COMP1511 PROGRAMMING FUNDAMENTALS

LECTURE 12

Linked Lists - traversing, inserting at the head or at the tail, searching for something

LAST TIME...

- Linked Lists - creating a list by inserting at the head.

TODAY...

- Linked Lists - traversing a list and inserting at the tail
- Linked Lists - searching for conditions

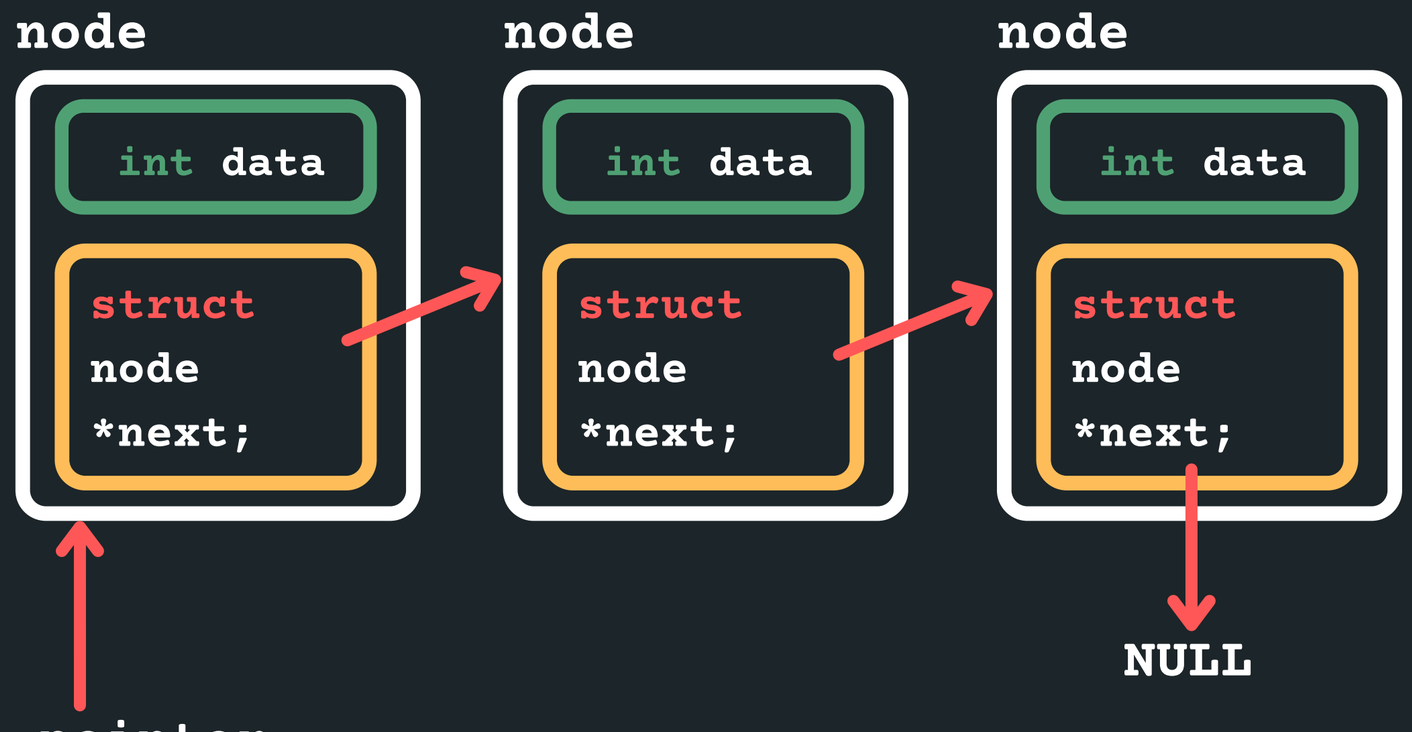
WHERE IS THE CODE?

Live lecture code can be found here:

HTTPS://CGI.CSE.UNSW.EDU.AU/~CS1511/22T1/LIVE/WEEK07/

A LINKED LIST IS MADE UP OF MANY NODES

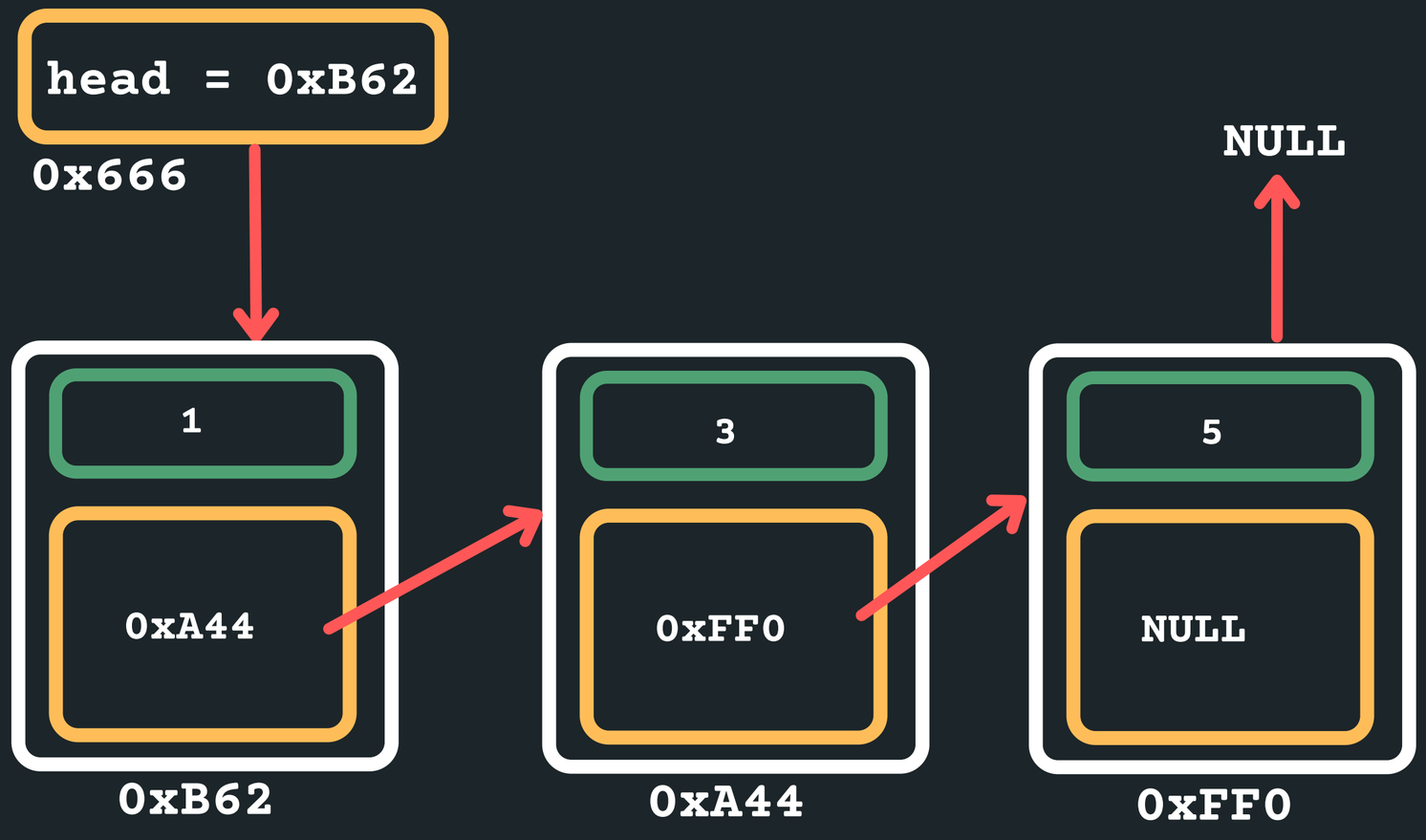
THE NODES ARE LINKED TOGETHER (A SCAVENGER HUNT OF POINTERS)



