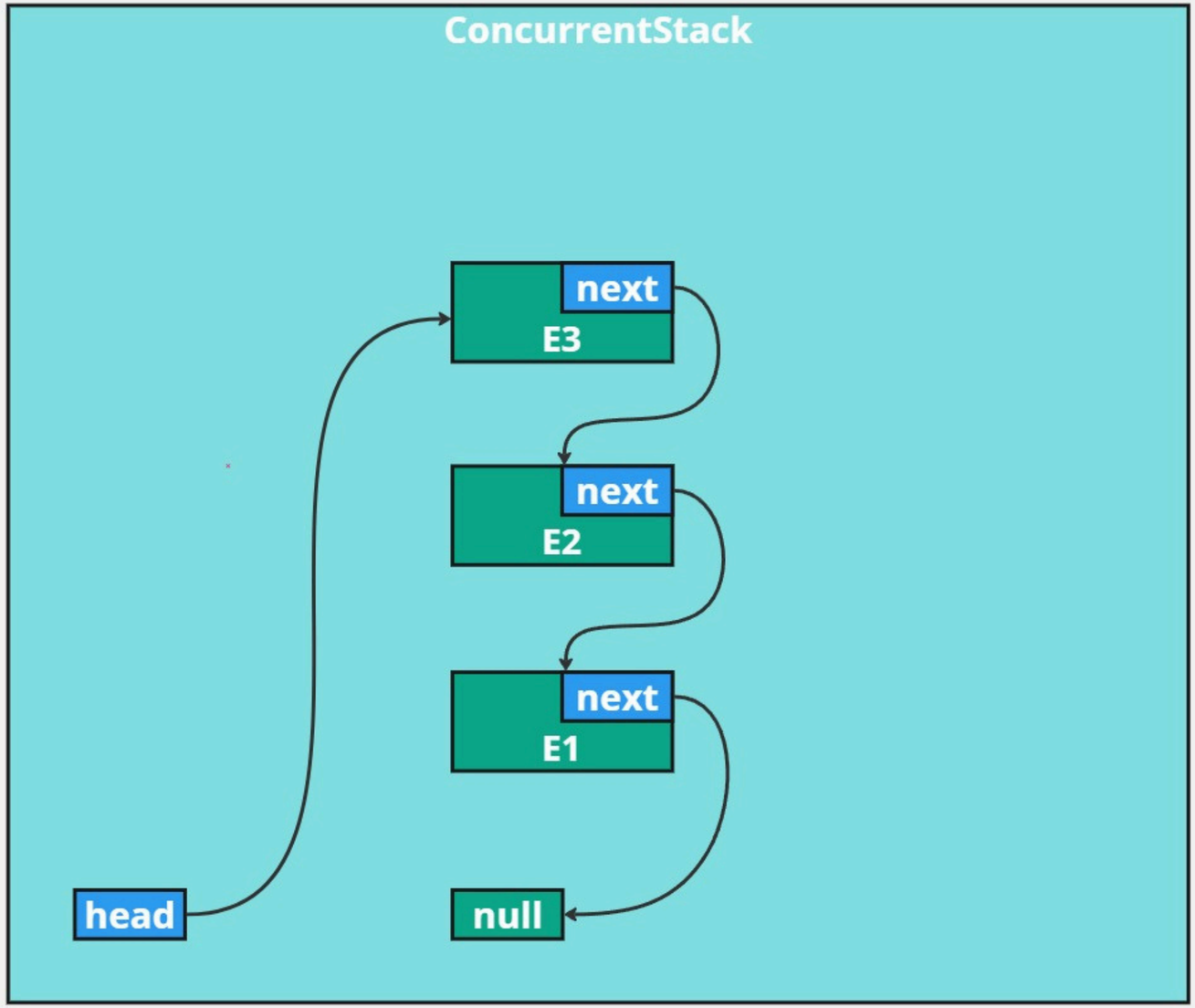
JAVA

МНОГОПОТОЧНОСТЬ.

Стек Трайбера



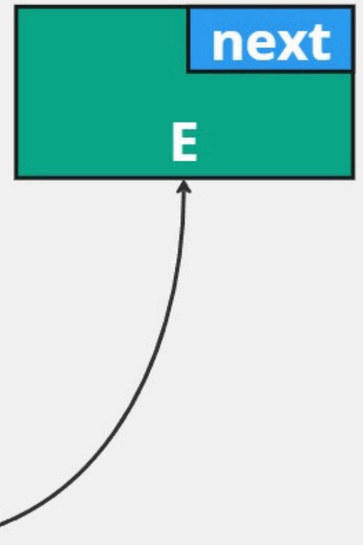
synchronized

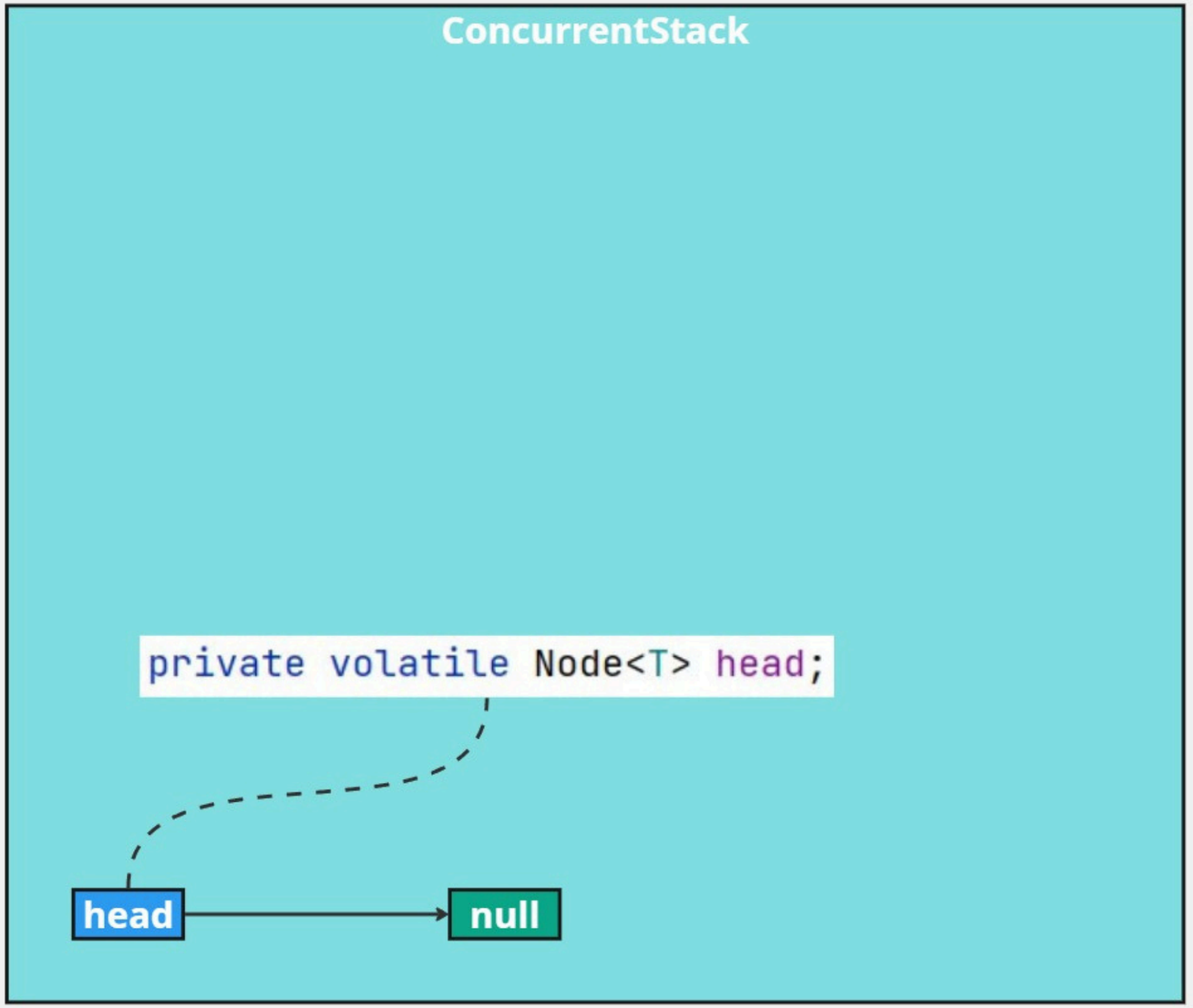
volatile

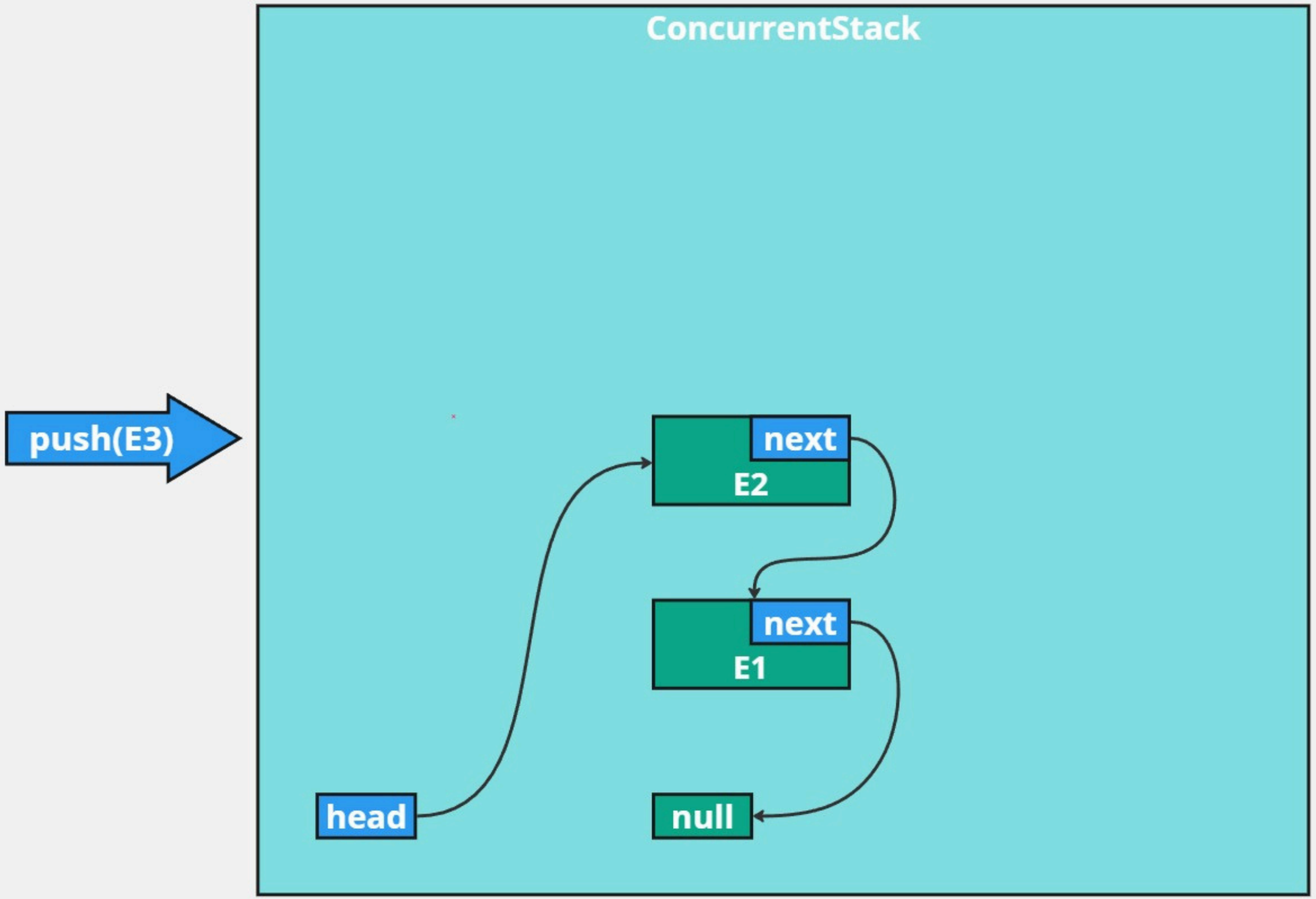
ReentrantLock

AtomicReference

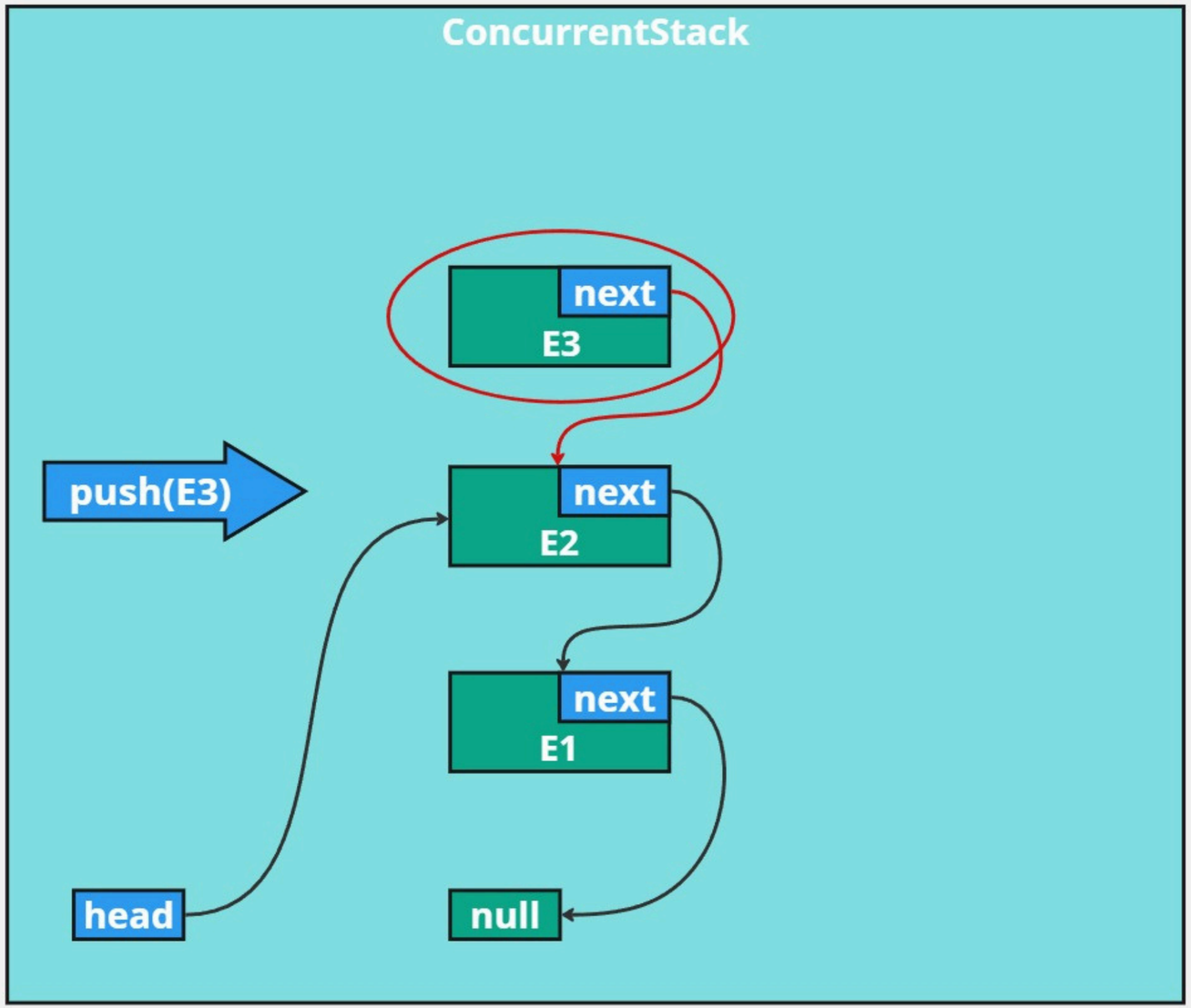
```java
private static final class Node<T> {
    private final T value;
    private final Node<T> next;

    public Node(final T value, final Node<T> next) {
        this.value = value;
        this.next = next;
    }
}
```



