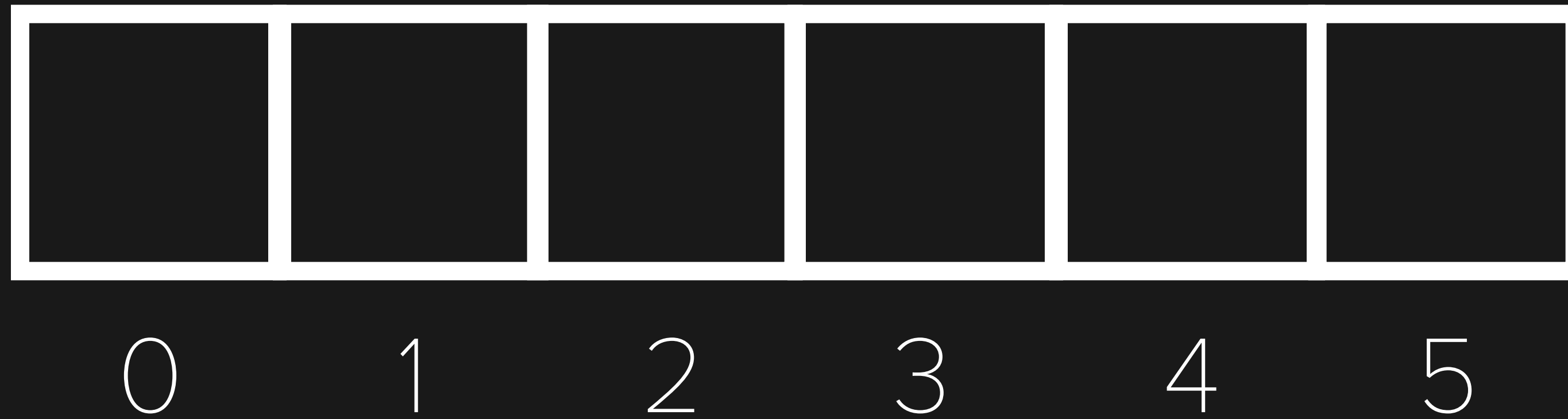
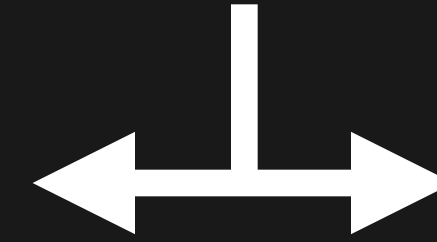
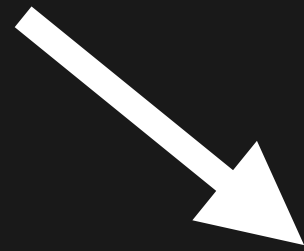
Equation to find memory location for index 4?