A LINKED LIST IS MADE UP OF MANY NODES

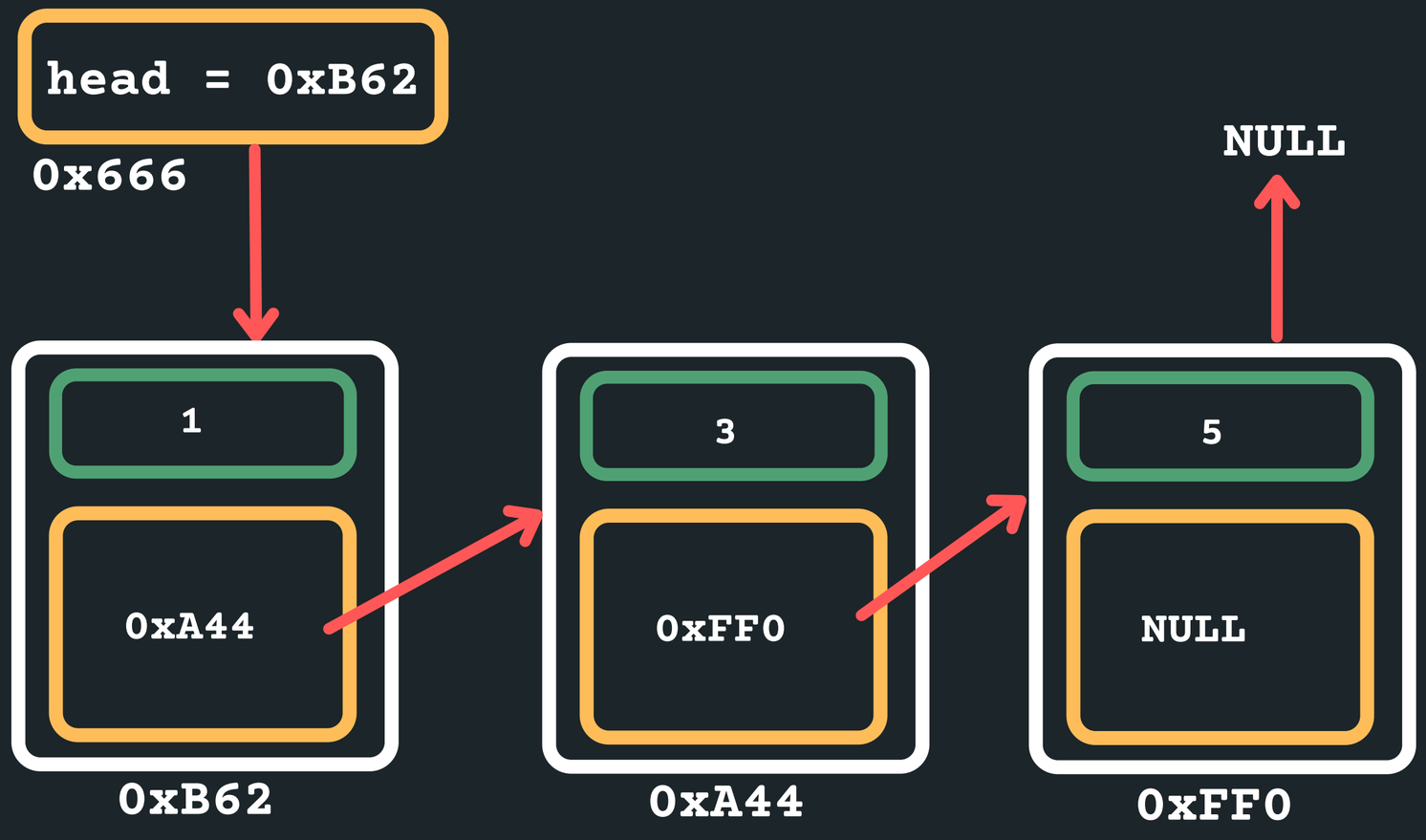
THE NODES ARE LINKED TOGETHER (A SCAVENGER HUNT OF POINTERS)

- For example a list with 1, 3, 5



HOW CAN WE MOVE THROUGH THIS LIST TO FIND NEXT NODE?

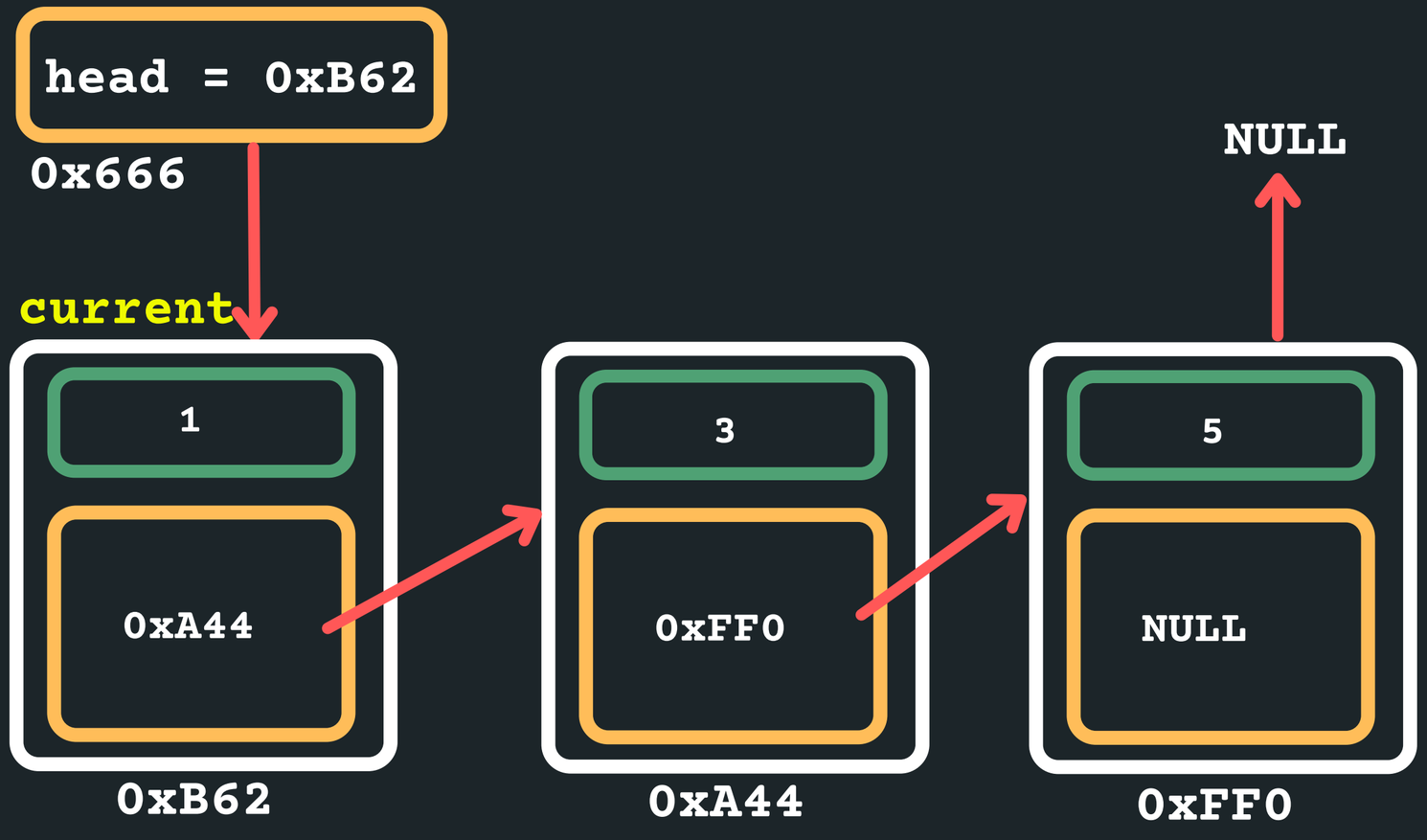
- How do you think we can move through the list to start a the head and then move to each subsequent node until we get to the end of the list...



HOW CAN WE MOVE THROUGH THIS LIST TO FIND NEXT NODE?

Set your head pointer to the current pointer to keep track of where you are currently located....

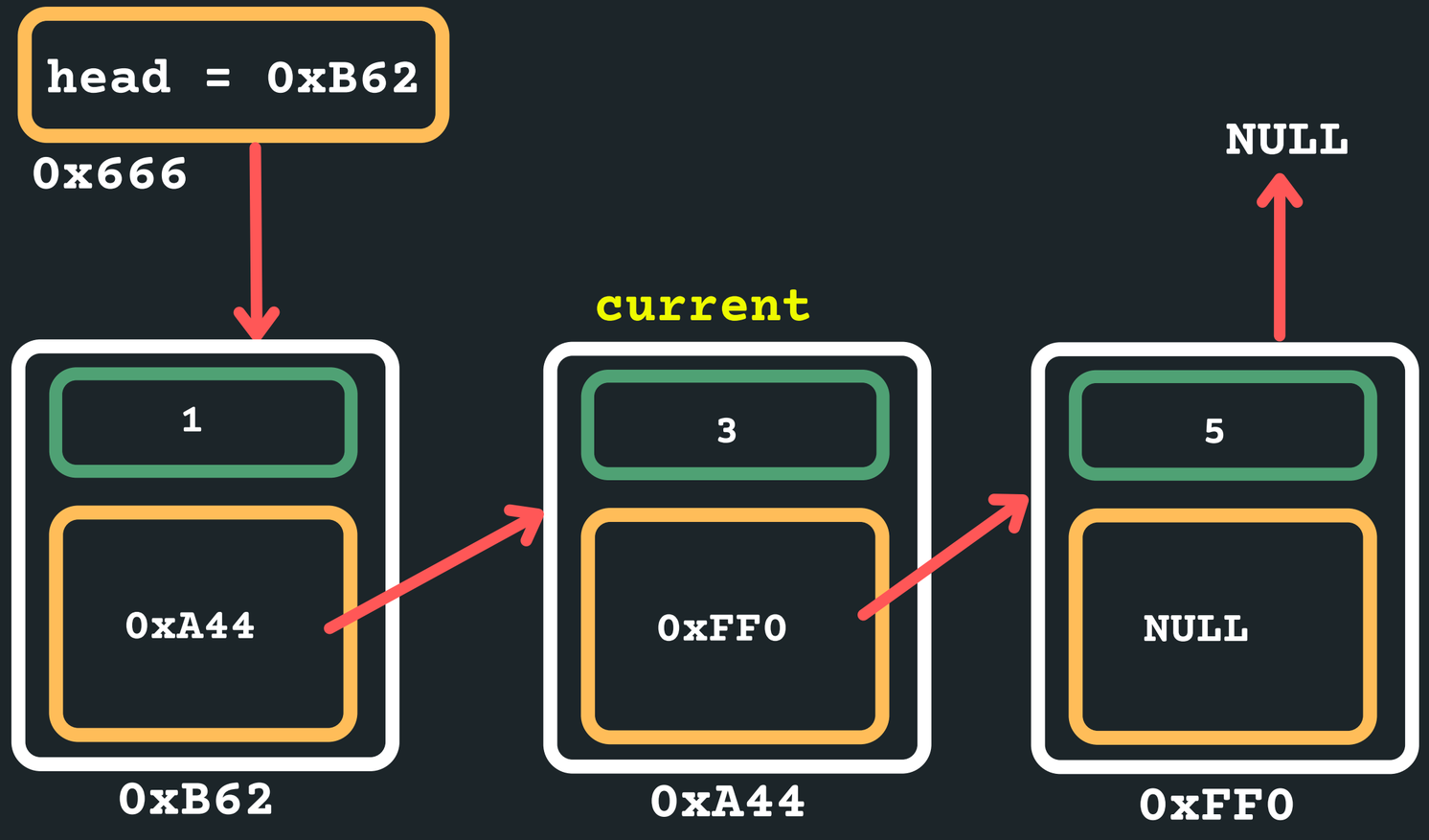
```
struct node *current = head
```



HOW CAN WE MOVE THROUGH THIS LIST TO FIND NEXT NODE?