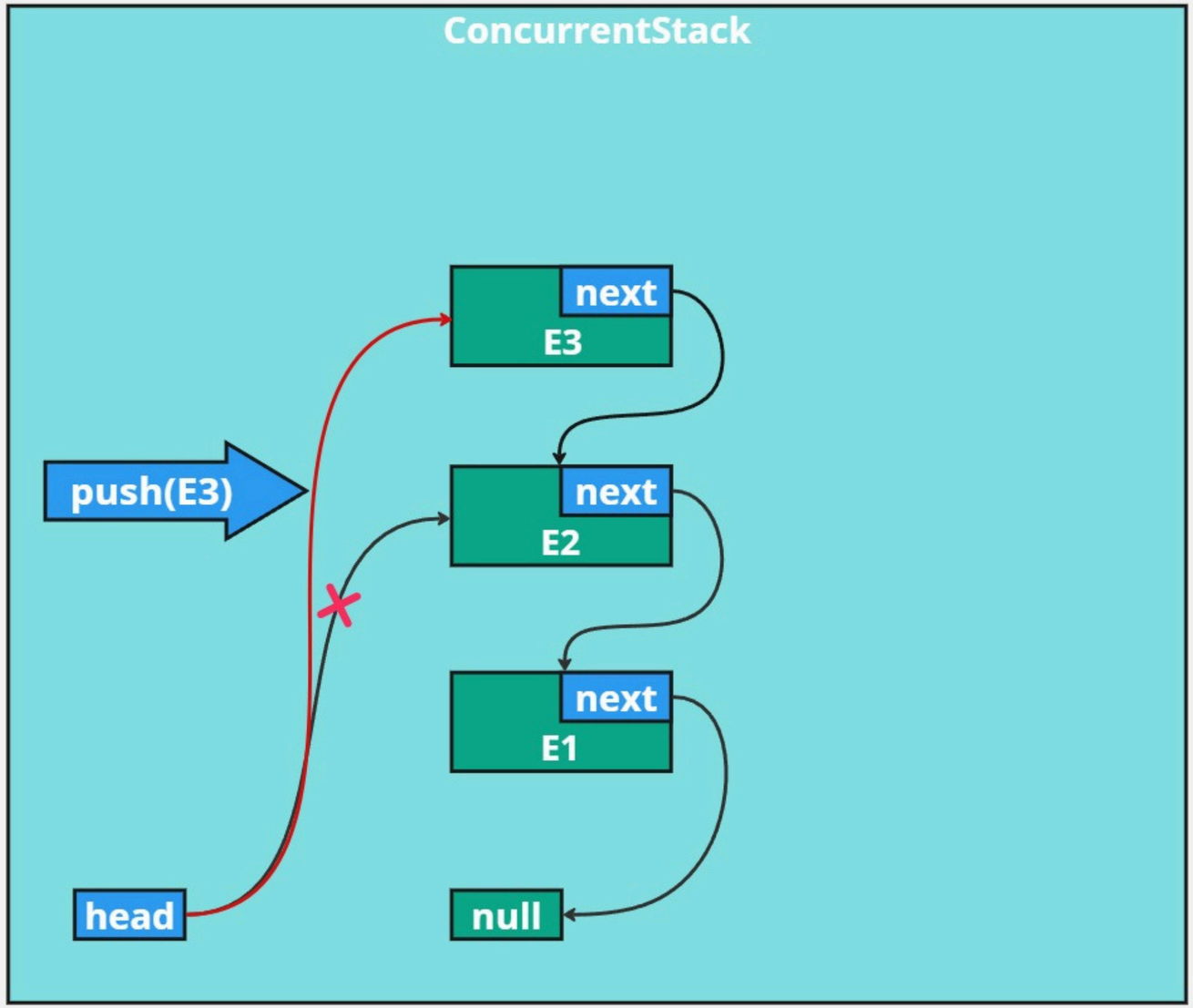

ConcurrentStack
private volatile Node<T> head;
head
null



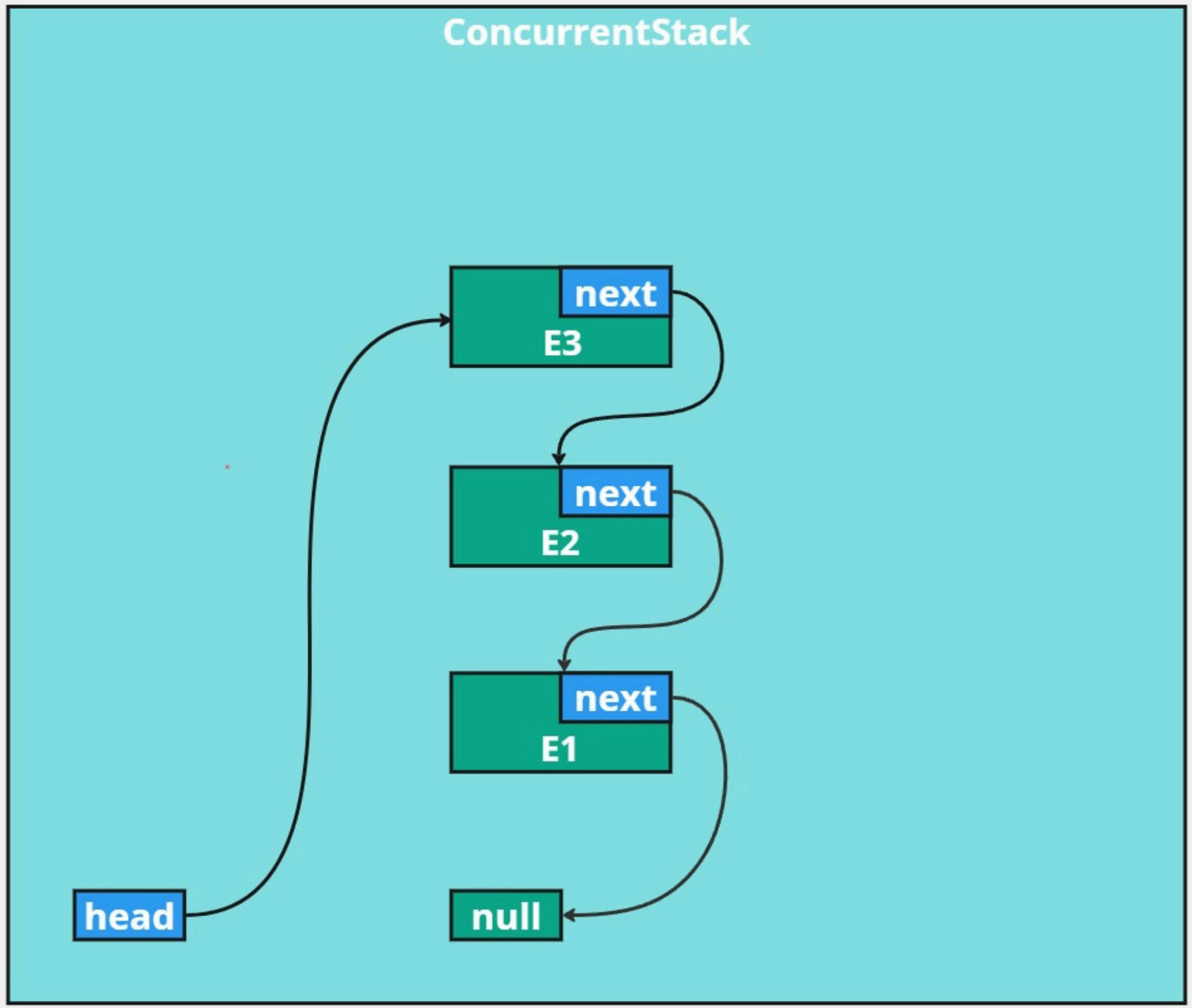

```
public void push(final T element) {
    //create new node
    //head should be assigned new node
}
```

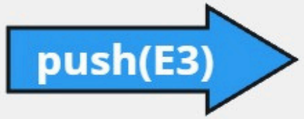



```
public void push(final T element) {
    //create new node
    //head should be assigned new node
}
```

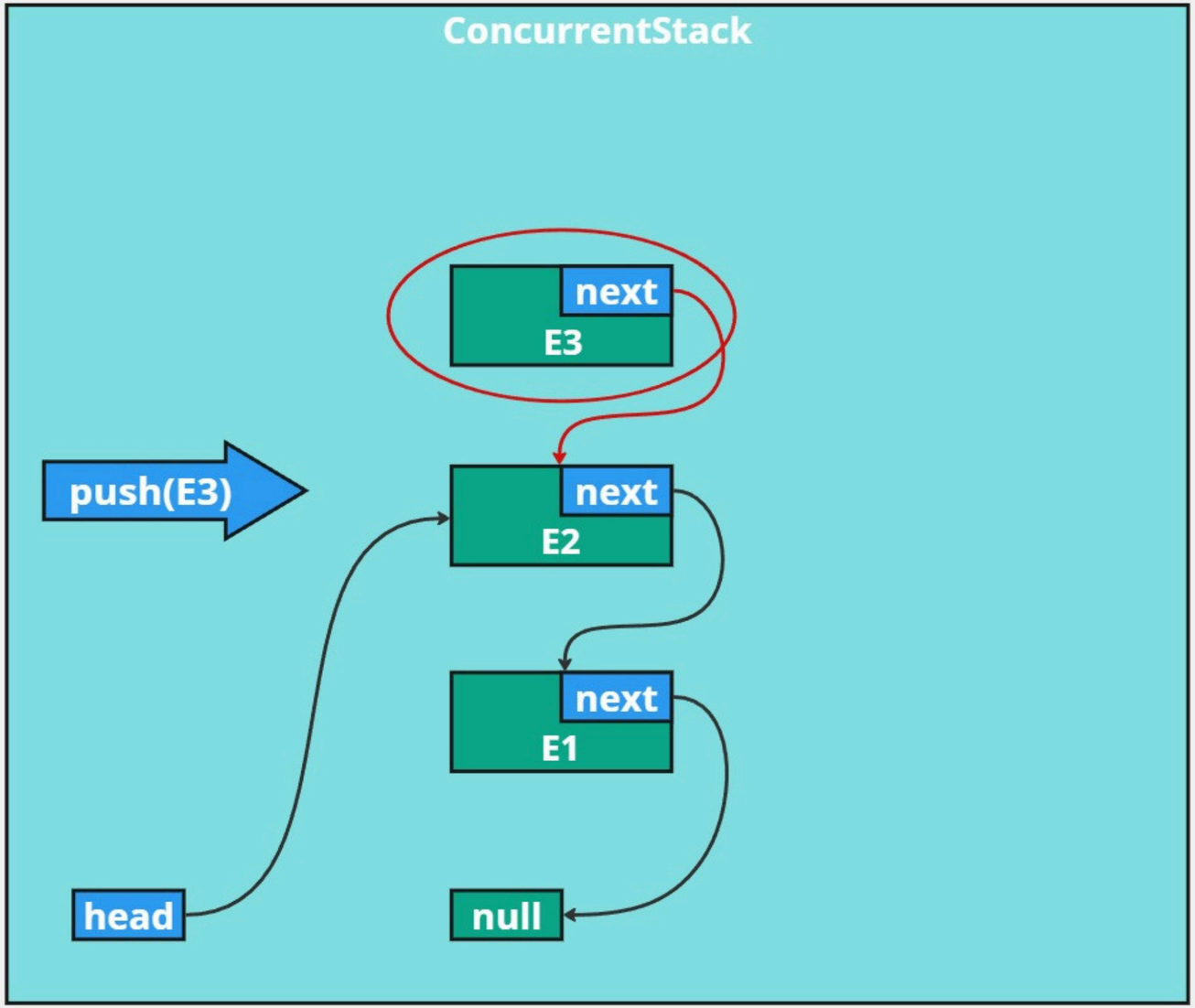



```
public void push(final T element) {
    //create new node
    //head should be assigned new node
}
```