Address

$A[0] = 2000$

Size

$S = 6$



$$A[i] = A[0] + S * i$$

$$A[4] = 2000 + 6 * 4$$

$$A[4] = 2024$$

# DYNAMIC ARRAY RUNTIME

Operation	Worst Case
-----------	------------

Access Element Via Index	
--------------------------	--

	$O(1)$
--	--------



0 1 2 3 4 5

Insert or Delete Element (Beginning, Middle)	
---	--

	$O(n)$
--	--------

Insert or Delete Element (End)	
-----------------------------------	--

	$O(1)$
--	--------



# LINKED LISTS

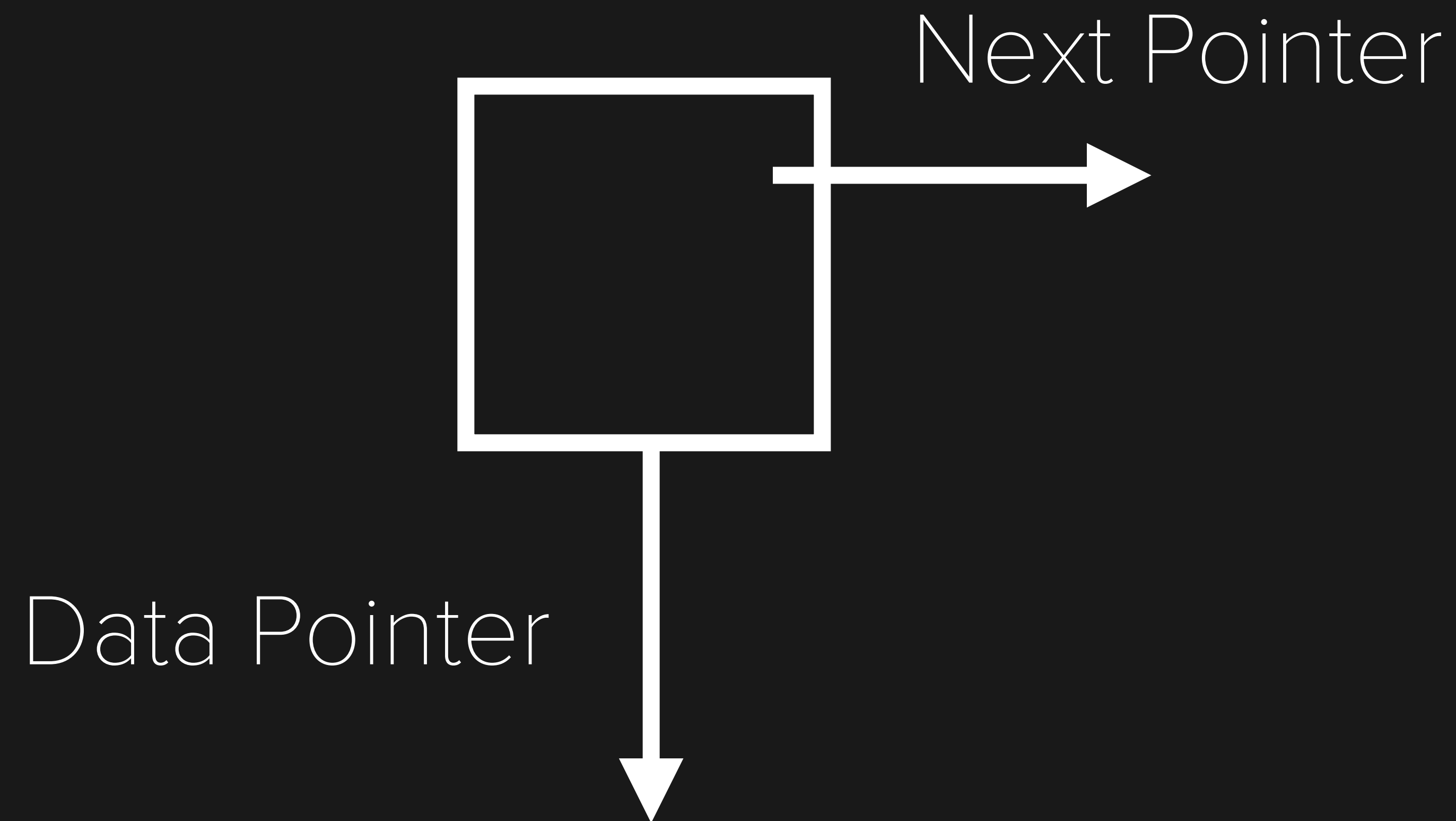
Not contiguous piece of memory

Differing size storage space at each index

Dynamic - New (small) piece of memory allocated

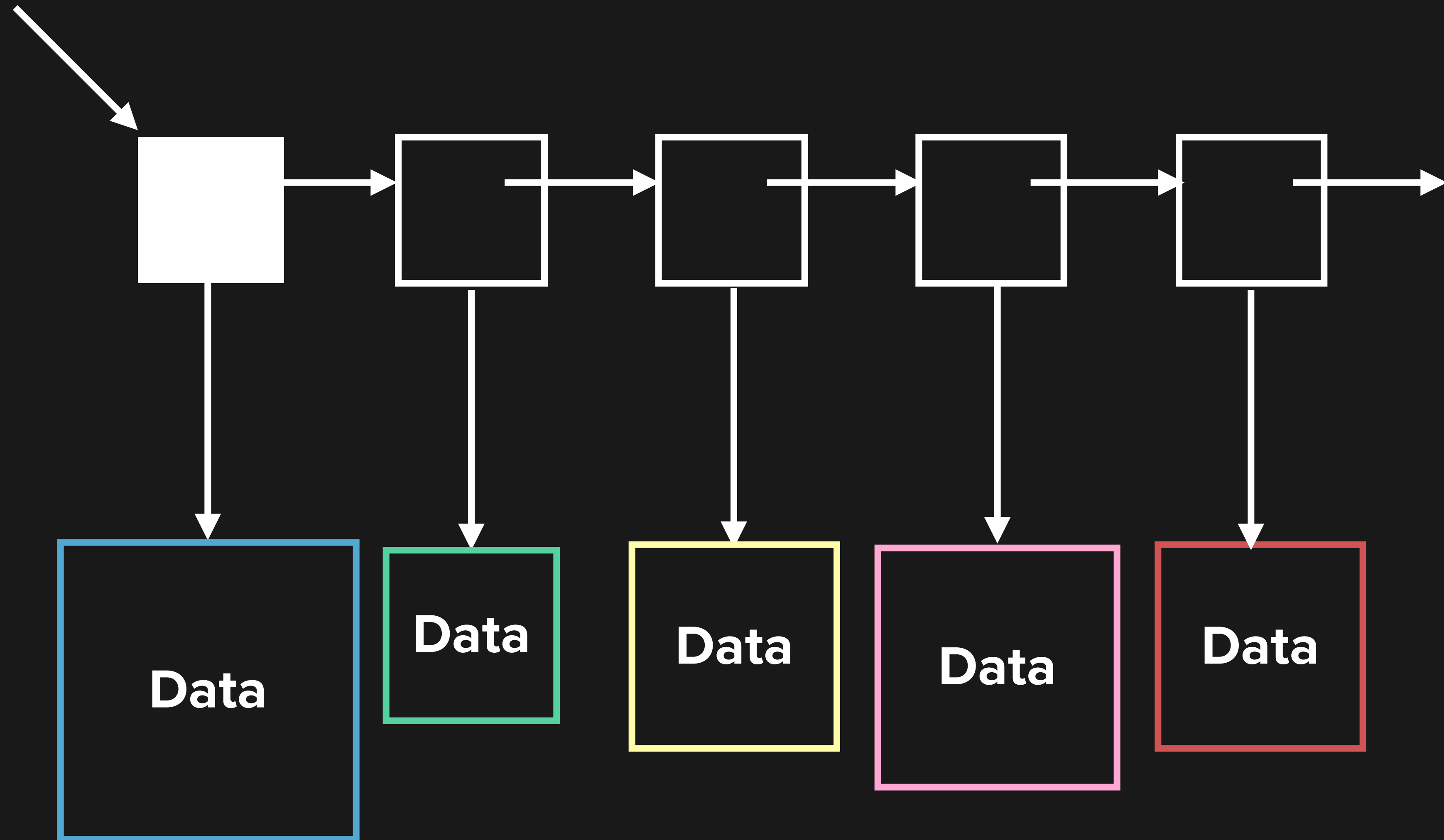
No need to copy the whole thing like an array