Now how would we move the current along?

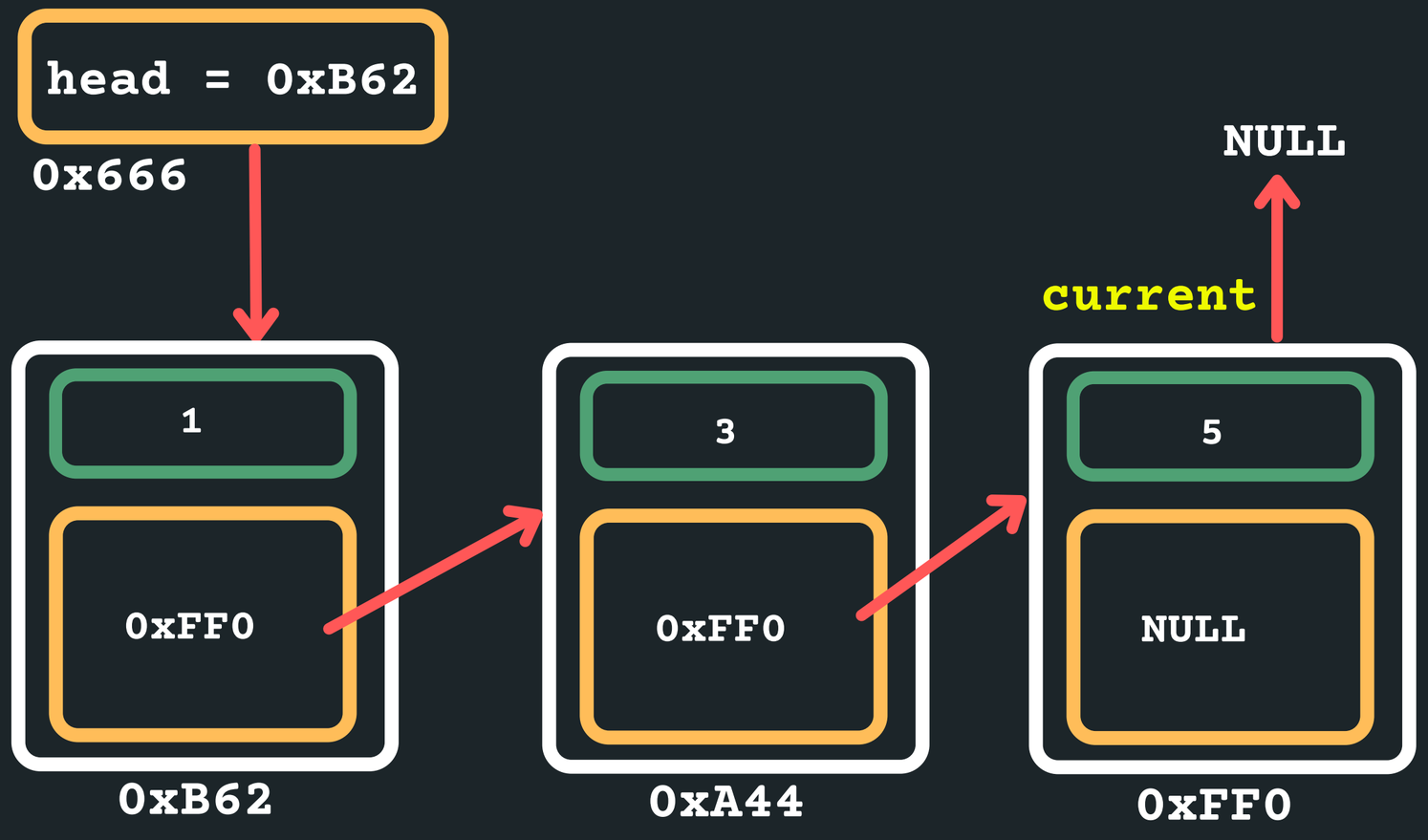
```
current = current->next
```



HOW CAN WE MOVE THROUGH THIS LIST TO FIND NEXT NODE?

Now how would we move the current along?

```
current = current->next
```



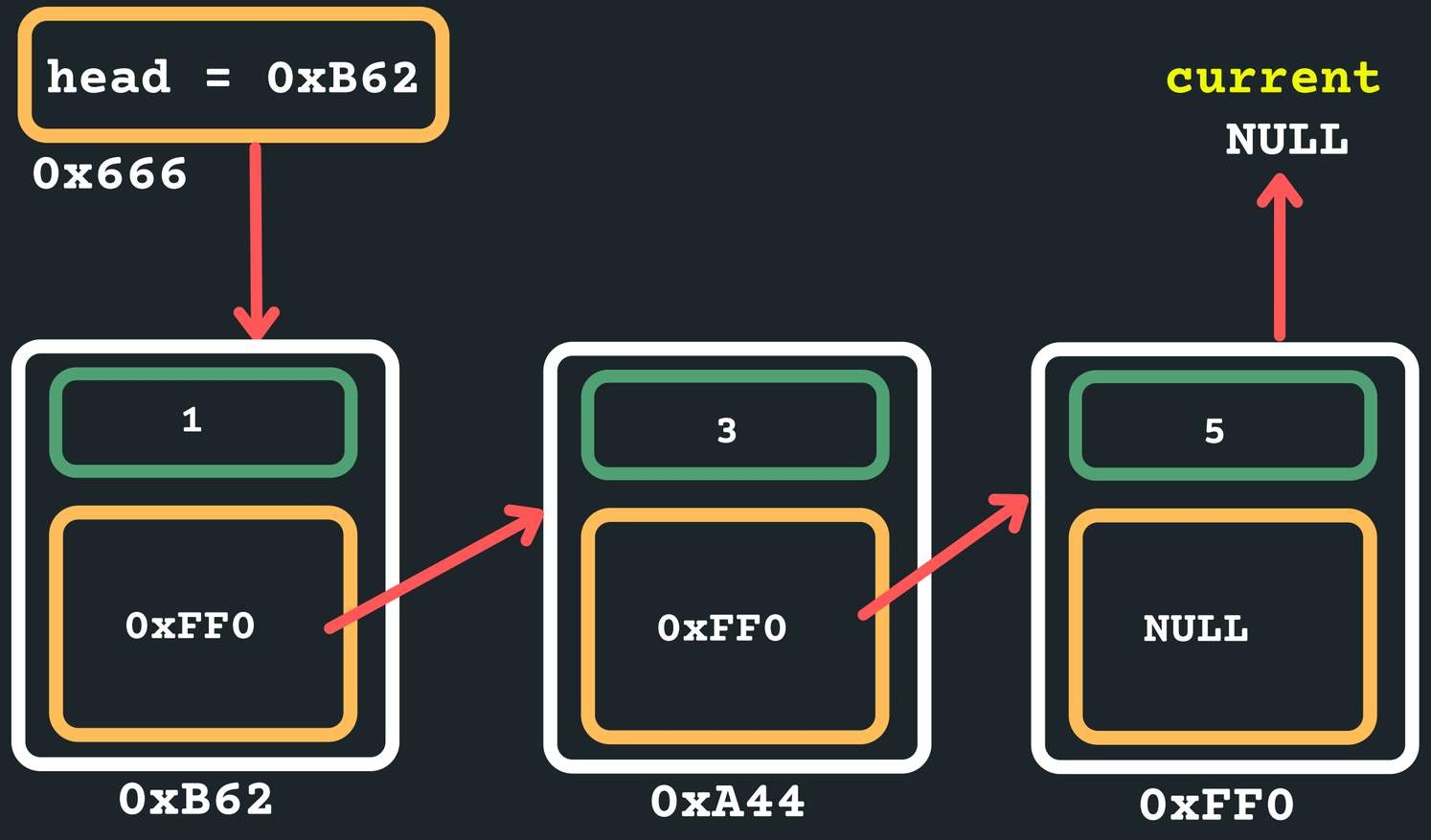
HOW CAN WE MOVE THROUGH THIS LIST TO FIND NEXT NODE?

Now how would we move the current along?

```
current = current->next
```

When should I be stopping?

```
while (current != NULL)
```



SO TRAVERSING A LINKED LIST...

- The only way we can make our way through the linked list is like a scavenger hunt, we have to follow the links from node to node (sequentially! we can't skip nodes)
- We have to know where to start, so we need to know the head of the list
- When we reach the NULL pointer, it means we have come to the end of the list.

SO NOW, LET'S PRINT EACH NODE OUT...