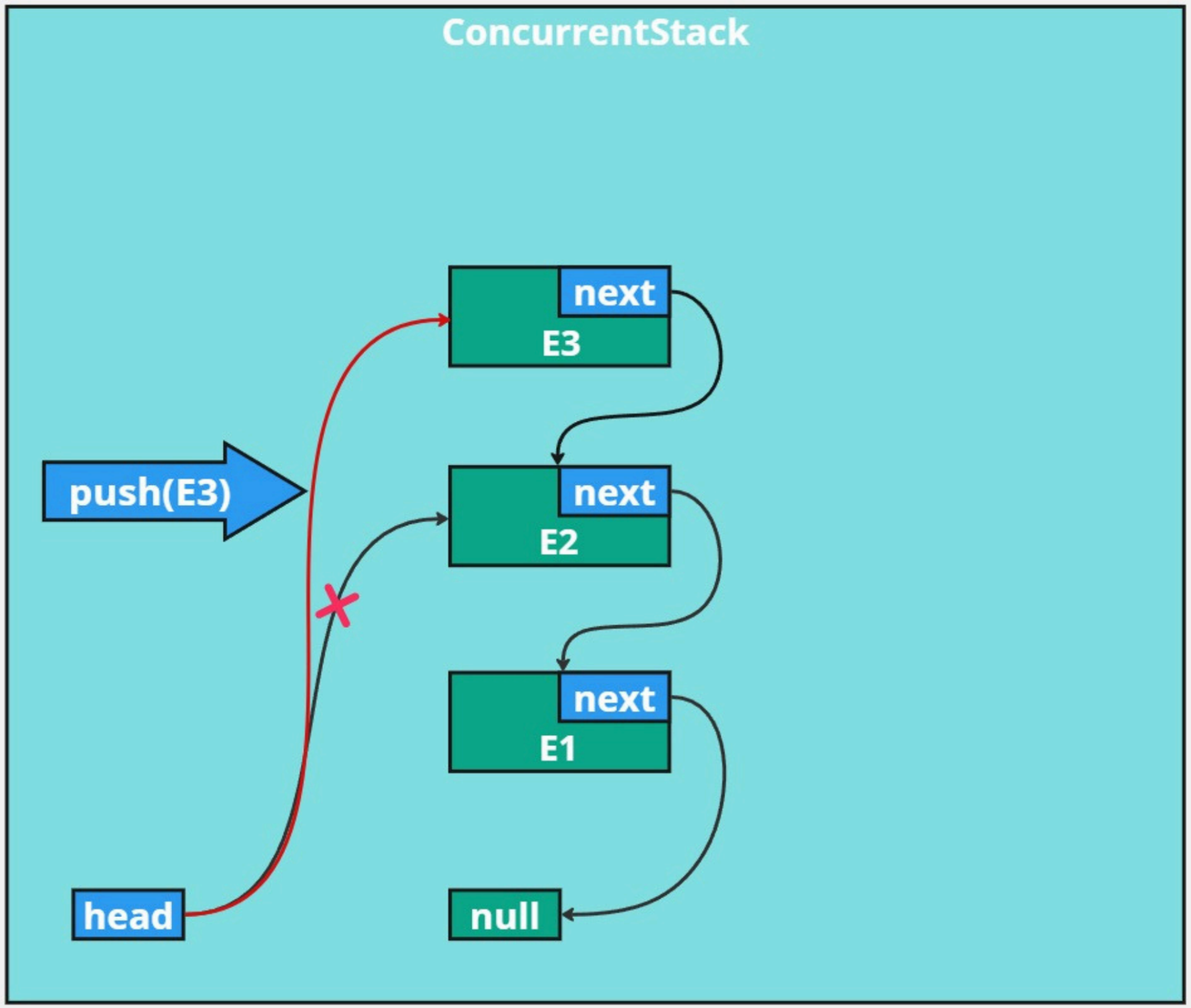




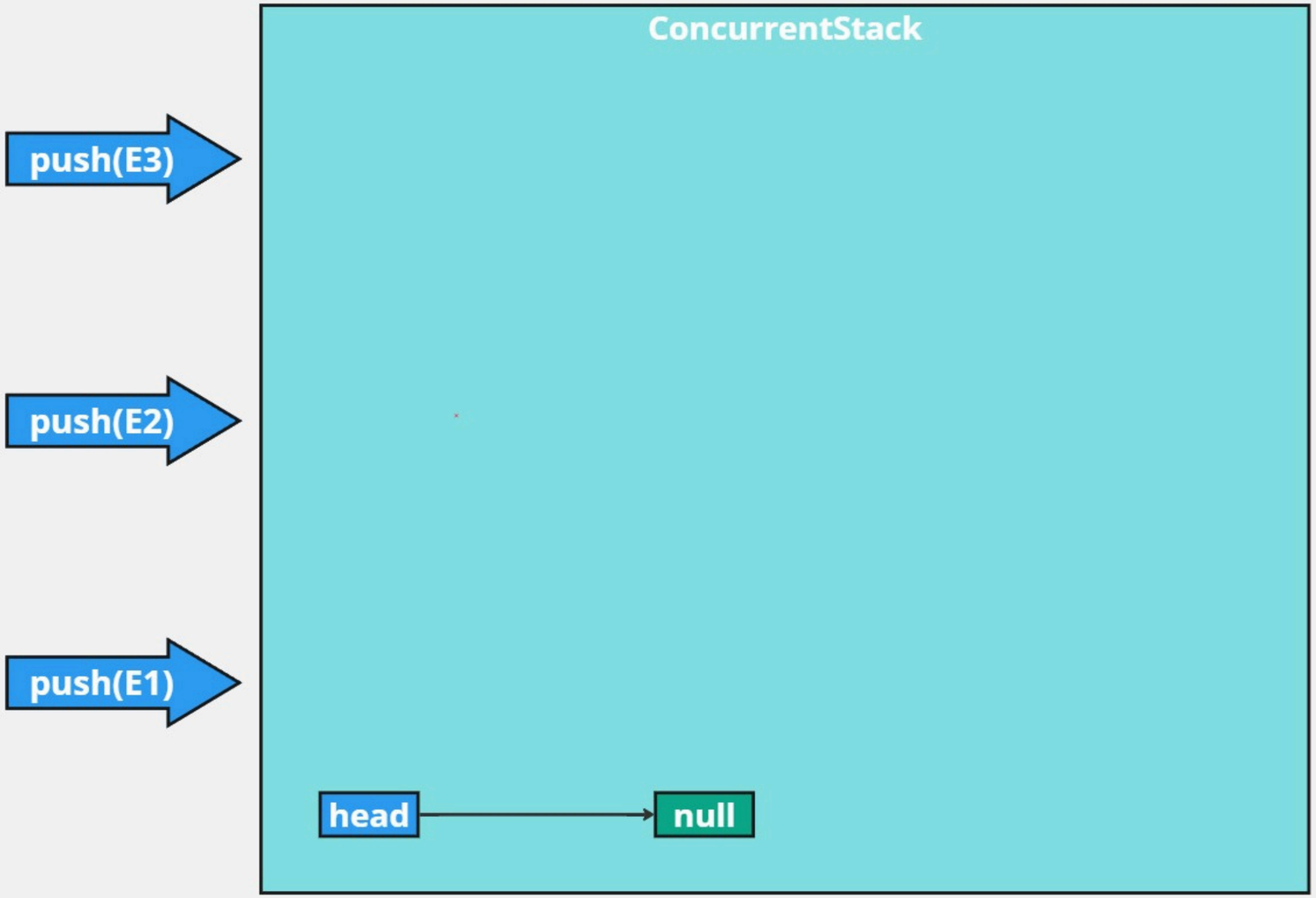

```
public void push(final T element) {
    //create new node
    //head should be assigned new node
}
```




```
public void push(final T element) {
    head = new Node<>(element, head);
}
```




```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E1)

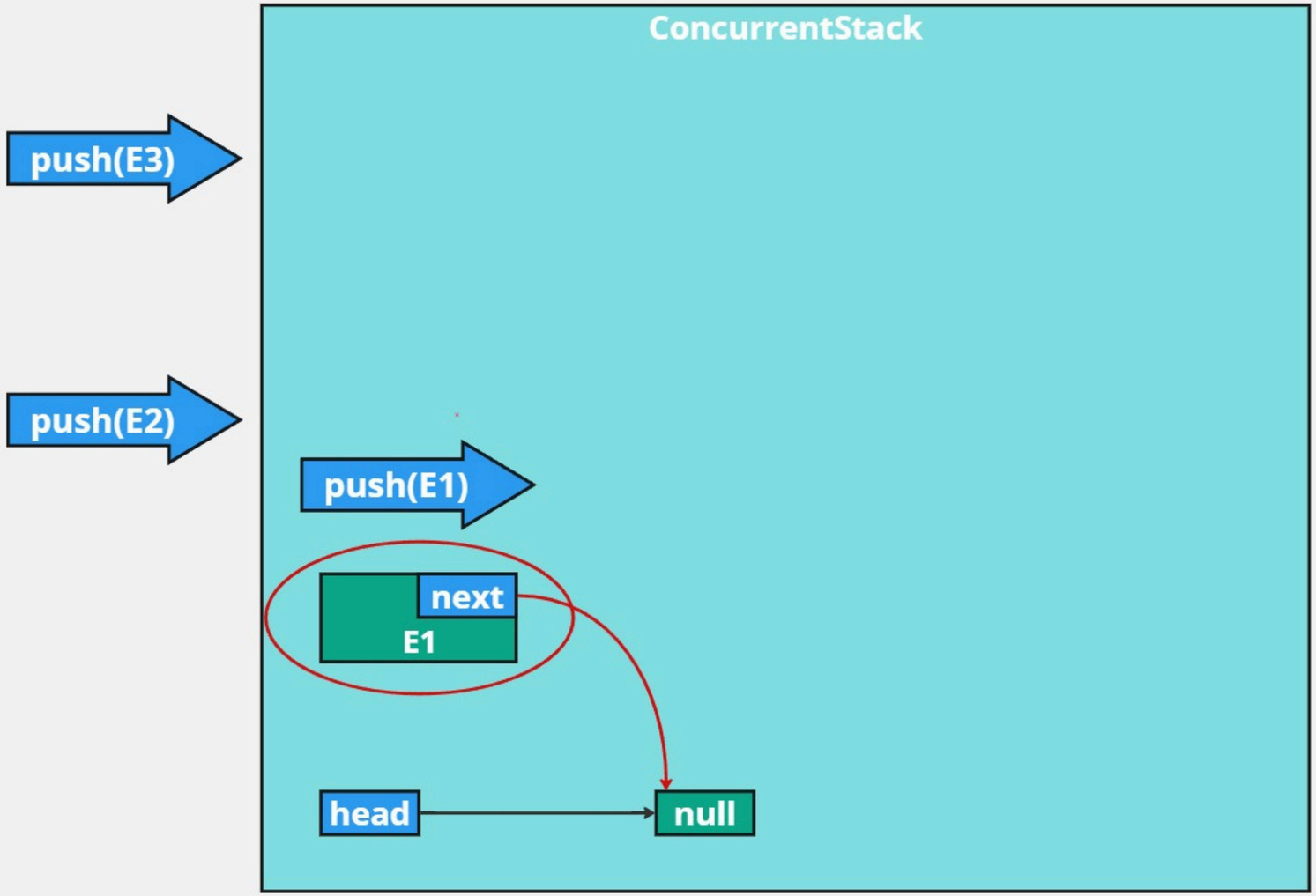
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E1)

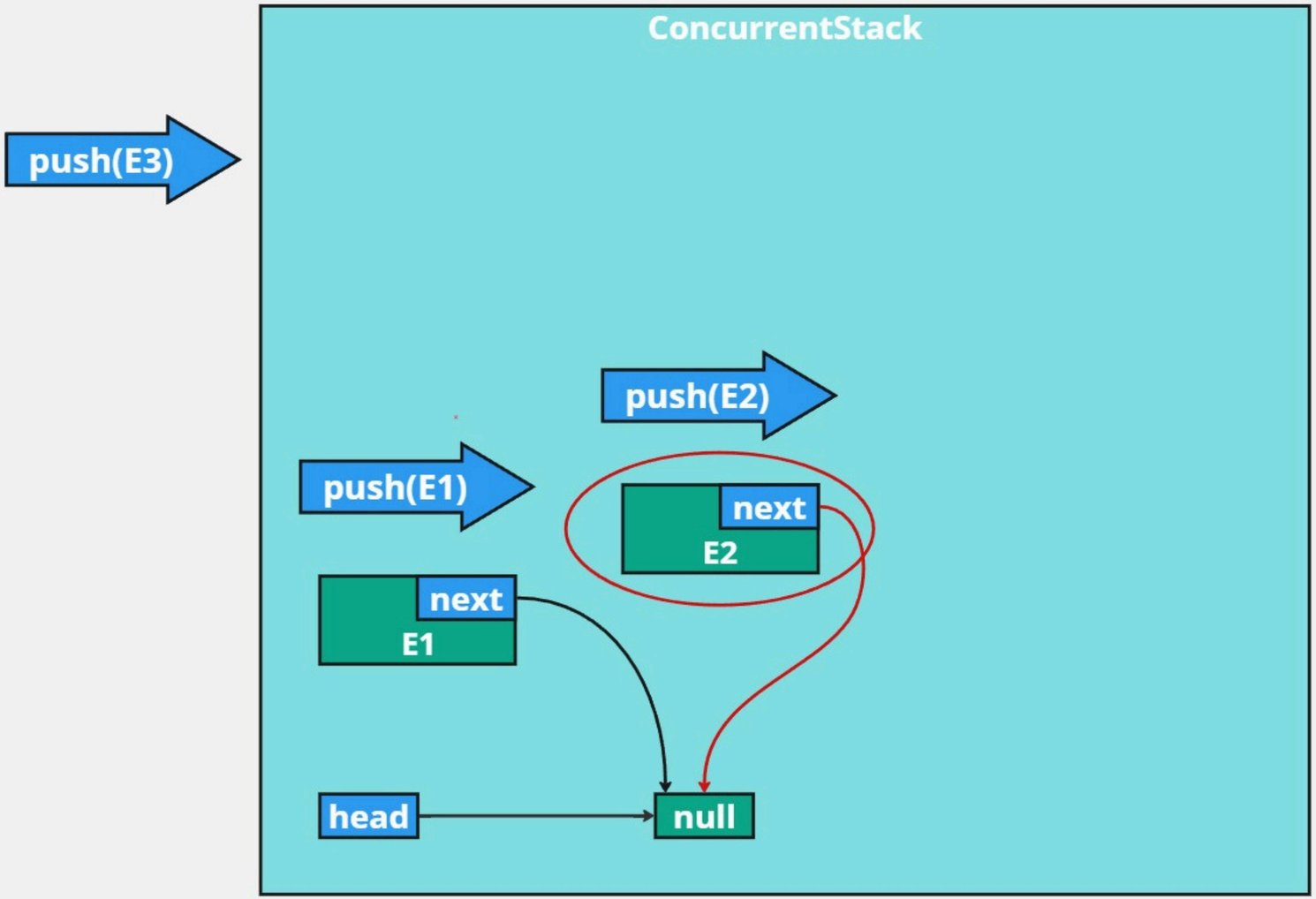
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E1)

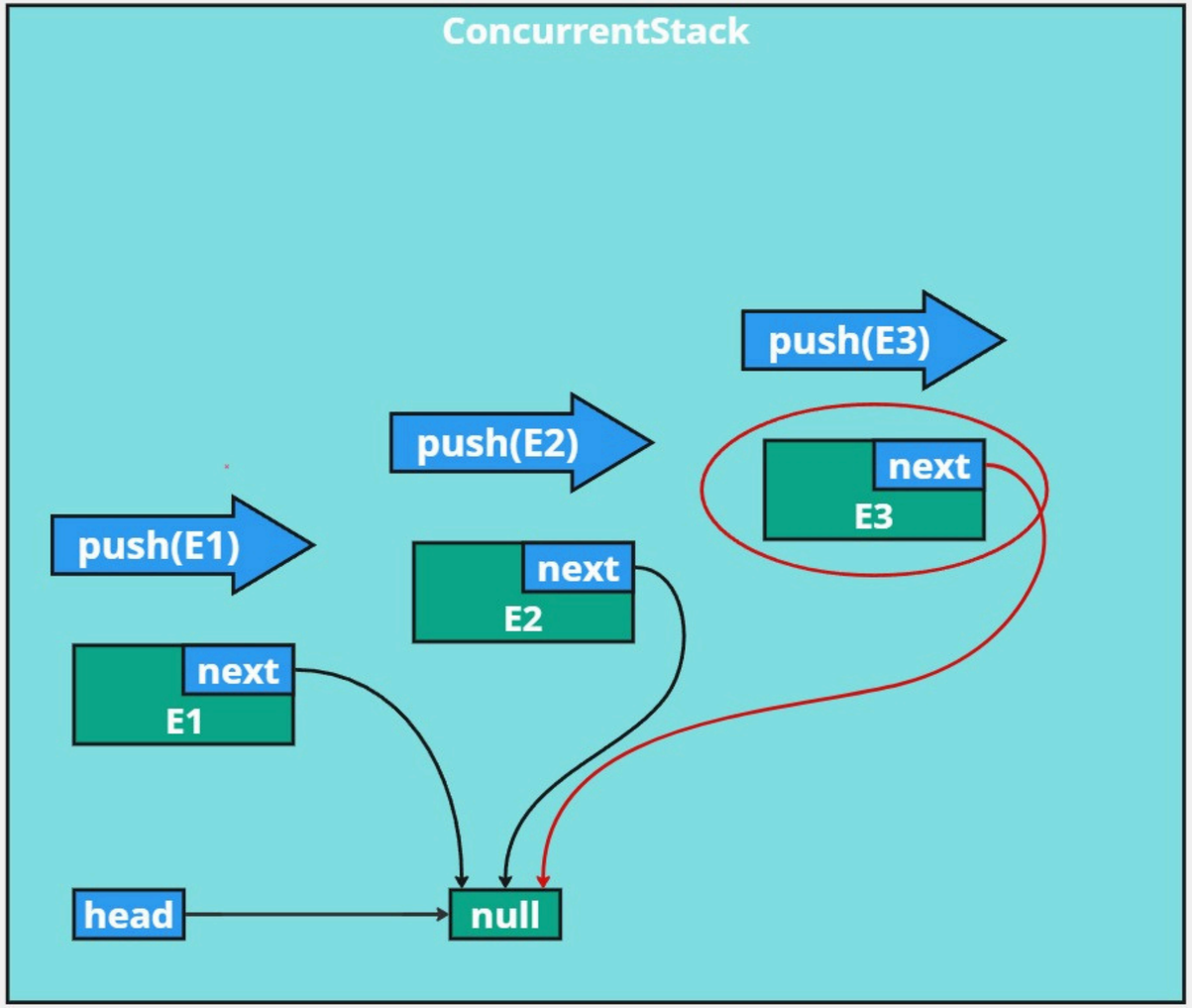
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E1)