# NODE



# LINKED LIST

Head Pointer



# LINKED LIST RUNTIME

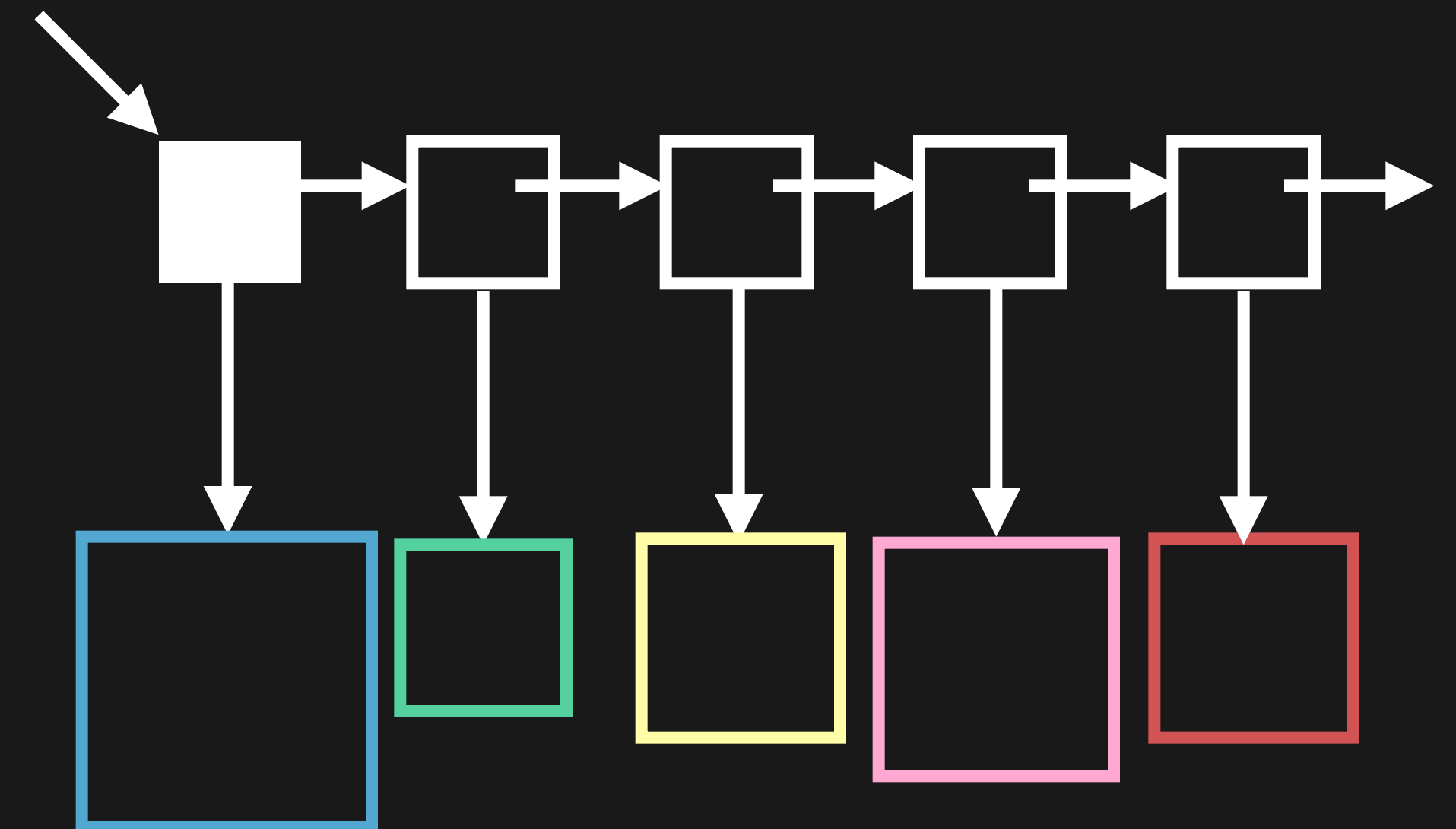
Operation	Worst Case
-----------	------------