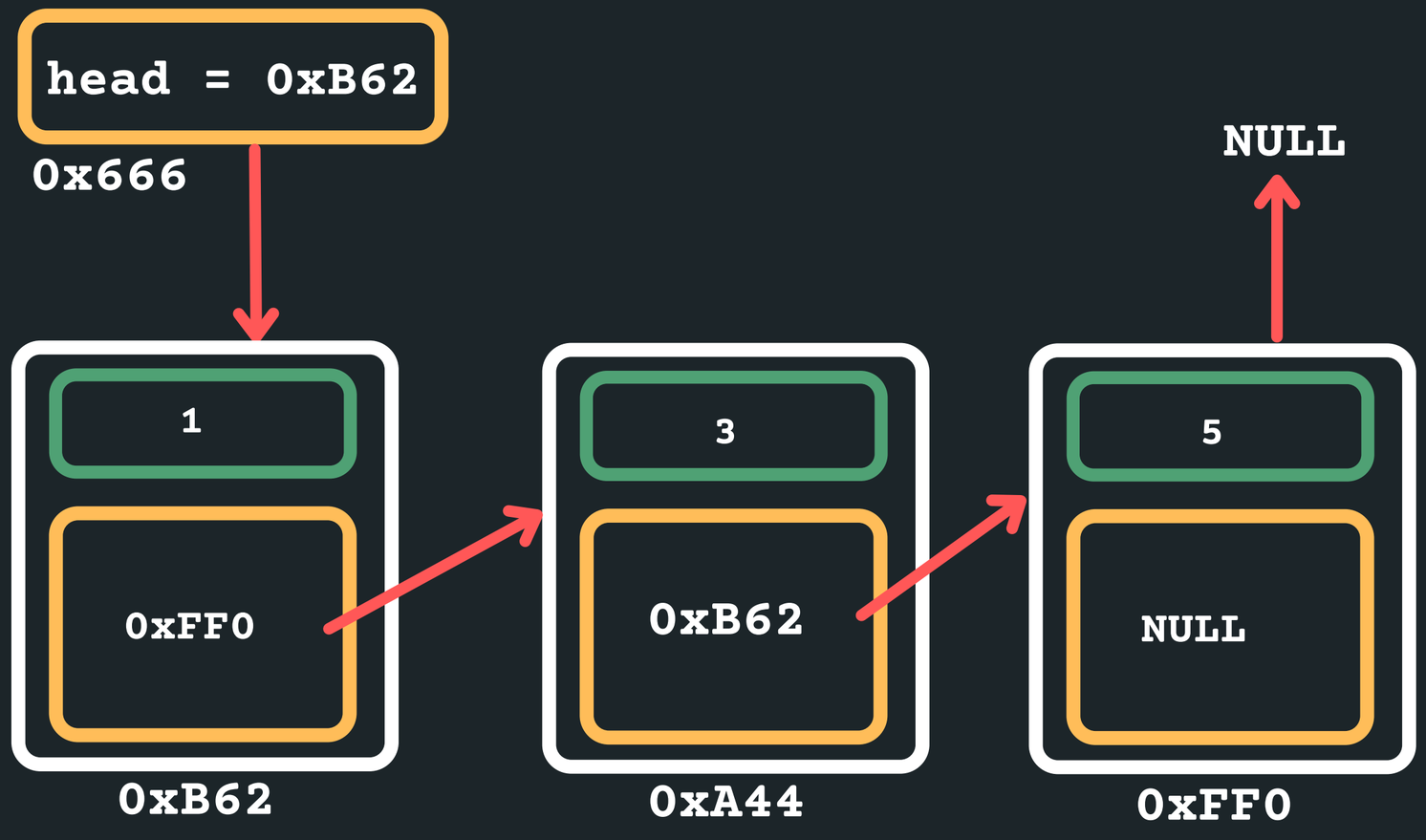
```
void print_list(struct node *head){
    struct node *current = head;
    while (current != NULL){
        printf("%d\n", current->data);
        current = current->next;
    }
}
```

BREAK TIME...

You have five boxes in a row numbered 1 to 5, in one of which, a cat is hiding. Every night he jumps to an adjacent box, and every morning you have one chance to open a box to find him. How do you win this game of hide and seek - what is your strategy? What if there are n boxes?

INSERTING ANYWHERE IN A LINKED LIST...

- Where can I insert in a linked list?
  - At the head (last lecture)
  - Between any two nodes that exist
  - After the tail as the last node



FINDING WHERE TO INSERT

- I could have a condition that will help me find at which point to insert (specified by my problem)
- In my list, for example, it could be that I want to put a 4 between 3 and 5...
- This would involve searching through the list to find 3
    - Or if the list is in order, it may be to find the value less than the one I am inserting and the value after to be greater than the value I am inserting and then insert by creating a new node and linking it to the right space...

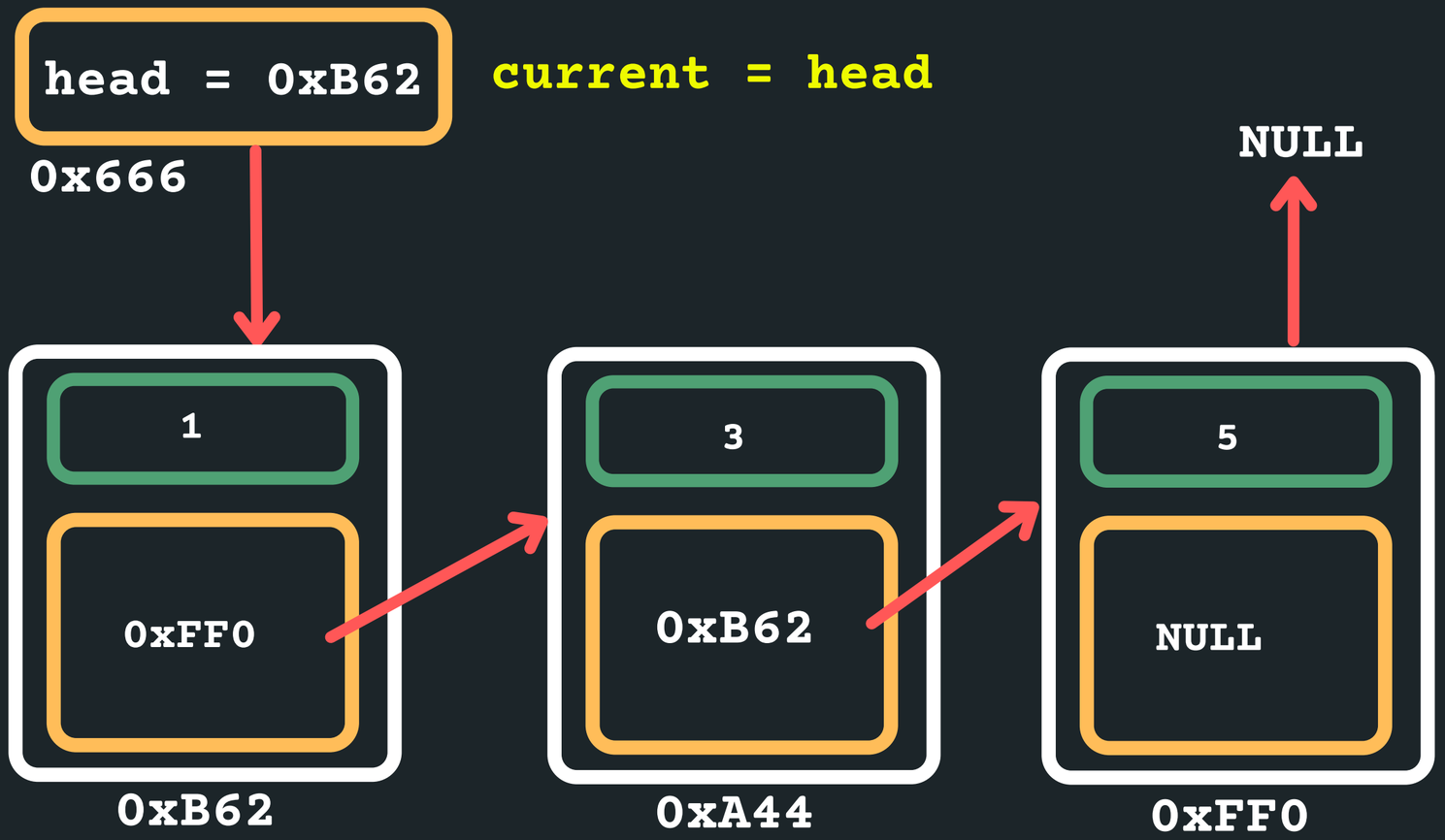
LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Find where to insert: Is current less than 5 AND next more than 5?

Set current to the head of the list to begin traversal

```
struct node *current = head
```



