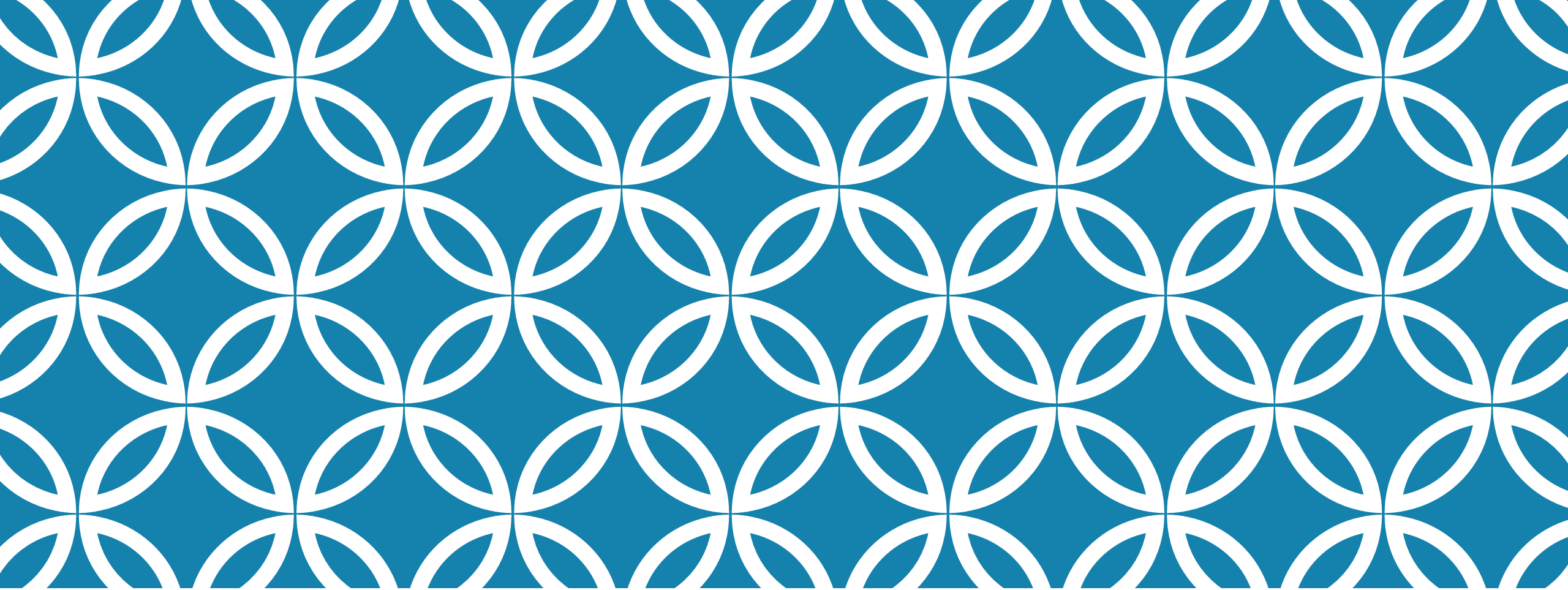
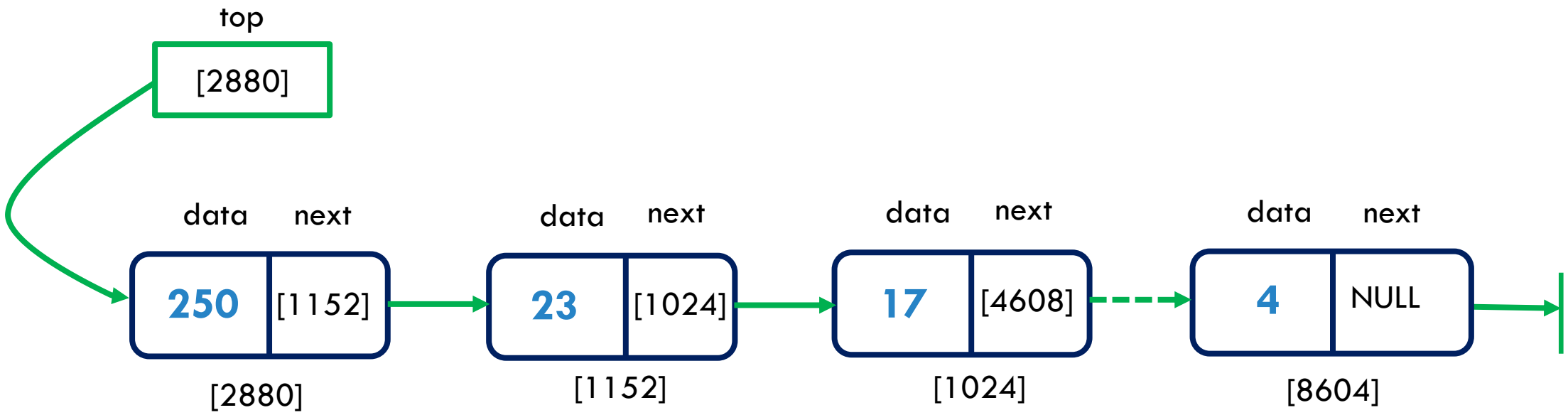