Access Element via Index	$O(n)$
--------------------------	--------

Find and Insert or Delete Element (Beginning)	$O(1)$
---	--------

Find and Insert or Delete Element (Middle)	$O(n)$
--	--------

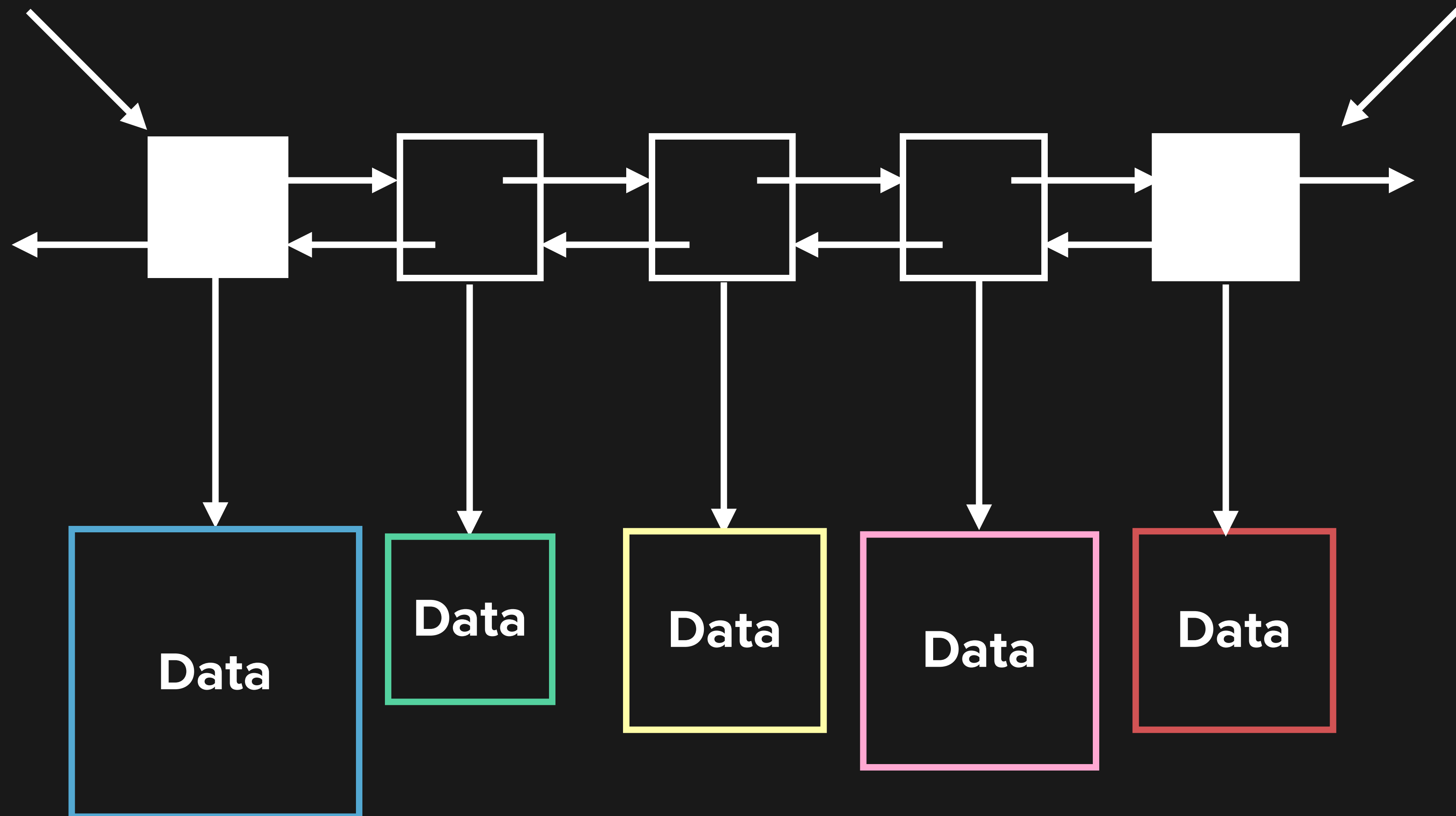
Find and Insert or Delete Element (End)	$O(n)$
---	--------