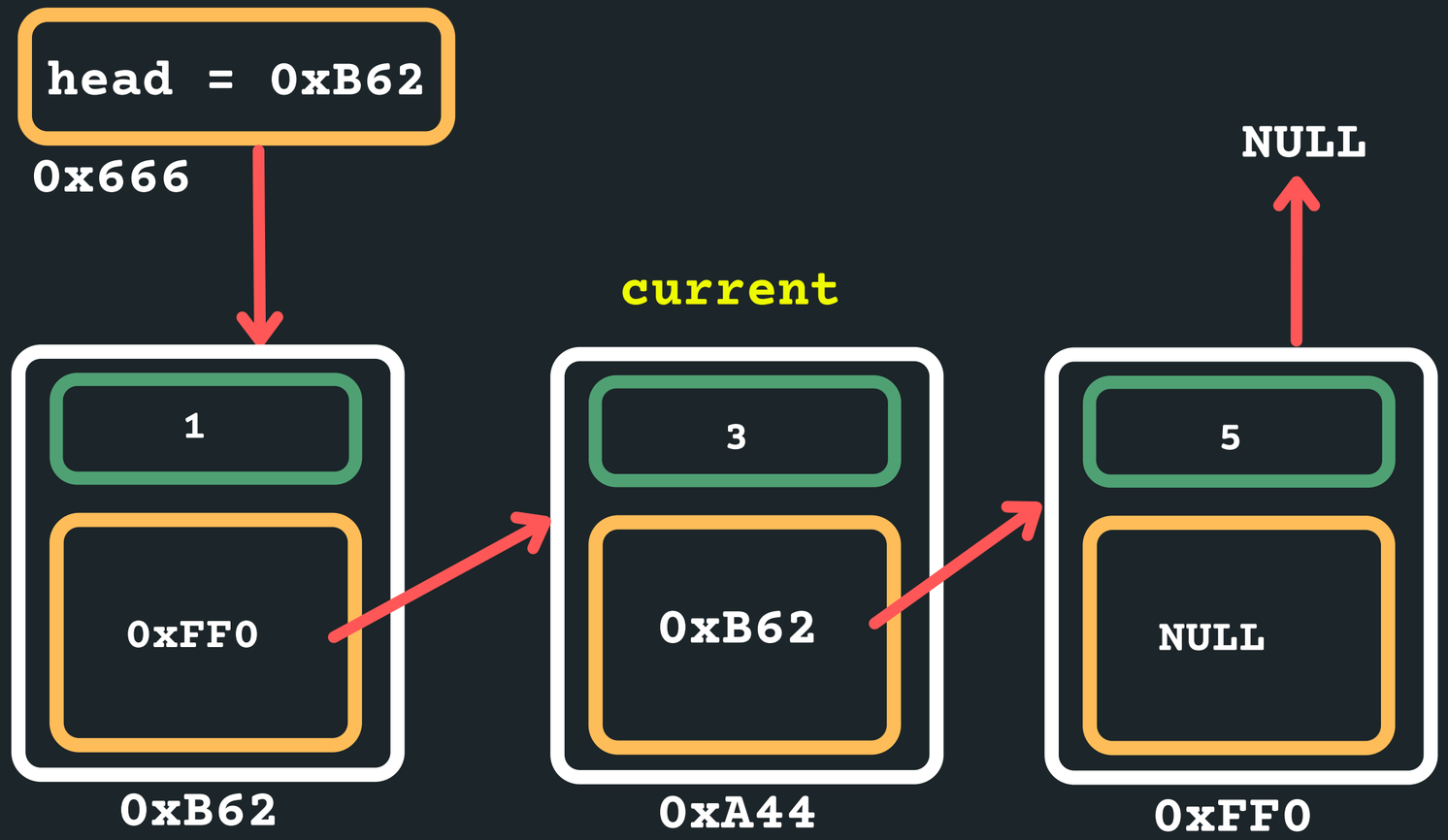
LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Find where to insert: Is current less than 5 AND next more than 5?

Traverse list until you find the right node to insert after...

```
current = current->next
```



LINKED LIST EXAMPLE INSERTING BETWEEN NODES

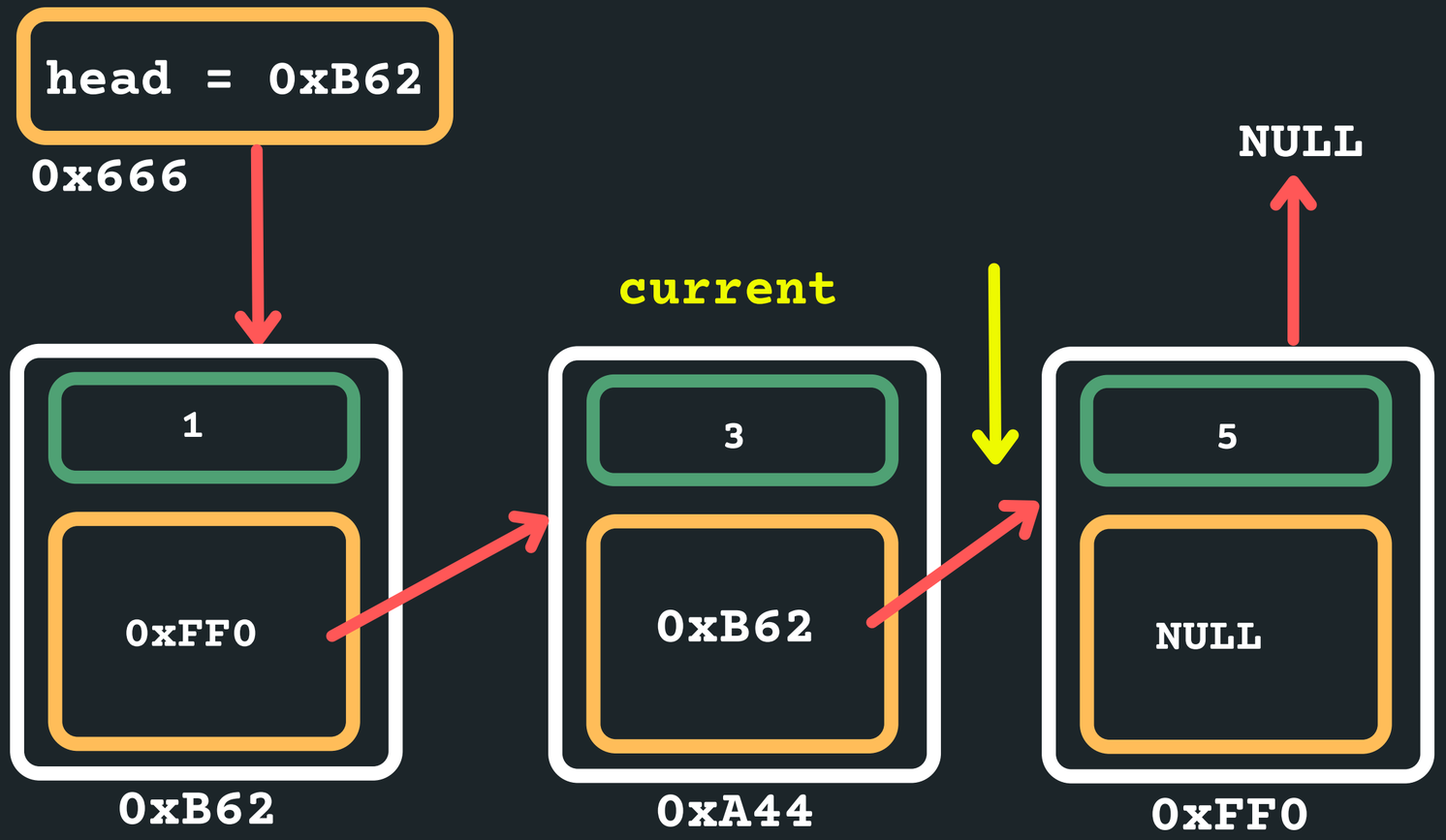
INSERT 4 (AFTER 3 AND BEFORE 5)

Find where to insert: Is current less than 5 AND next more than 5?

```
current->data < 5 && current->next->data > 5
```

Traverse list until you find the right node to insert after...

```
current = current->next
```

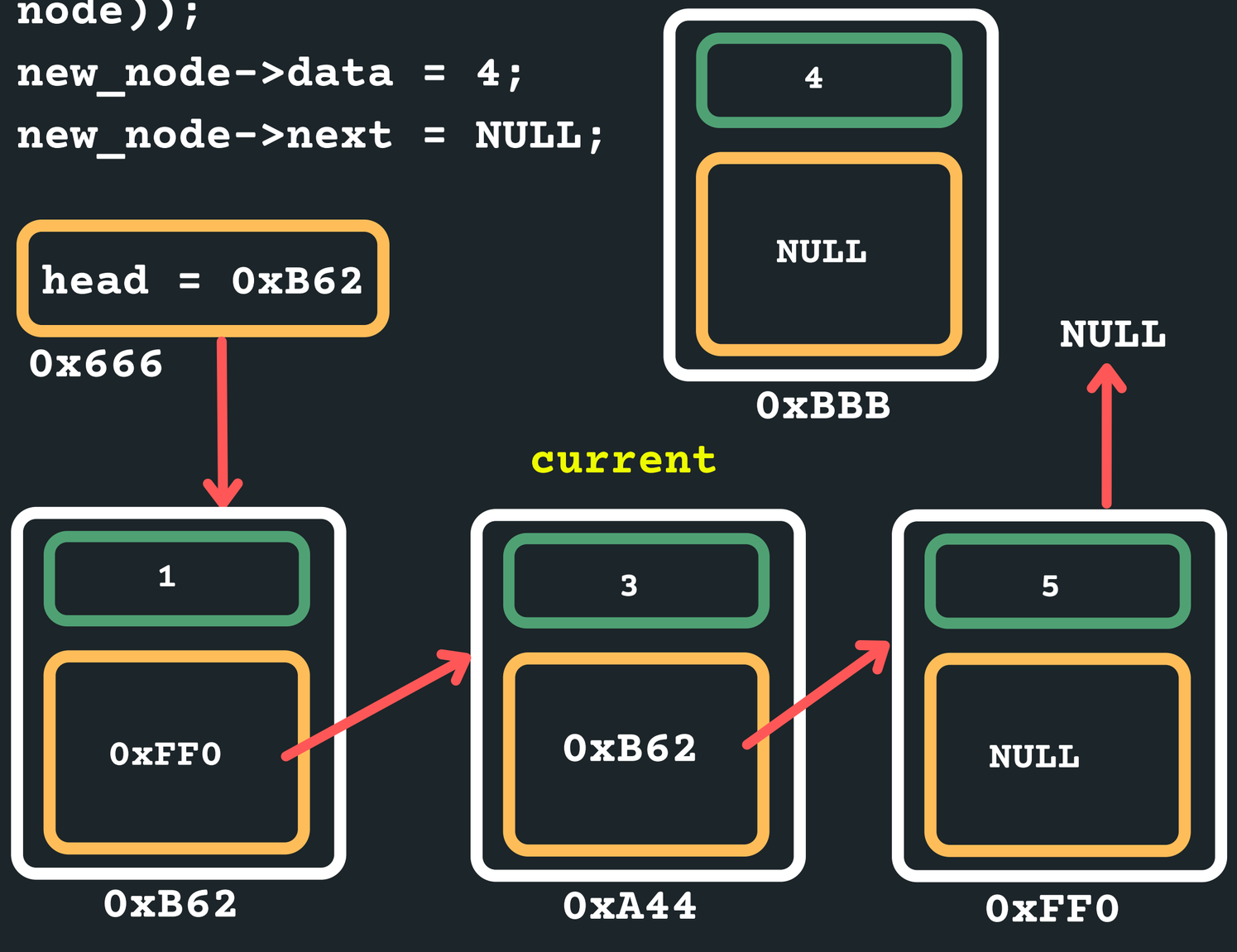


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Now that you found location to insert, create the node

```
struct node *new_node = malloc(sizeof(struct node));
new_node->data = 4;
new_node->next = NULL;
```