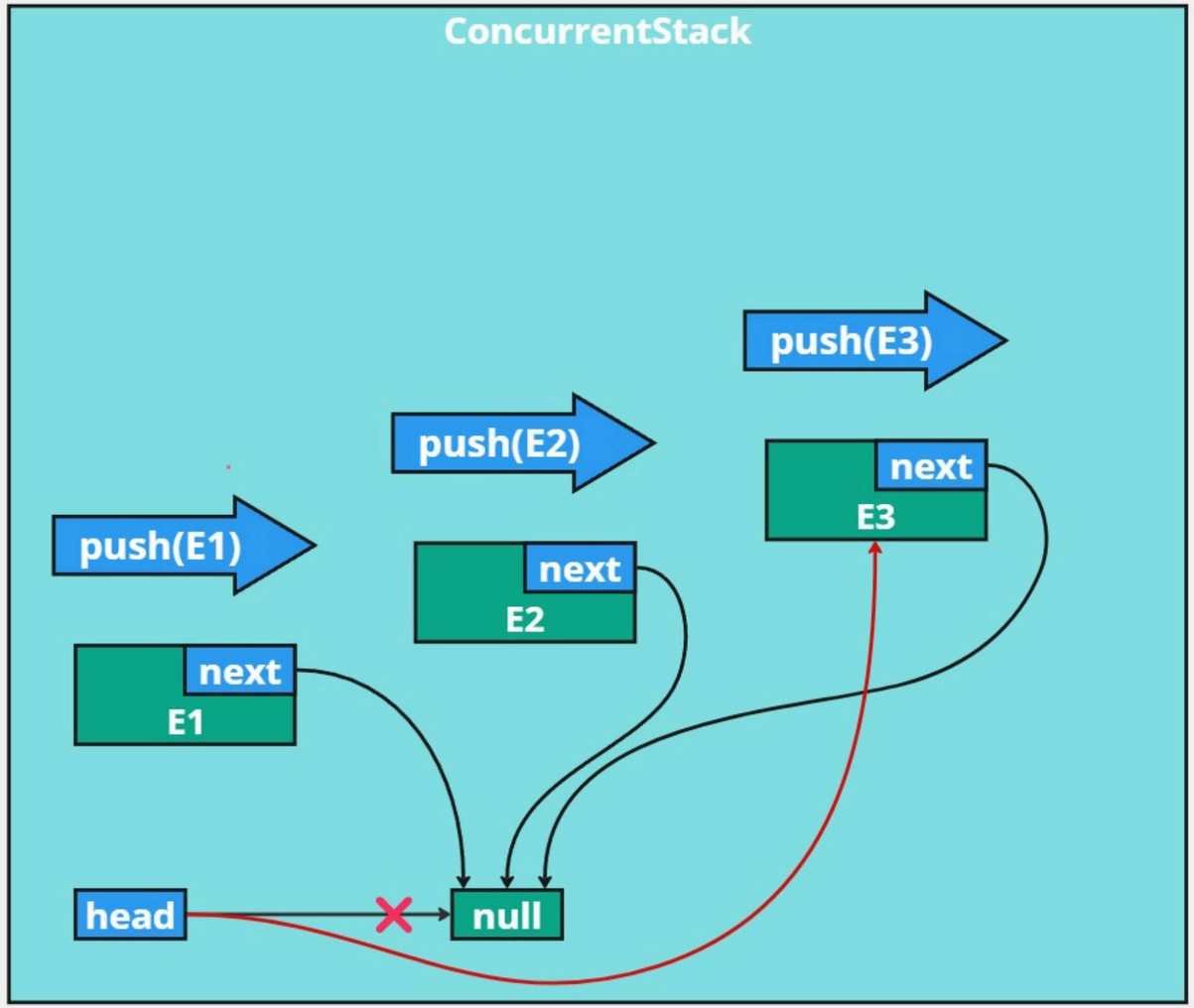
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E1)

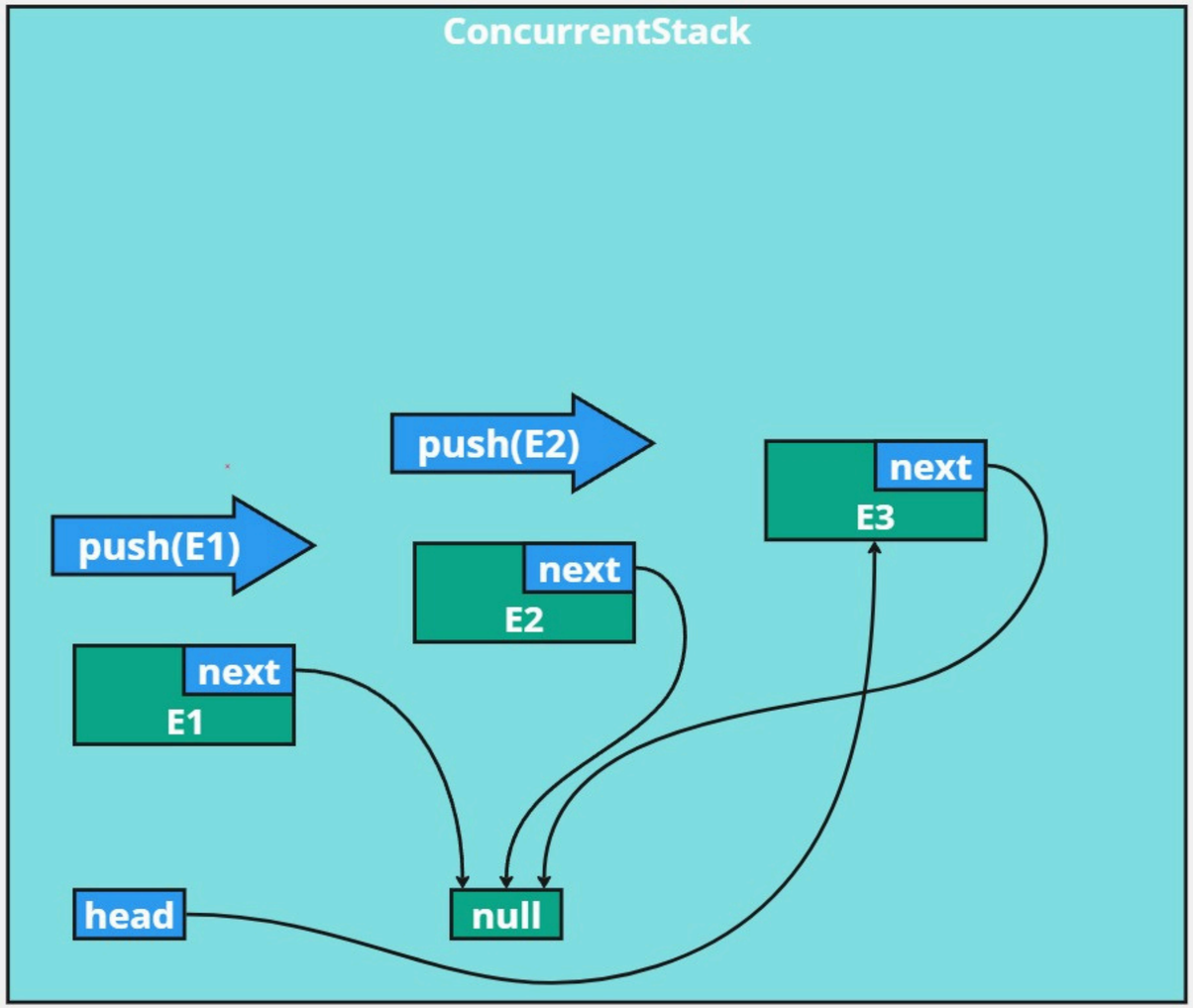
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E3)

push(E1)

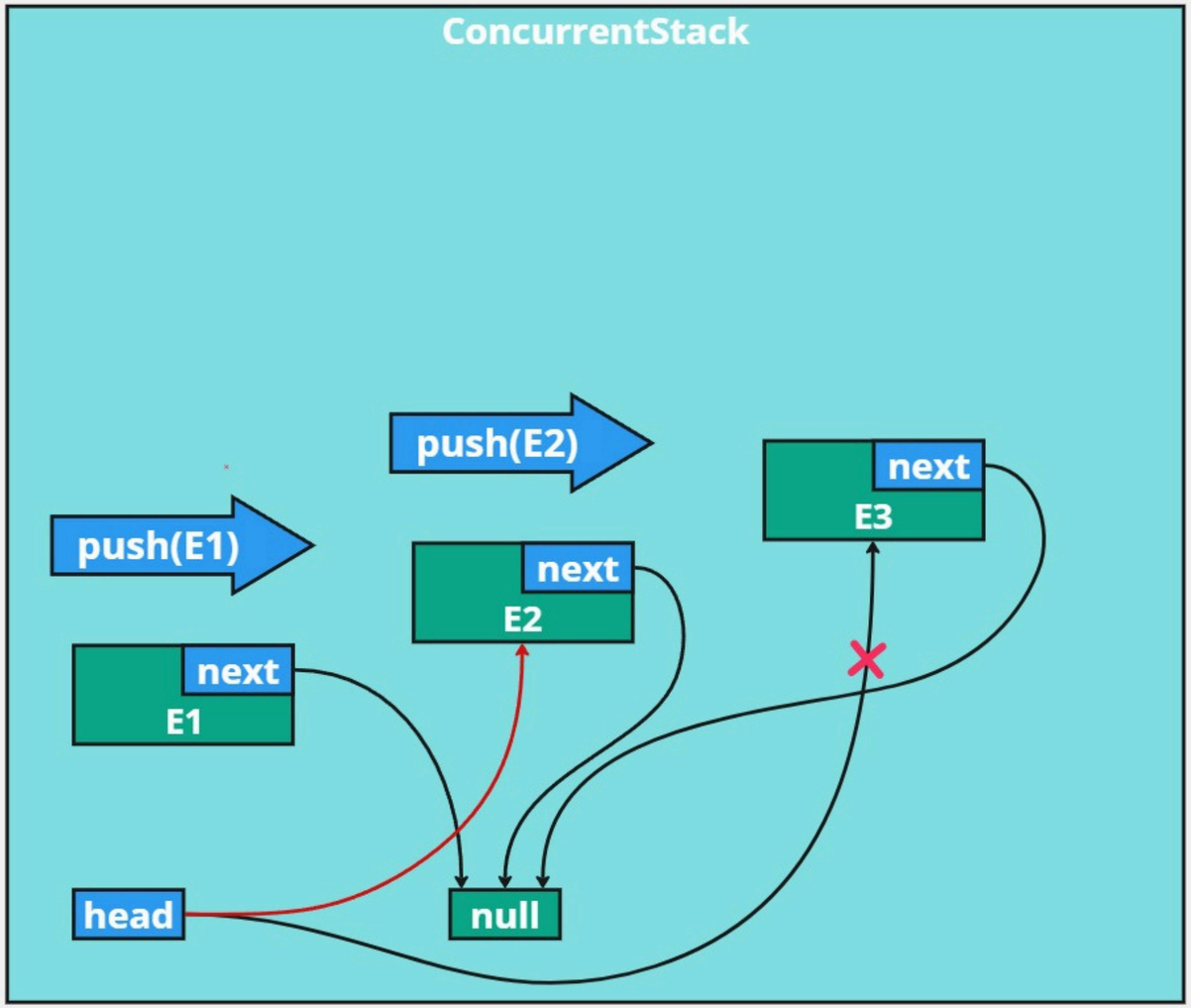
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E3)

push(E1)

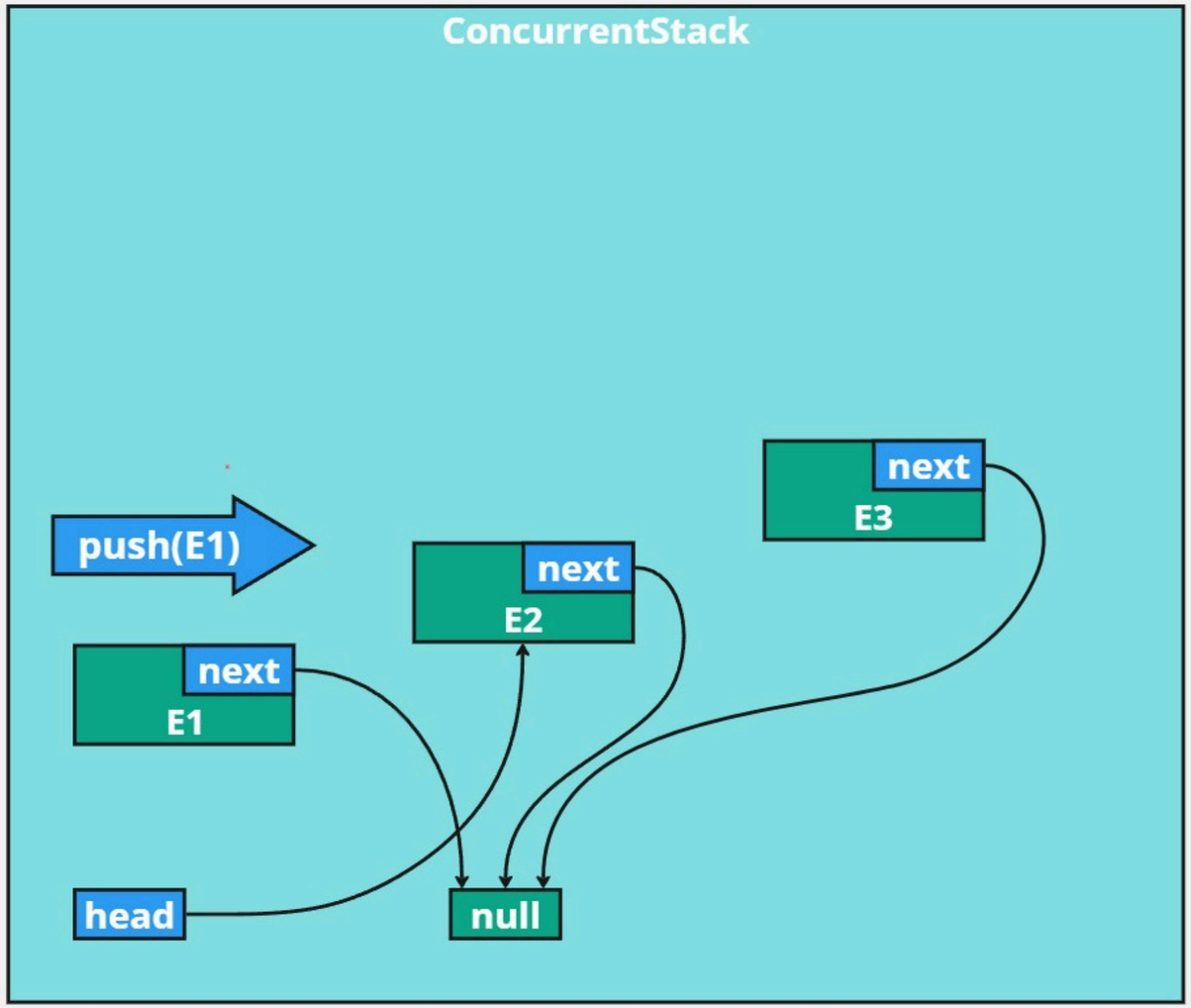
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E2)

push(E3)

push(E1)

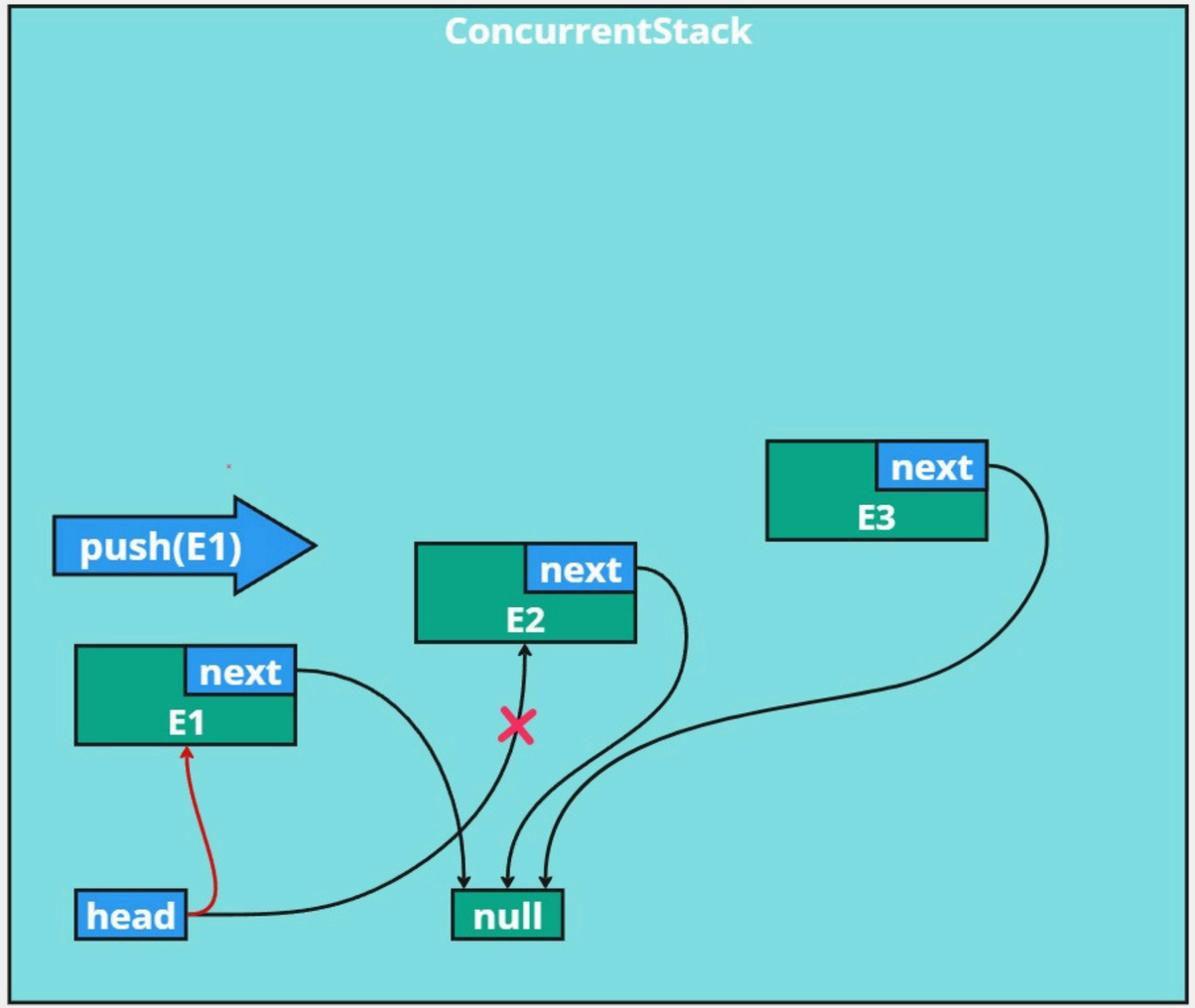
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E2)

push(E3)

push(E1)

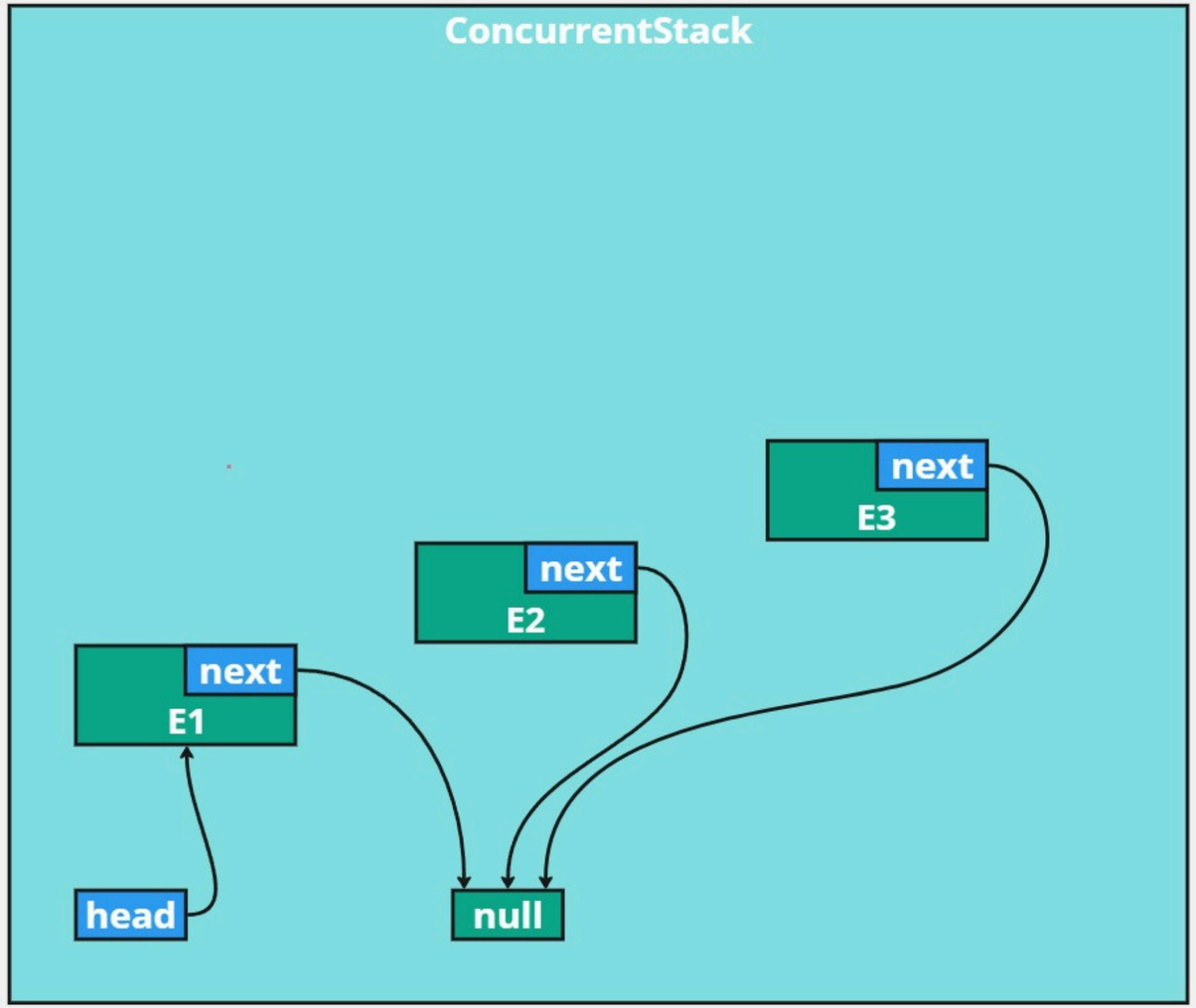
```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```