# DOUBLY LINKED LIST

Head Pointer

Tail Pointer



# DOUBLY LINKED LIST RUNTIME

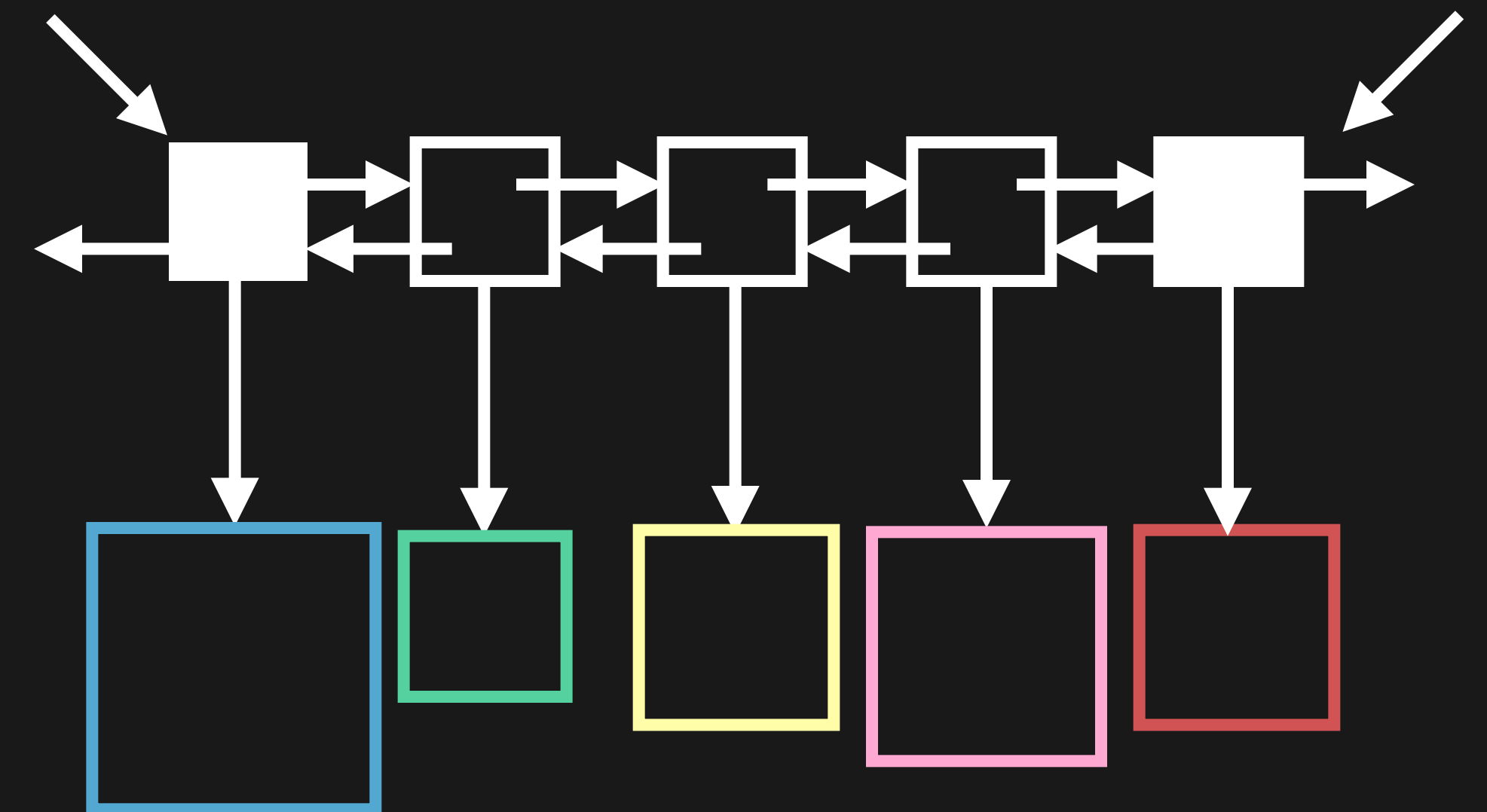
Operation	Worst Case
-----------	------------

Access Element Via Index	$O(n)$
--------------------------	--------

Insert or Delete Element (Beginning)	$O(1)$
--------------------------------------	--------

Insert or Delete Element (Middle)	$O(n)$
-----------------------------------	--------

Insert or Delete Element (End)	$O(1)$
--------------------------------	--------



**A LINKED LIST IS LIKE A  
FREIGHT TRAIN**



MAKE  
SCHOOL