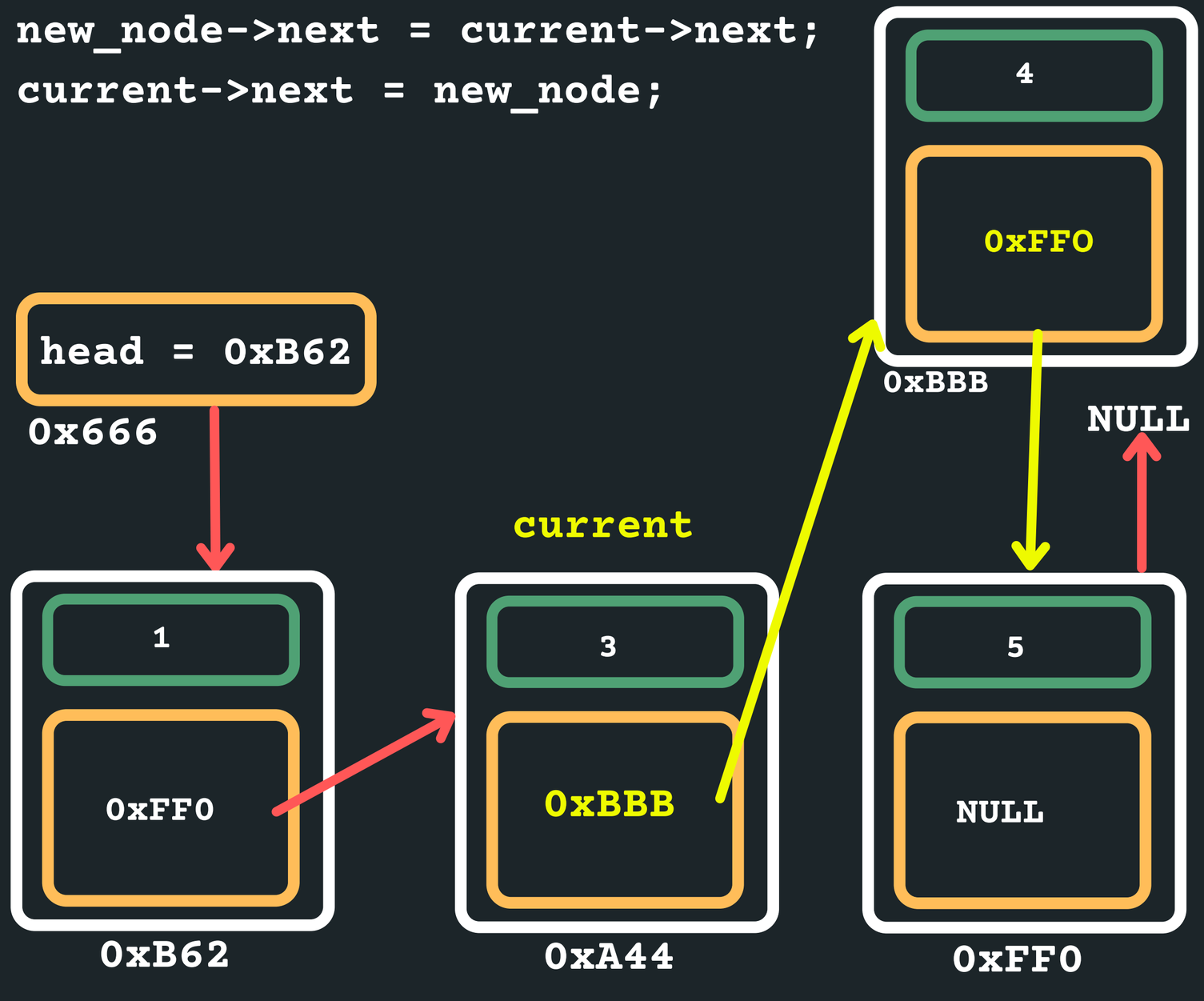


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Now insert the node after the current node...

```
new_node->next = current->next;
current->next = new_node;
```



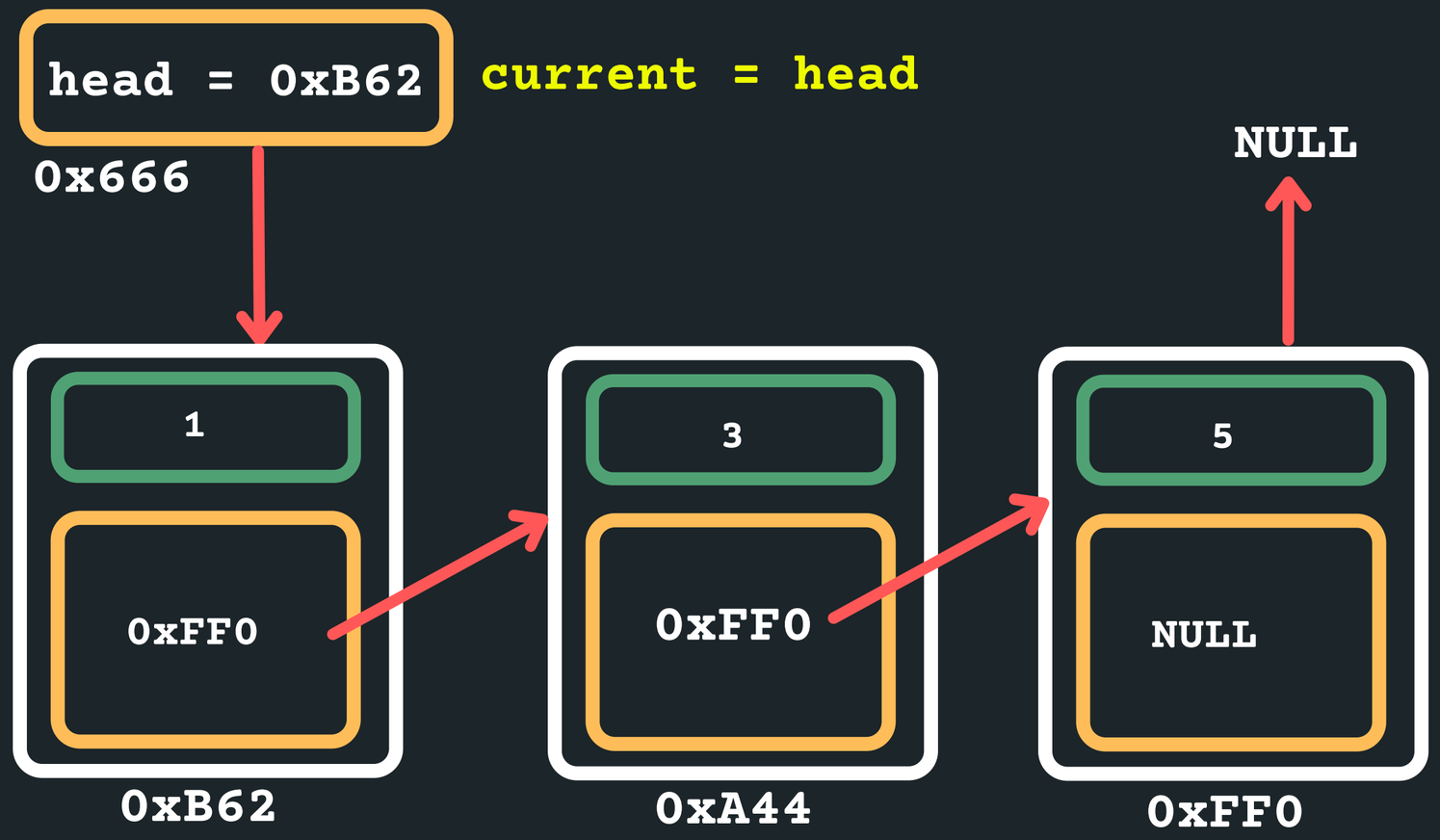
LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Now let's insert at the end of the list... let's insert 10

Set current to the head of the list to begin traversal until the end of the list....

```
struct node *current = head
```