push(E1)

push(E1)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E2)

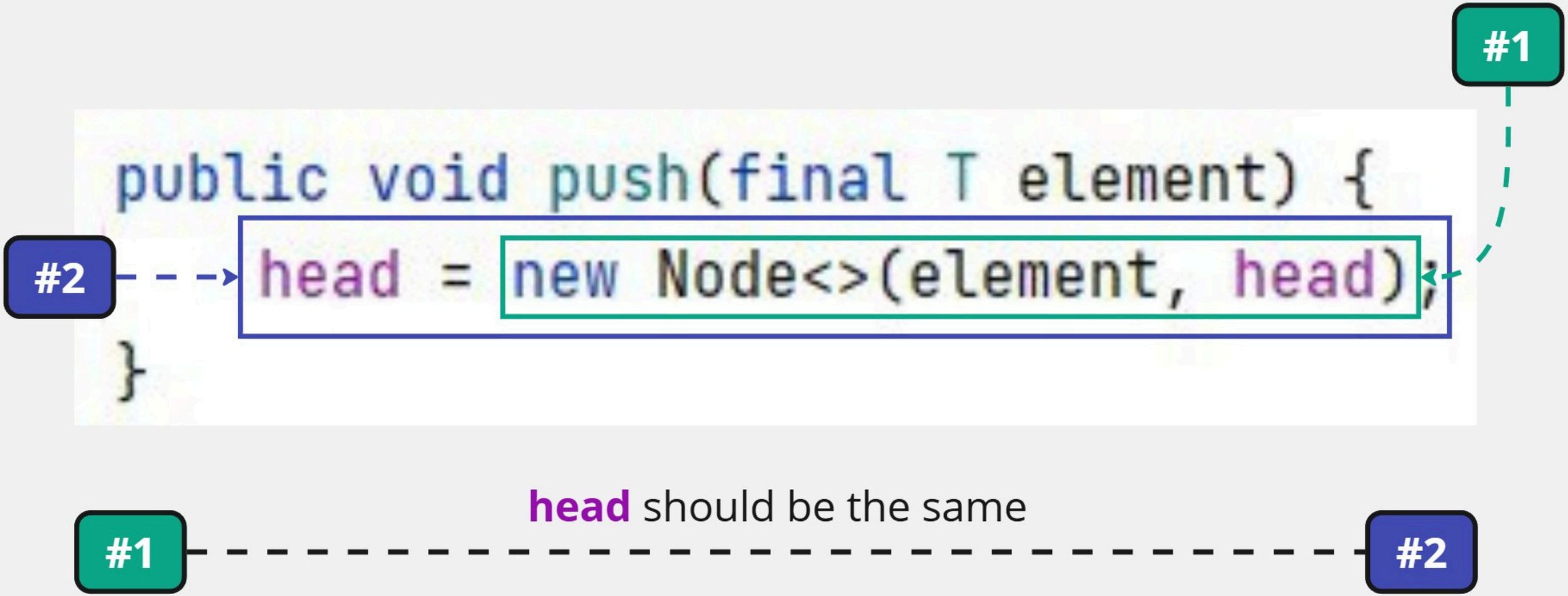
push(E2)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

push(E3)

push(E3)