COMP 2611, Data Structures

LECTURE 3: ALGORITHM ANALYSIS & RECURSION WITH LINKED LISTS

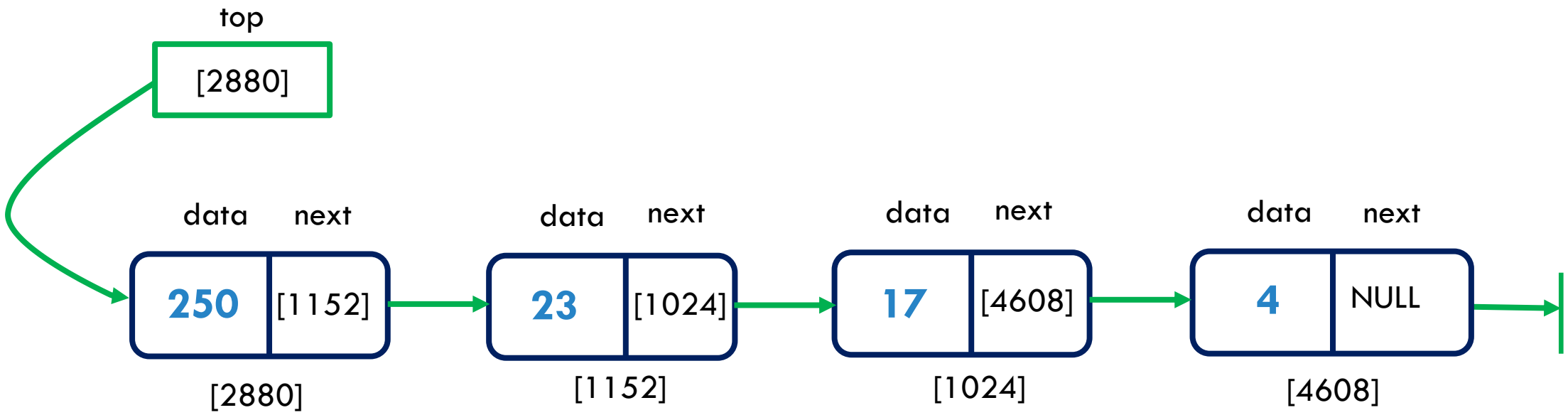
ANALYSIS OF LINKED LIST ALGORITHMS



Suppose that the linked list has *n* nodes:

- How long does it take to search for an element?
- How long does it take to insert an element at the head?
- How long does it take to insert an element at the end?
- How long does it take to delete an element?

RECURSION WITH LINKED LISTS



There are typically two cases:

The list is empty. → What to do?

The list has at least one element. → What to do?

AN EMPTY LIST: WHAT TO DO?

It depends on the problem:

- If you are printing the elements of a list:

  Do nothing and return.

- If you are calculating the sum of the elements in the list:

  Do nothing and return 0.

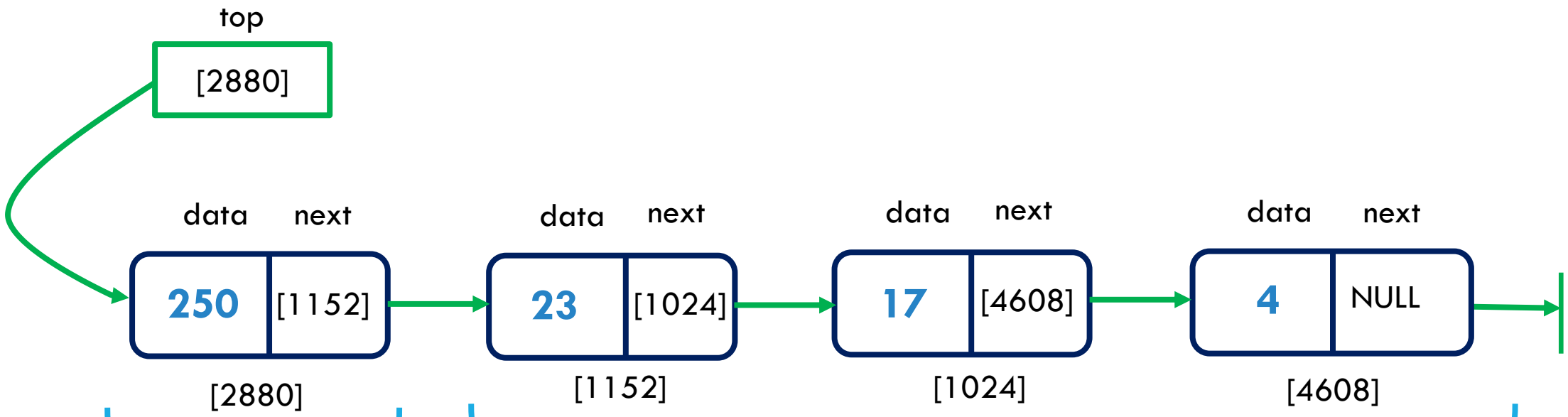
- If you are counting the number of elements in the list:

  Do nothing and return 0.

- If you are deleting elements from a list:

  Do nothing and return.

LIST HAS ≥ 1 ELEMENT: WHAT TO DO?



Process first element:

`top->data`

Process remaining elements:

`top->next`

Combine results in some way, if necessary

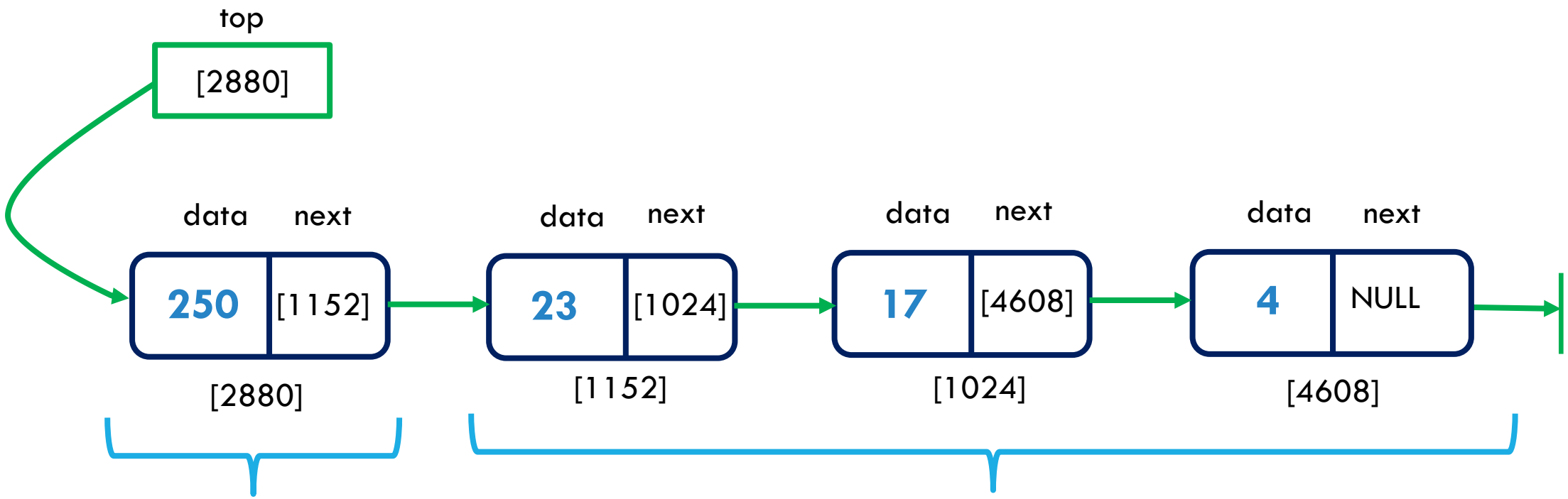
HOW TO PROCESS REMAINING ELEMENTS?

Call the function recursively with the shorter list (top->next). So, if we are printing the values in a list using the function *printListRec*, call *printListRec* with the shorter list:

```
printListRec (top->next);
```

```
void printListRec (Node * top);
    // prints the elements of a linked list
```

250



print first element:

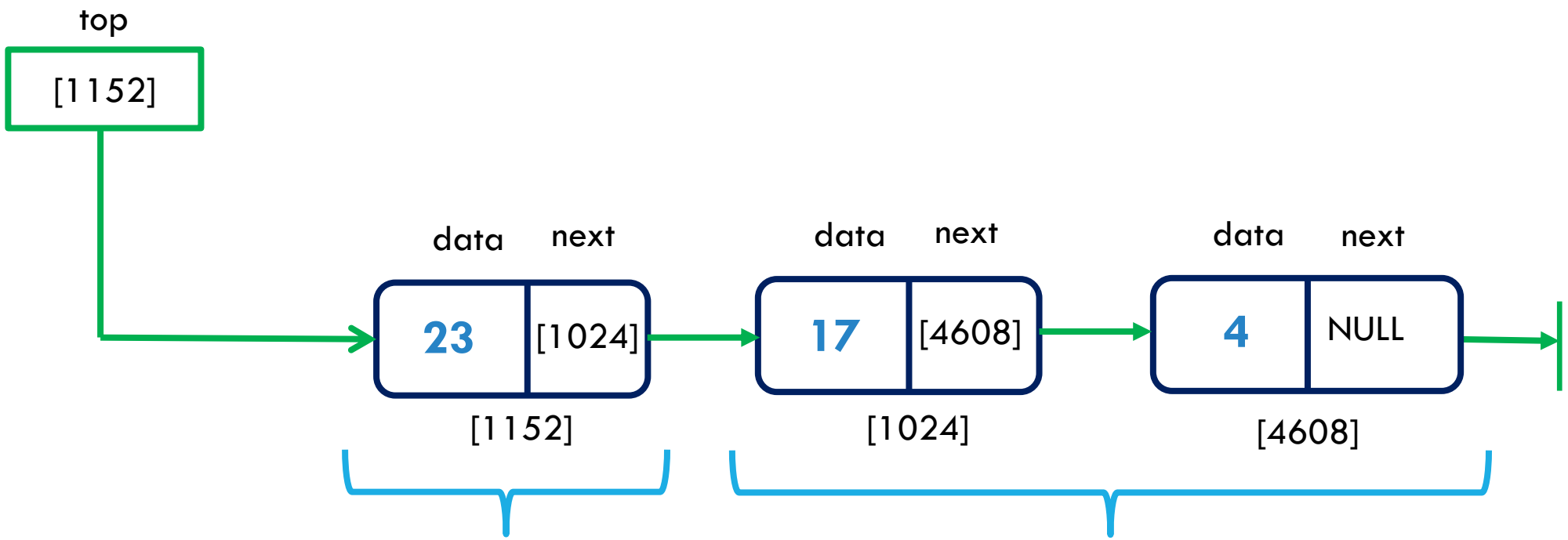
```
cout << top->data;
```

print remaining elements:

```
printListRec(top->next);
```

250 23

RECURSIVE CALL (#1)



print first element:

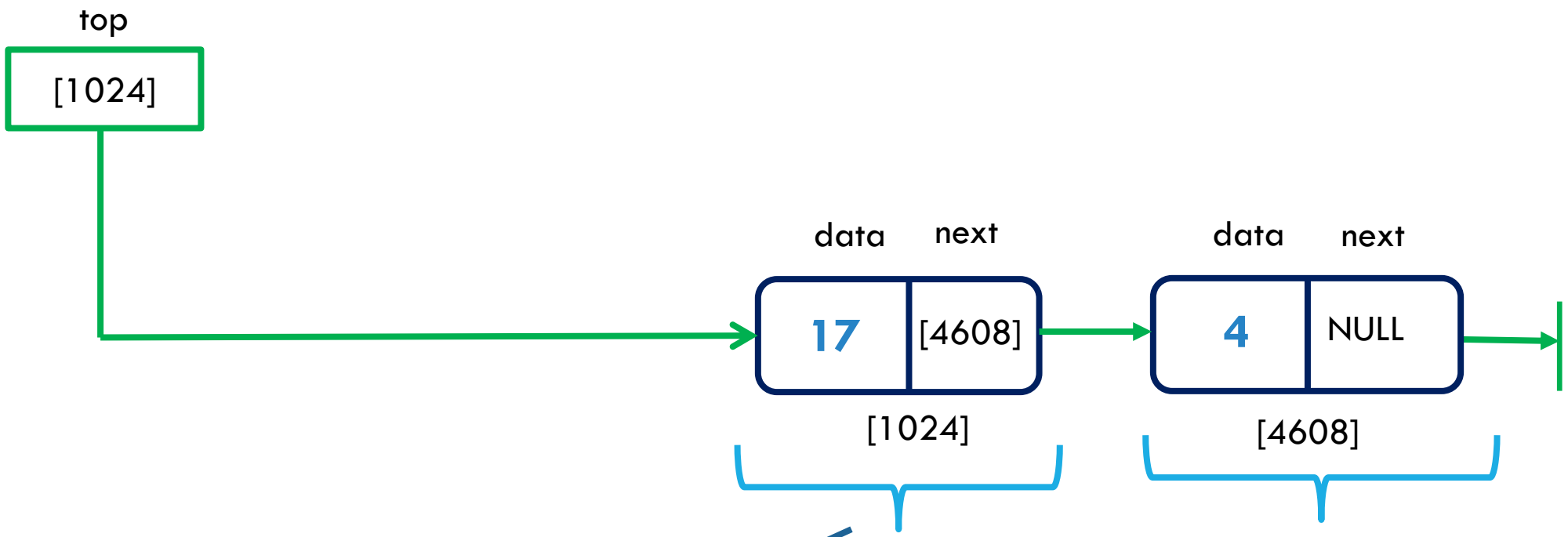
```
cout << top->data;
```

print remaining elements:

```
printListRec(top->next);
```

250 23 17

RECURSIVE CALL (#2)



print first element:

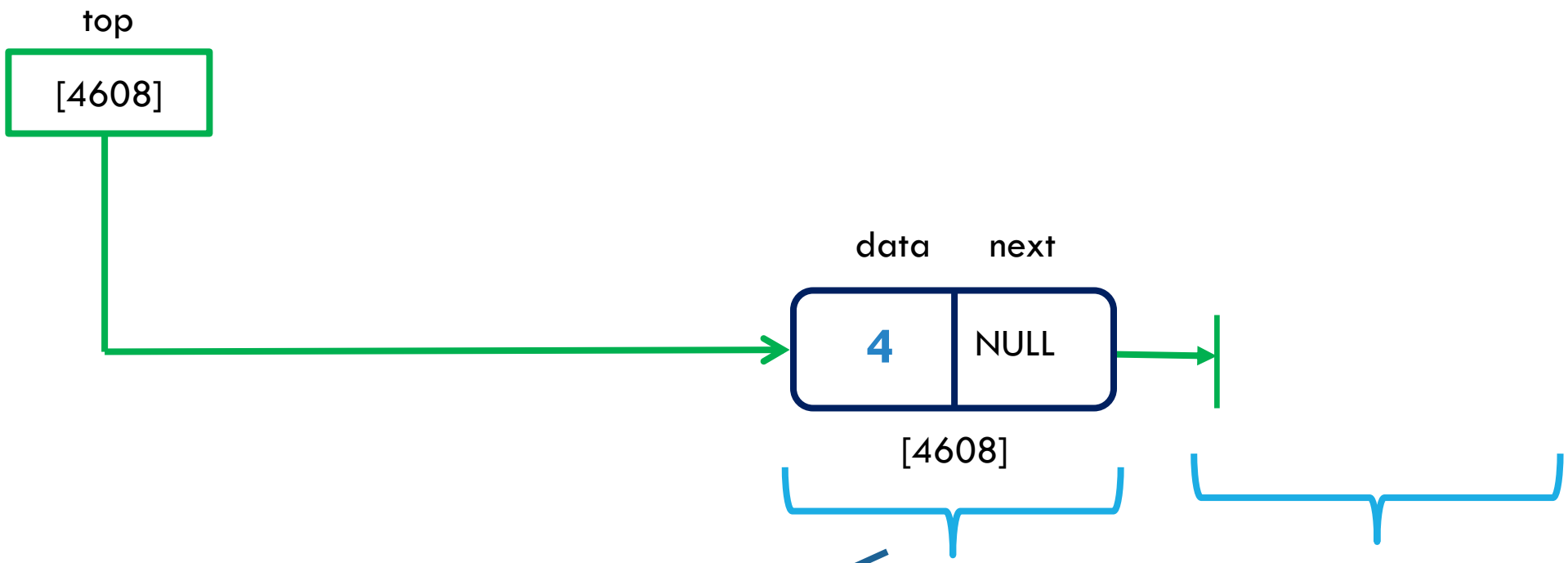
```
cout << top->data;
```

print remaining elements:

```
printListRec(top->next);
```

250 23 17 4

RECURSIVE CALL (#3)



print first element:

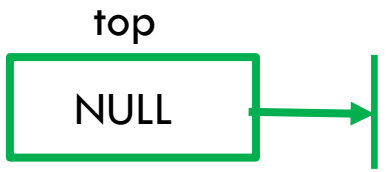
```
cout << top->data;
```

print remaining elements:

```
printListRec(top->next);
```

250 23 17 4

RECURSIVE CALL (#4)



Since the list is empty,
do nothing and return.

```
void printListRec (Node * top) {

    if (top == NULL)
        return;

    cout << top->data << "\t";

    printListRec (top->next);

}
```