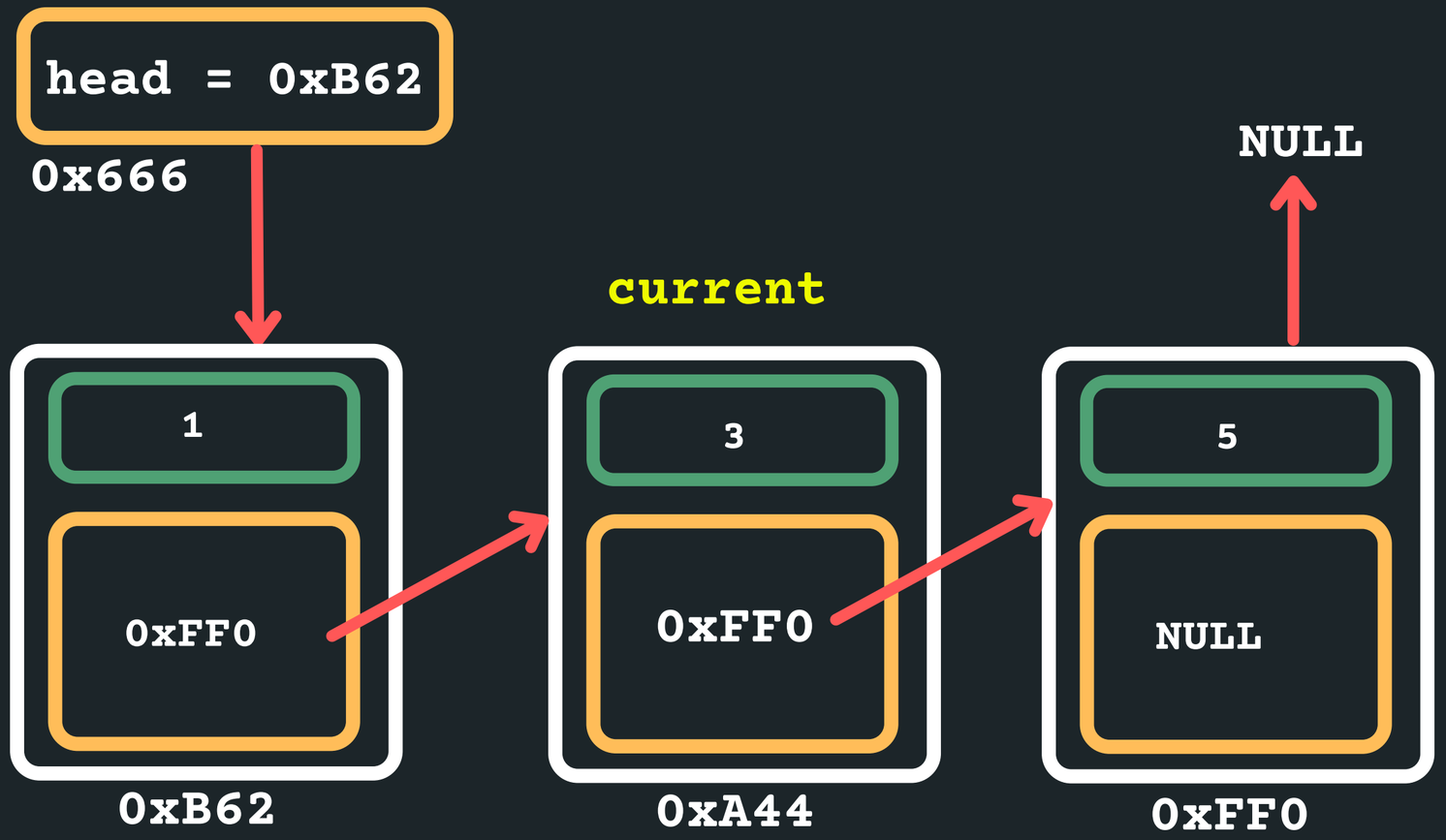


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Moving along the list now - 1st loop

`current = current->next`

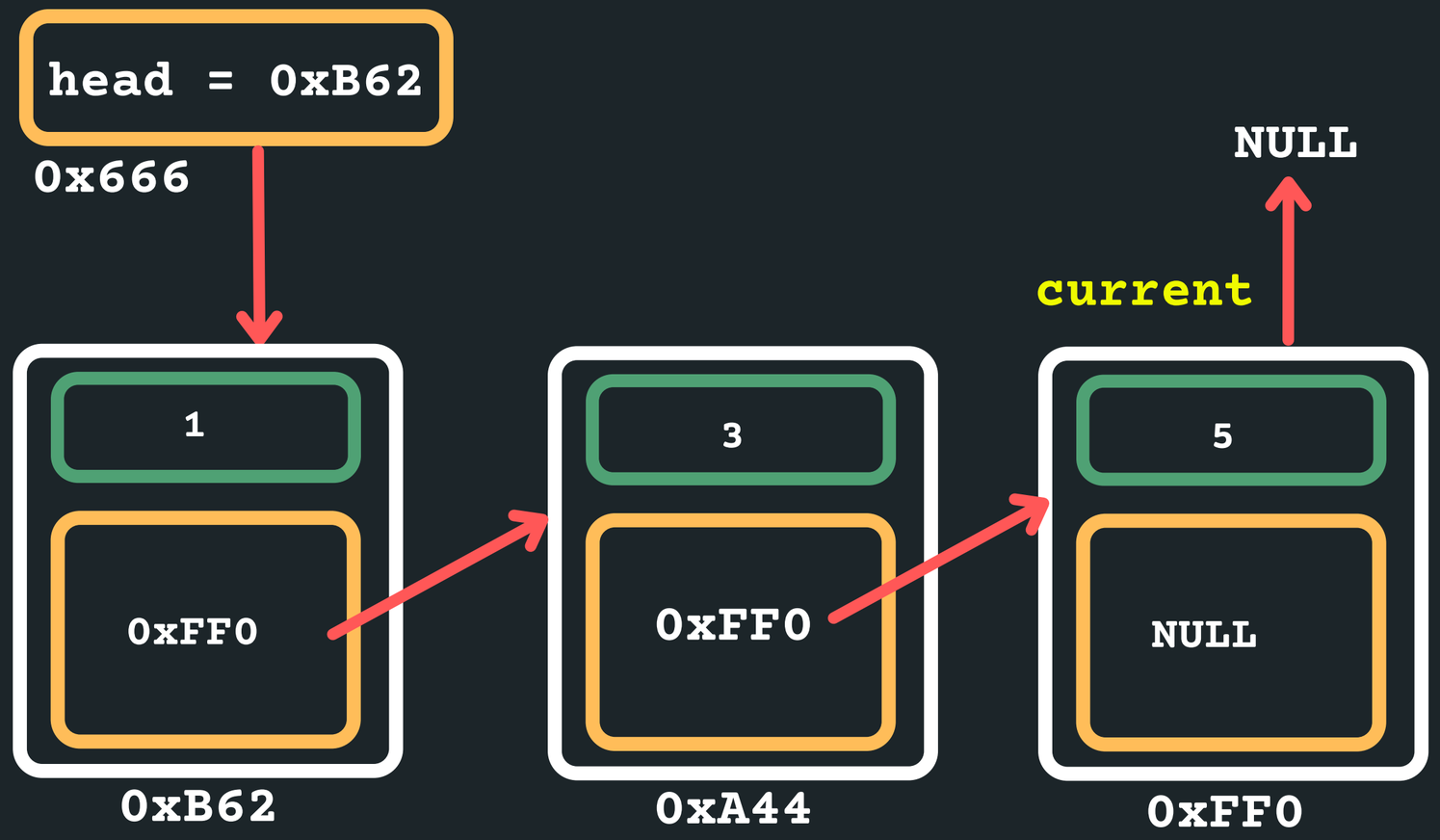


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Moving along the list now - 2nd loop

```
current = current->next
```

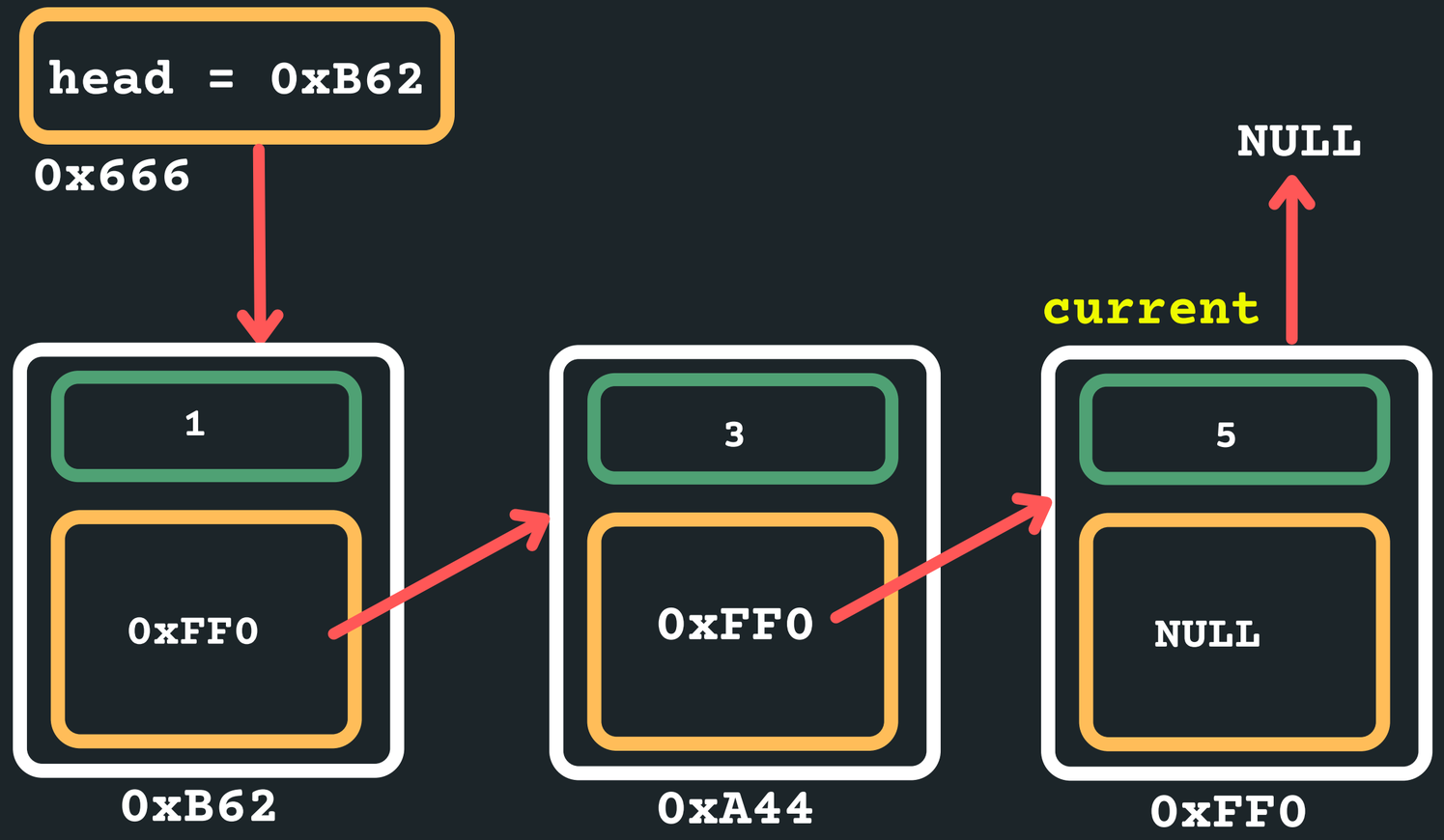


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Now we want to stop before we get to the NULL, so we want to stop at the last node and not go past it... so stop here and not at NULL

```
while (current->next != NULL)
```