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```


#1
public void push(final T element) {
#2
head = new Node<>(element, head);
}
head should be the same
#1
#2

```
public void push(final T element) {
    head = new Node<>(element, head);
}
```

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head;
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head == previousHead) {
            head = newNode;
            return;
        }
    }
}
```

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head;
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head == previousHead) {
            head = newNode;
            return;
        }
    }
}
```

not atomic

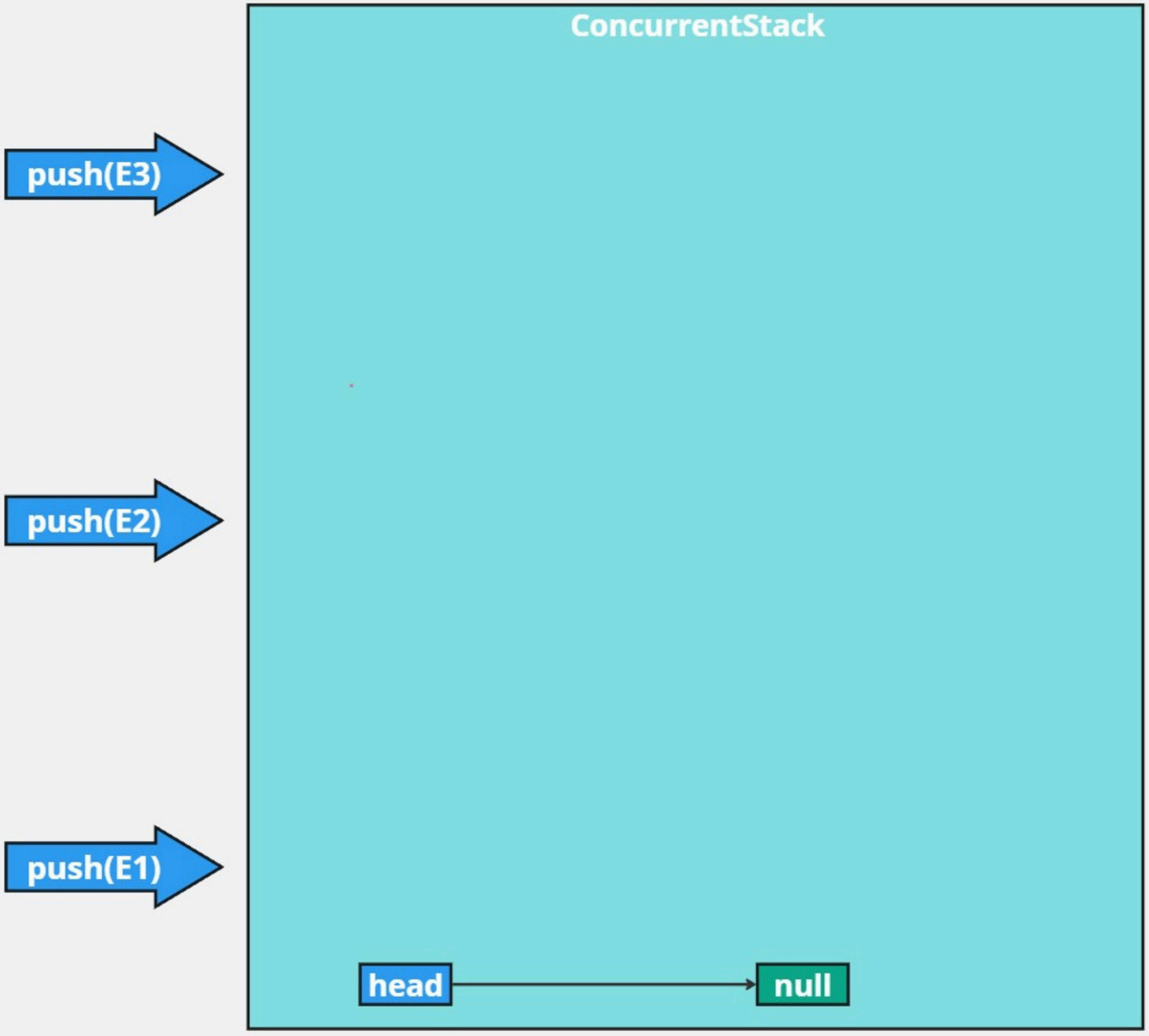
```
private volatile Node<T> head;
```

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head;
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head == previousHead) {
            head = newNode;
            return;
        }
    }
}
```

```
private final AtomicReference<Node<T>> head = new AtomicReference<>();
```

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

atomic



push(E1)

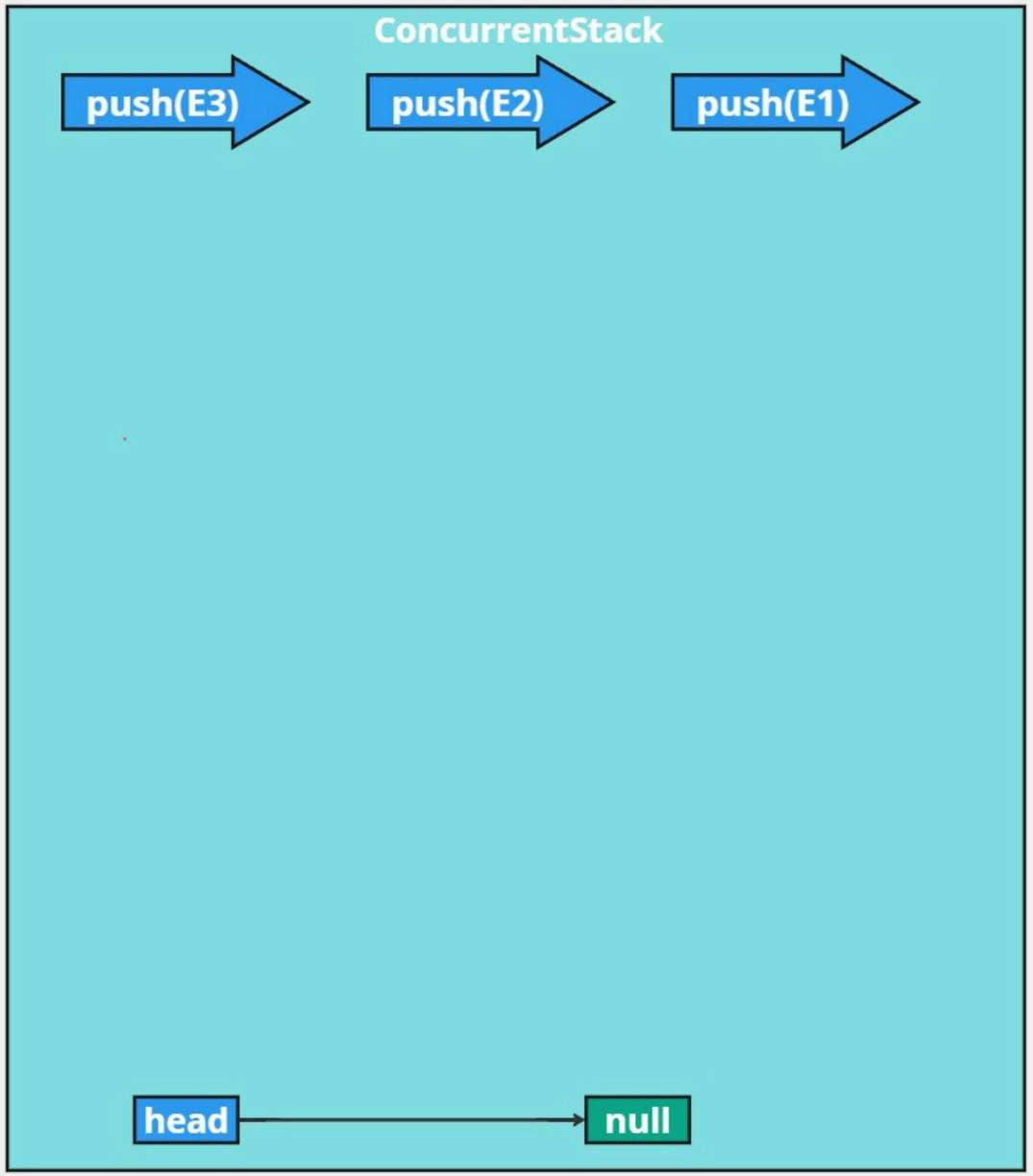
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```



push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E2)

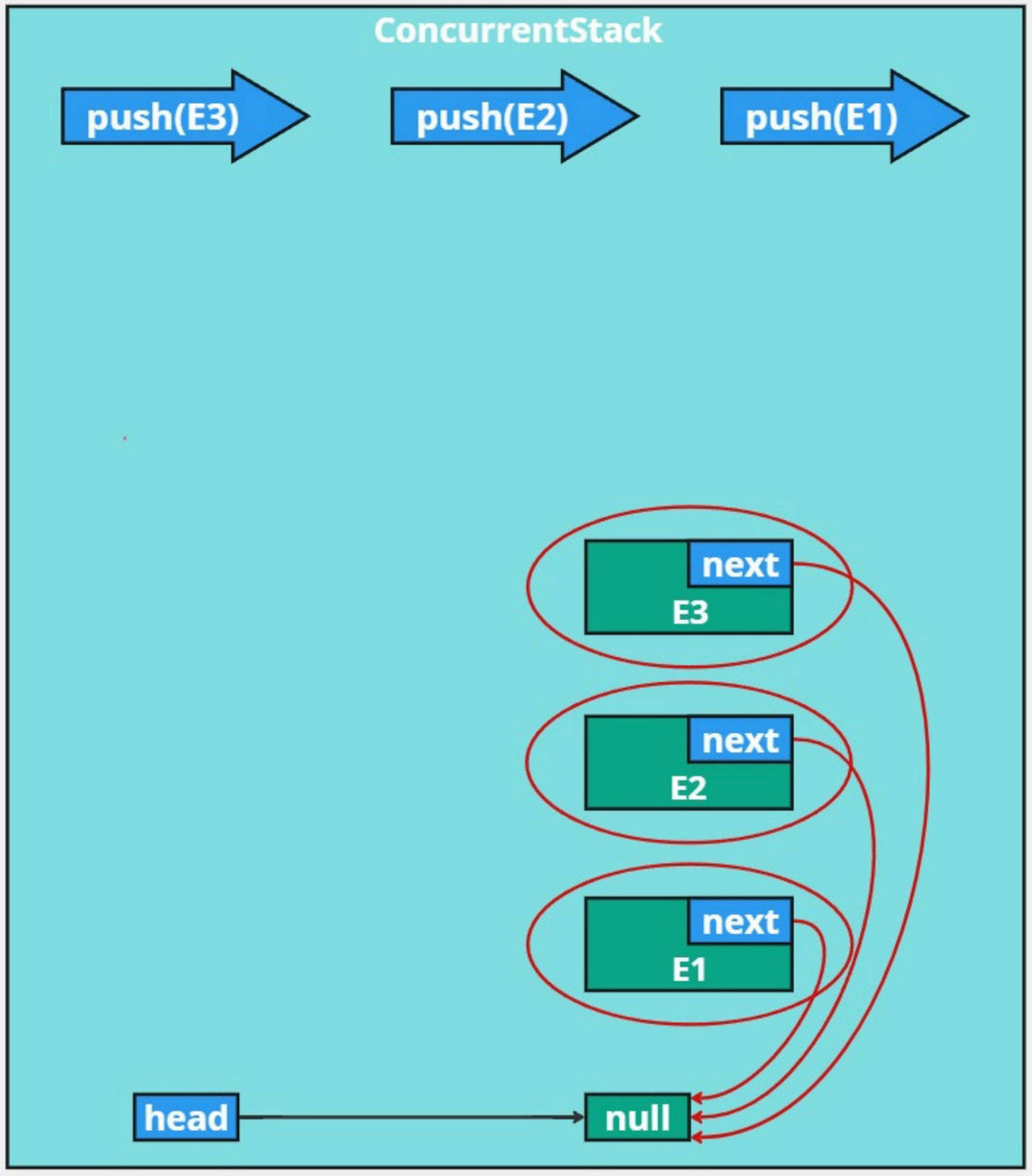
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null



push(E1)

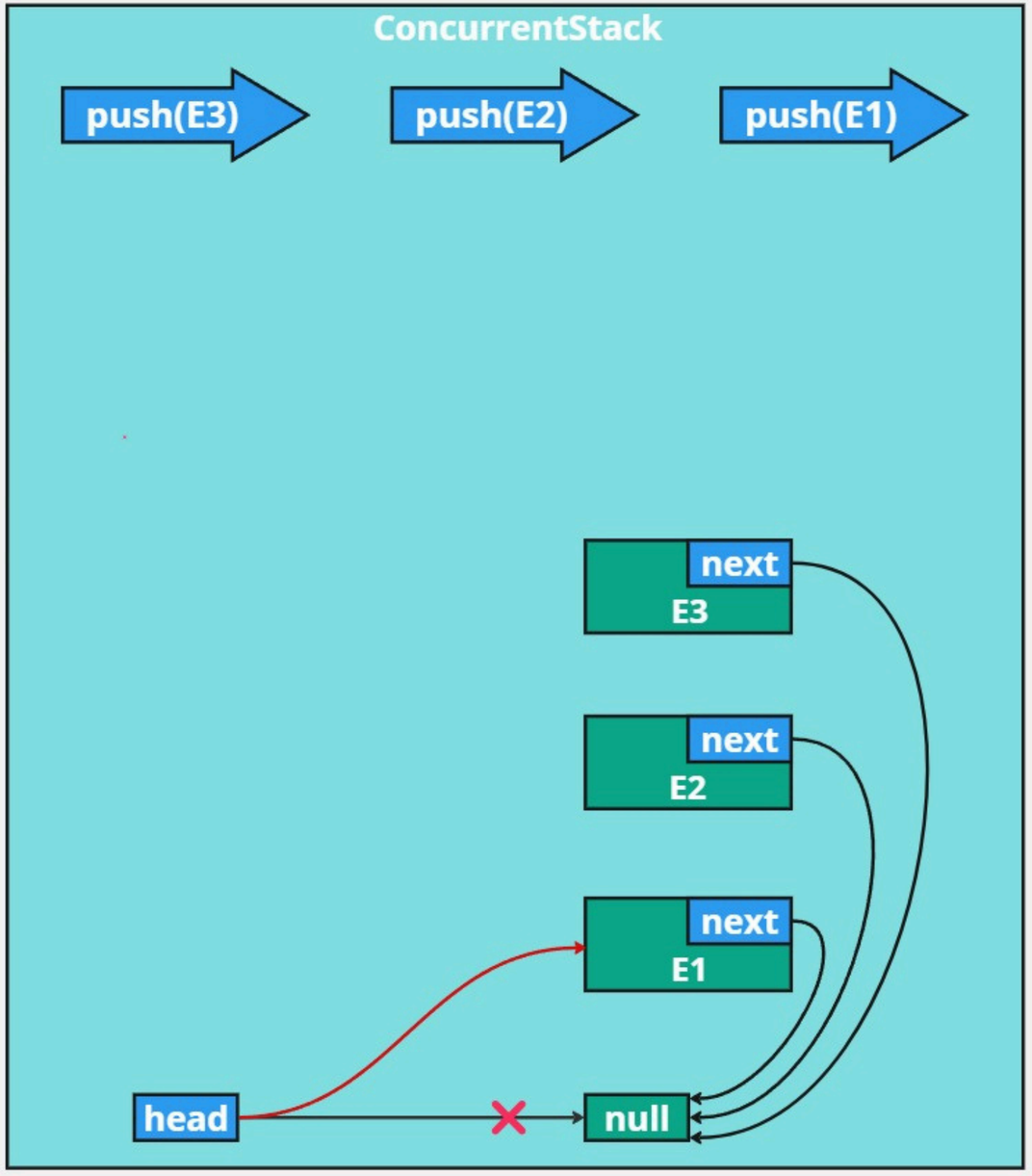
```
public void push(final T element) {
    while (true) {                                          null
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

```
public void push(final T element) {
    while (true) {                                          null
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {                                          null
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```



push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E2)

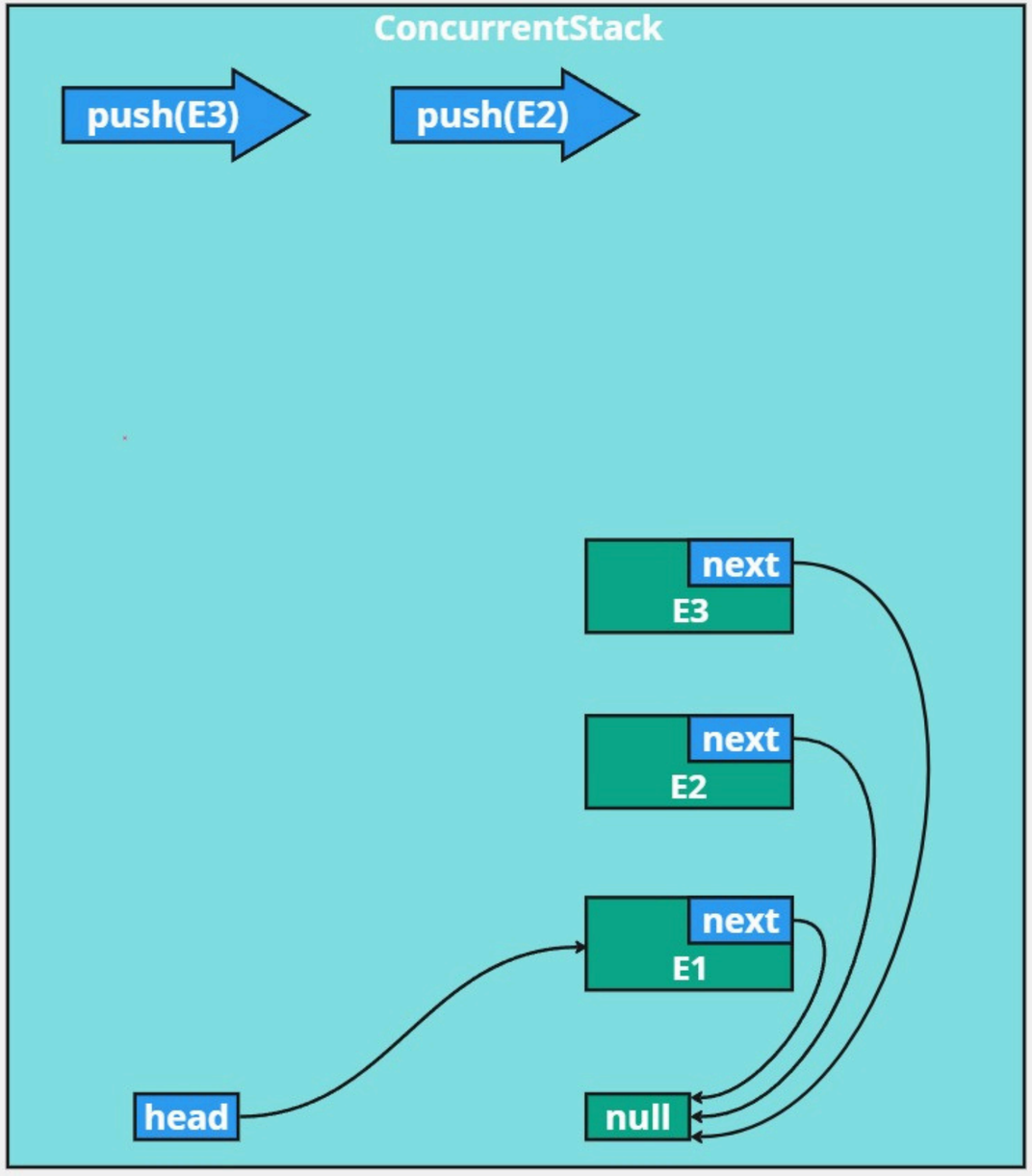
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null



push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E2)

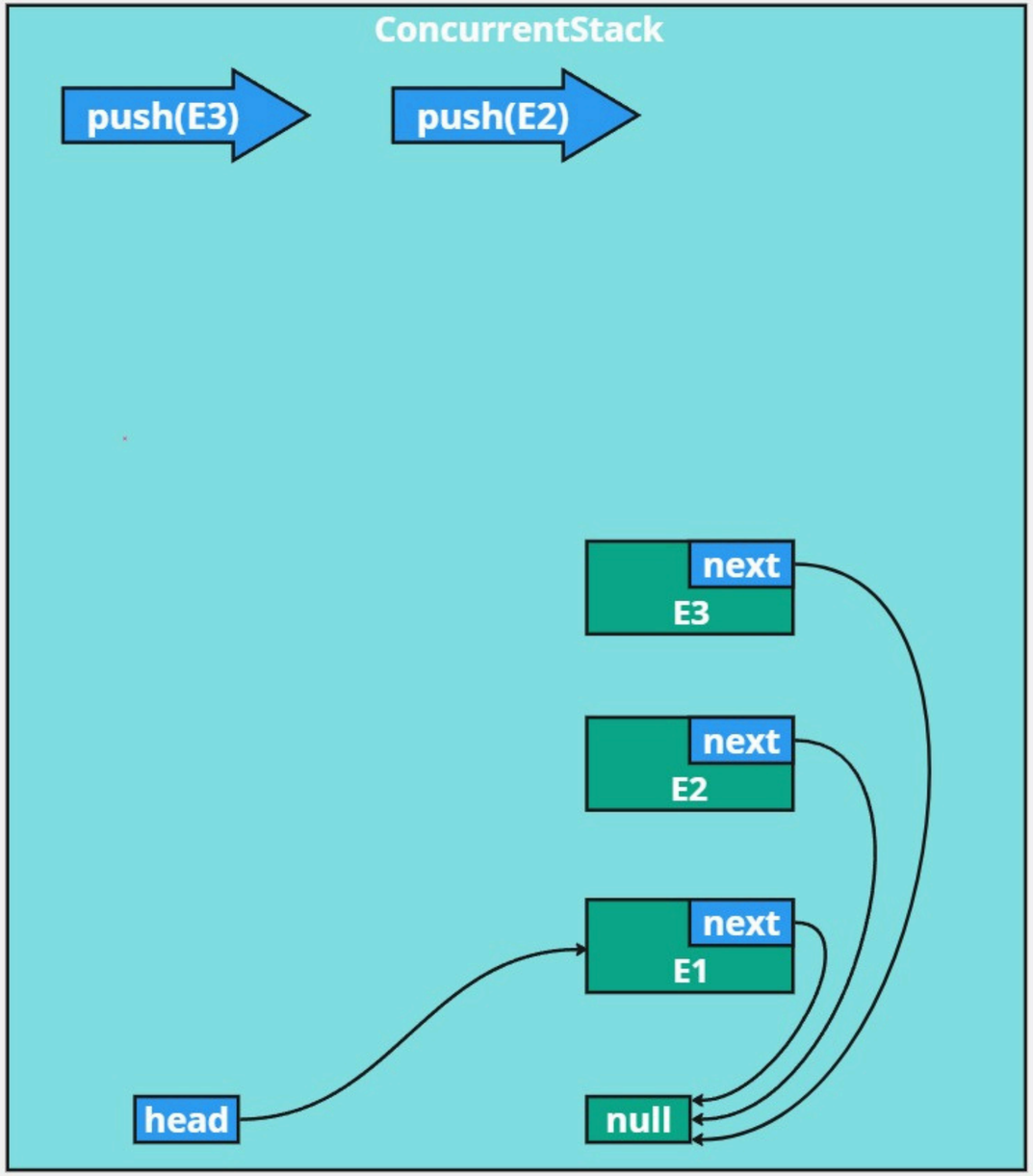