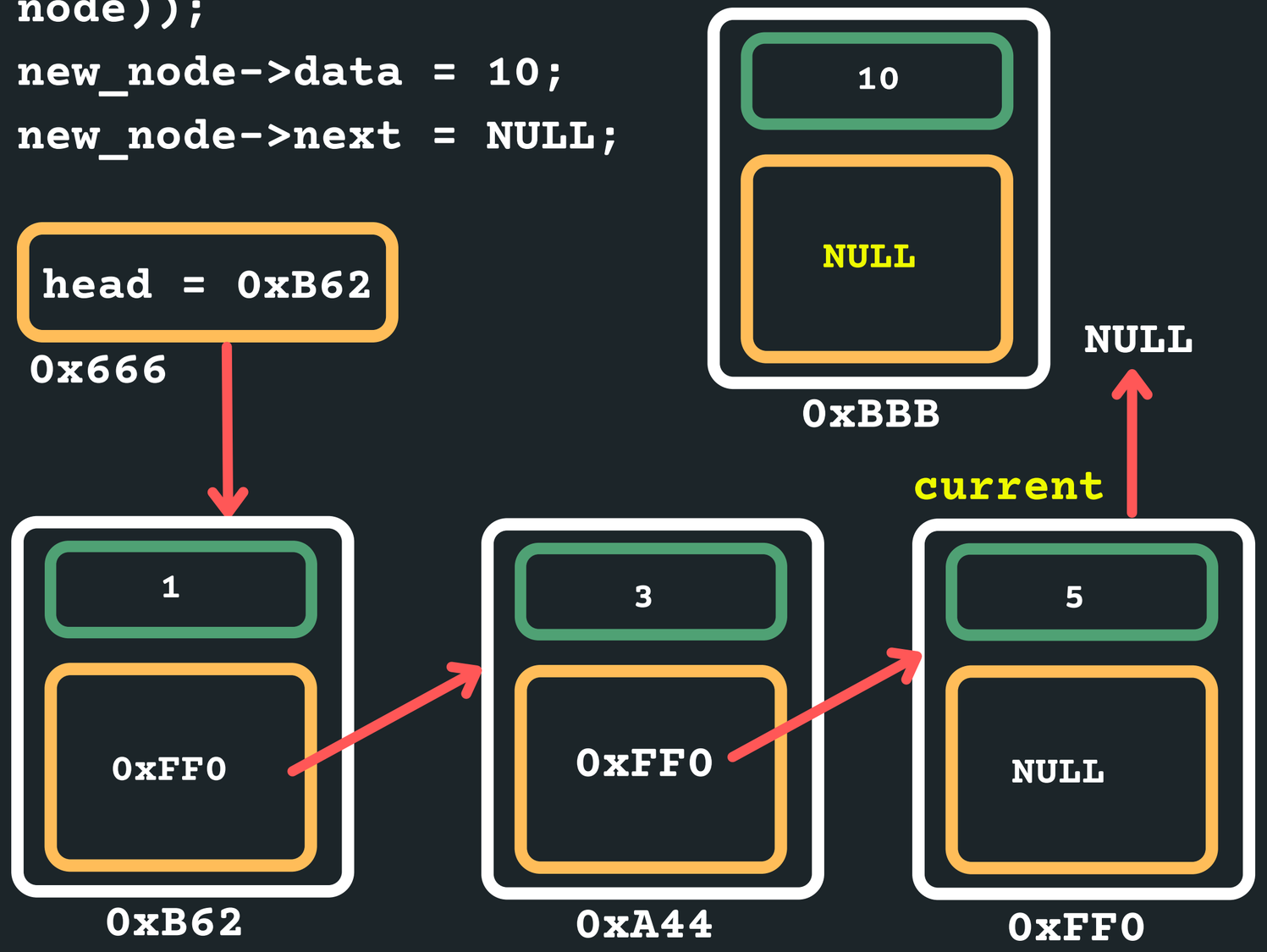


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Now that we know where to insert, create the node

```
struct node *new_node = malloc(sizeof(struct node));
new_node->data = 10;
new_node->next = NULL;
```

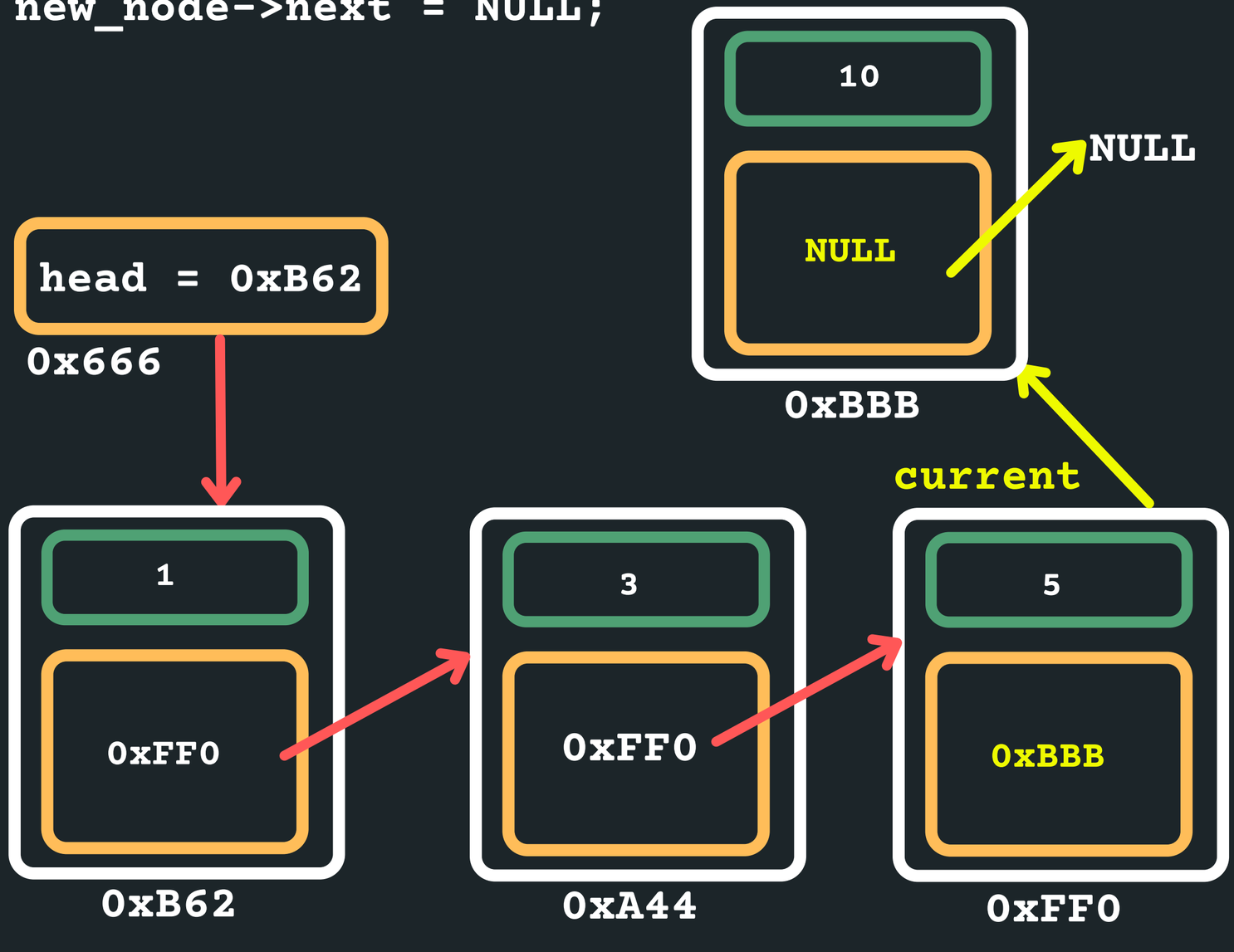


LINKED LIST EXAMPLE INSERTING BETWEEN NODES

INSERT 4 (AFTER 3 AND BEFORE 5)

Now we have a node, connect it...

```
current->next = new_node;
new_node->next = NULL;
```



Feedback please!

I value your feedback and use to pace the lectures and improve your overall learning experience. If you have any feedback from today's lecture, please follow the link below. Please remember to keep your feedback constructive, so I can action it and improve the learning experience.

https://www.menti.com/7fpo8e3pa1

WHAT DID WE LEARN TODAY?

LINKED LIST

Traverse a list

linked_list.c

LINKED LIST

Insert anywhere

linked_list.c

REACH OUT

CONTENT RELATED QUESTIONS

Check out the forum

ADMIN QUESTIONS

cs1511@cse.unsw.edu.au