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null



push(E1)

push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

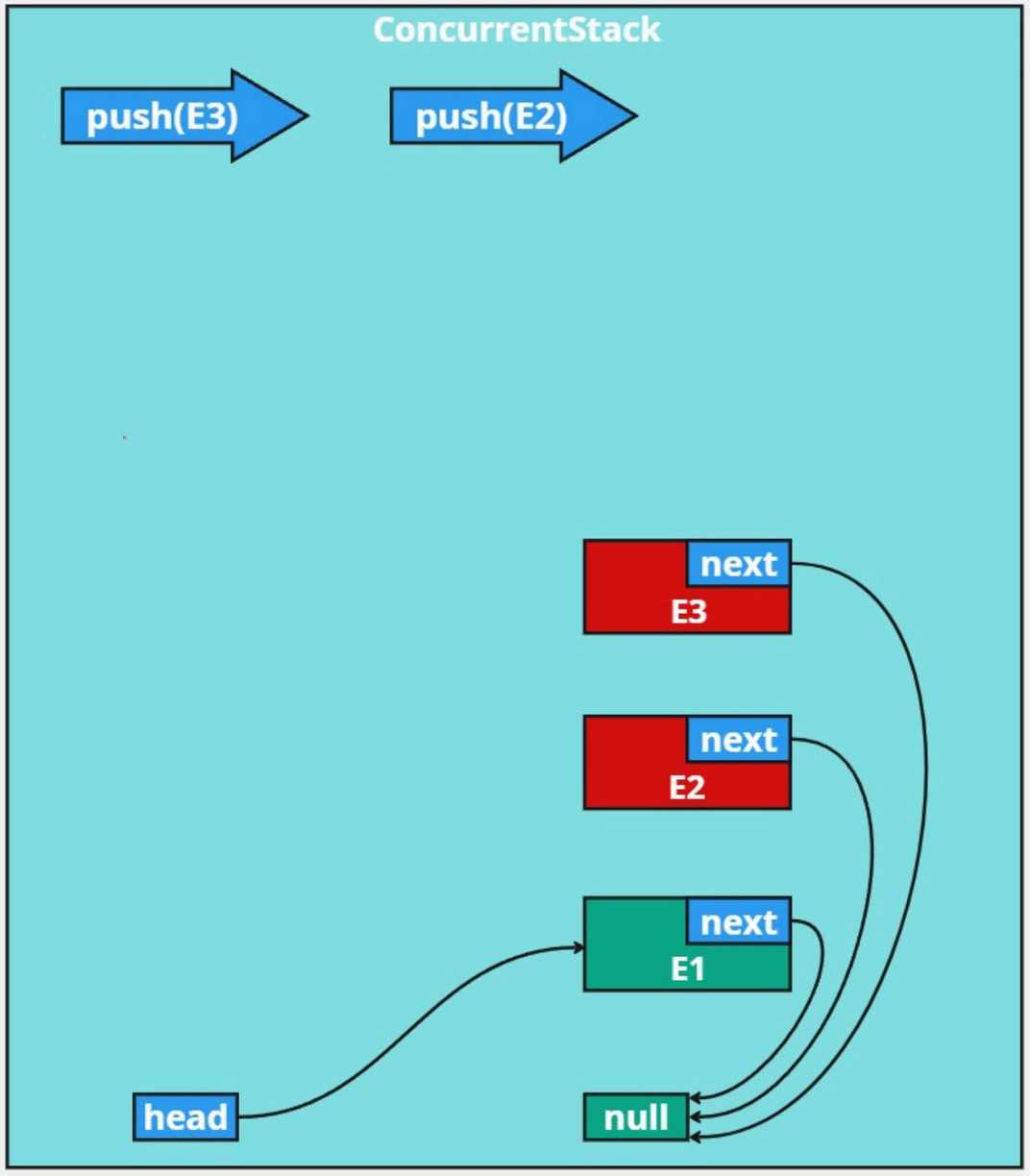
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

null



push(E1)

push(E1)

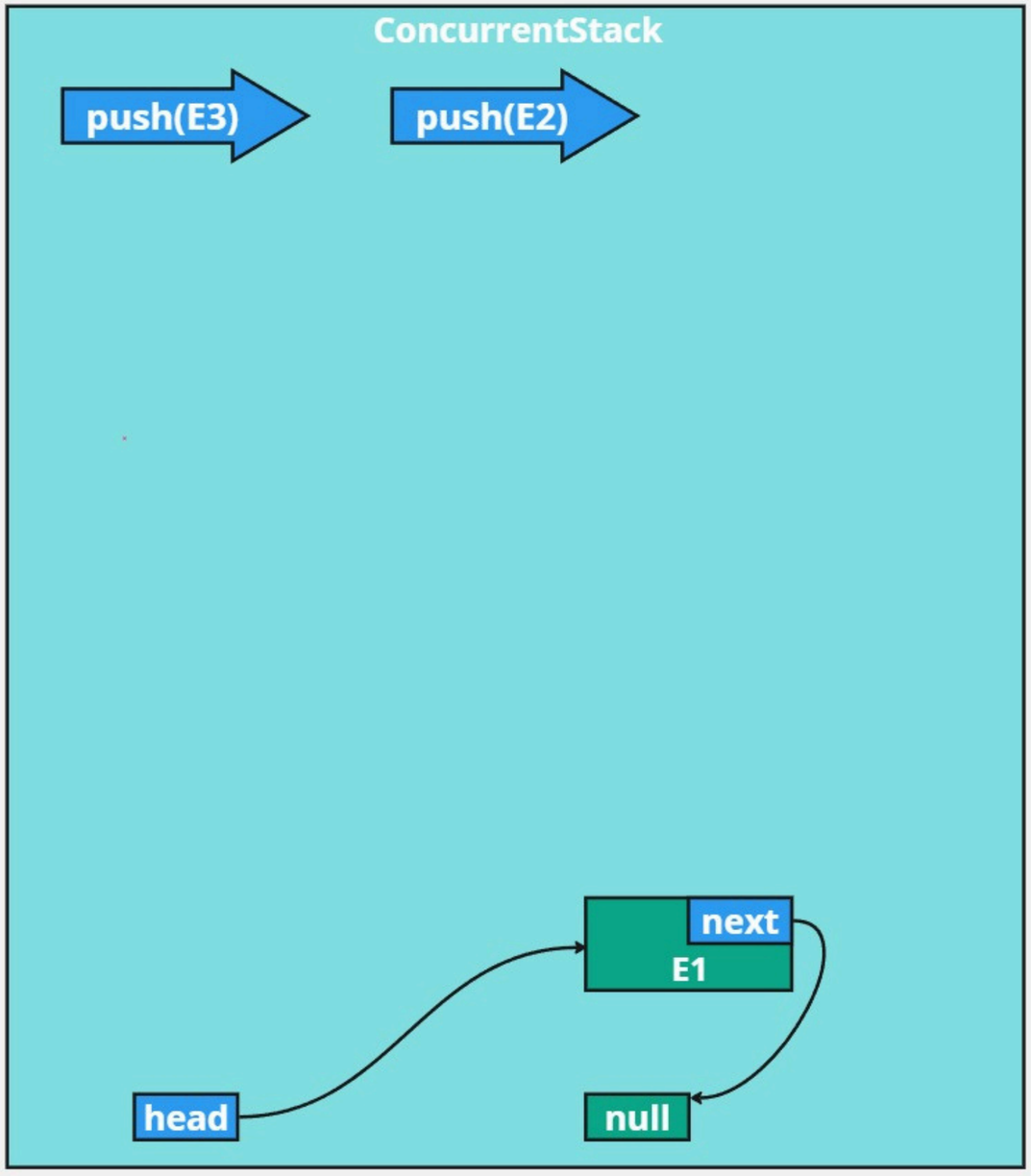
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```



push(E1)

push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

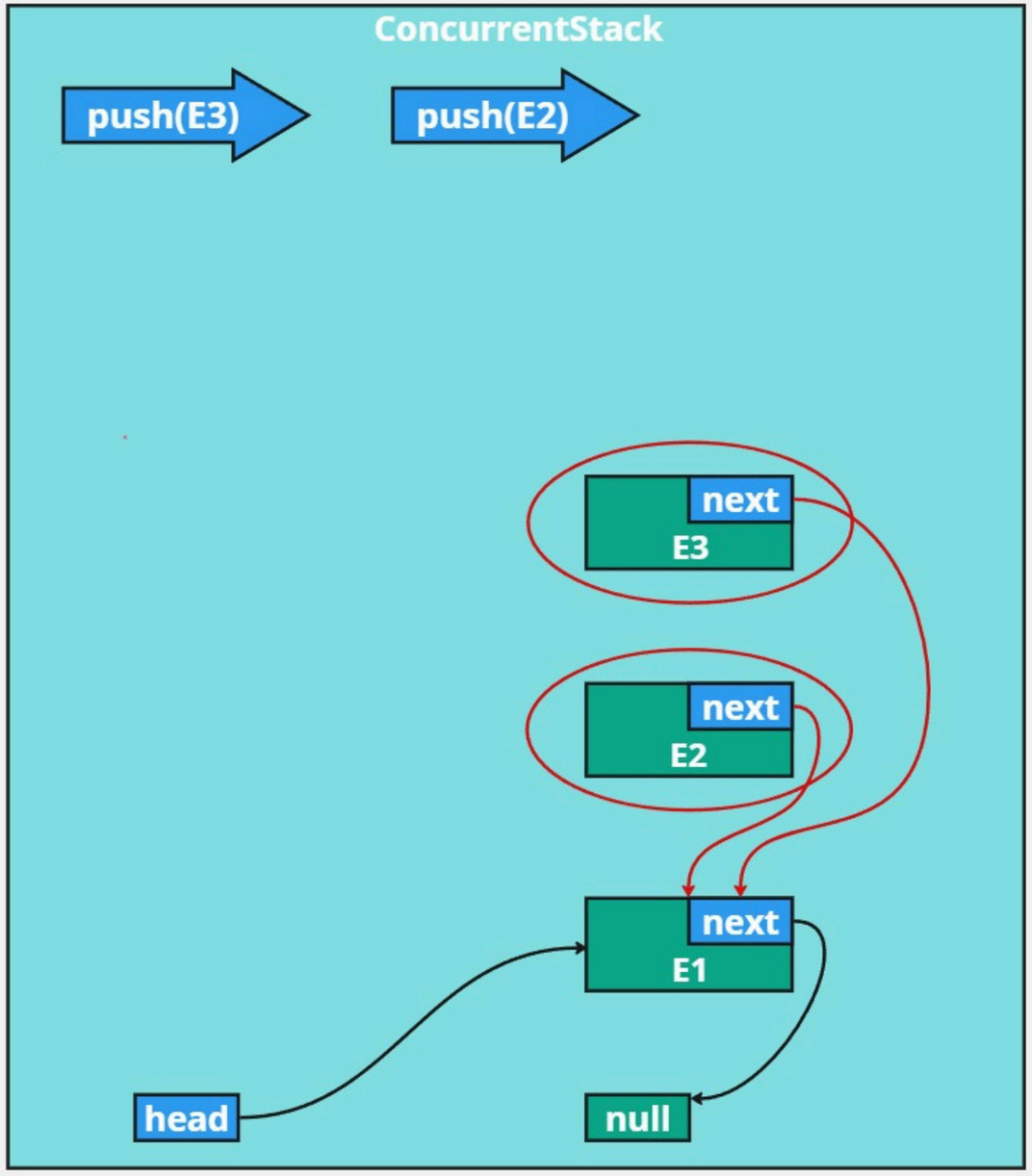
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E1

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E1



push(E1)

push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

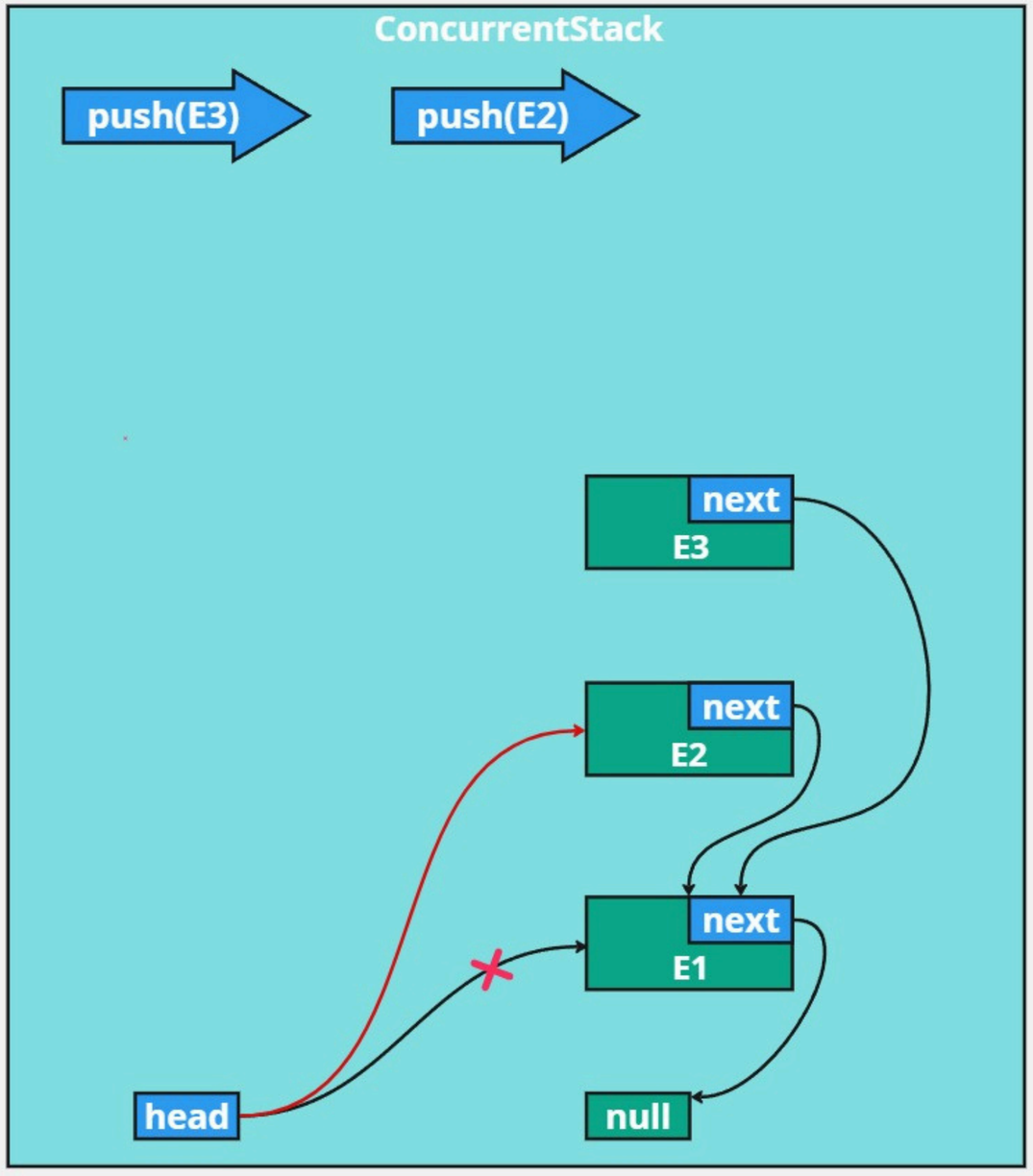
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E1

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E1



push(E1)

push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

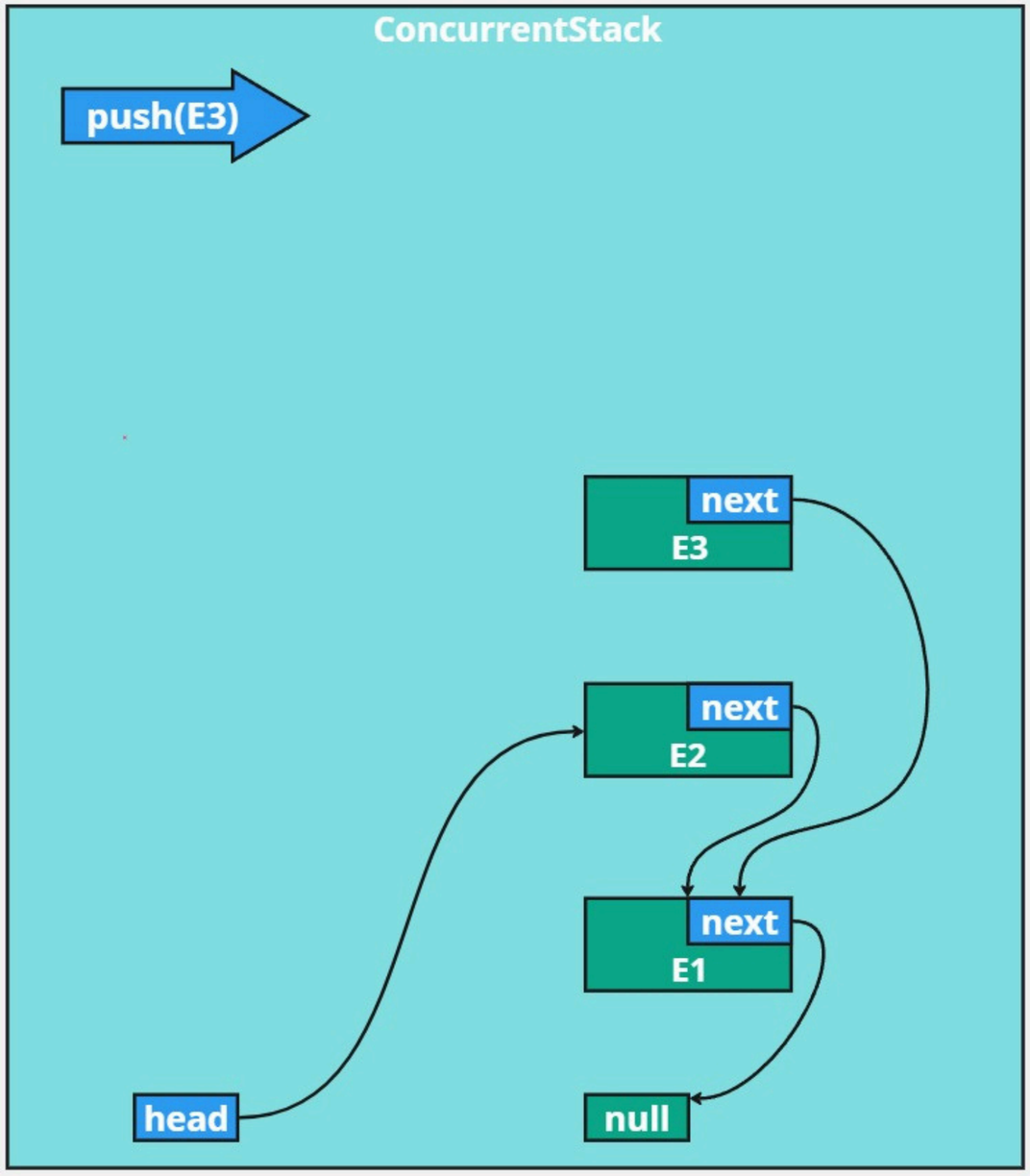
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E1

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

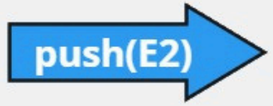
next E1



push(E1)


```java
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

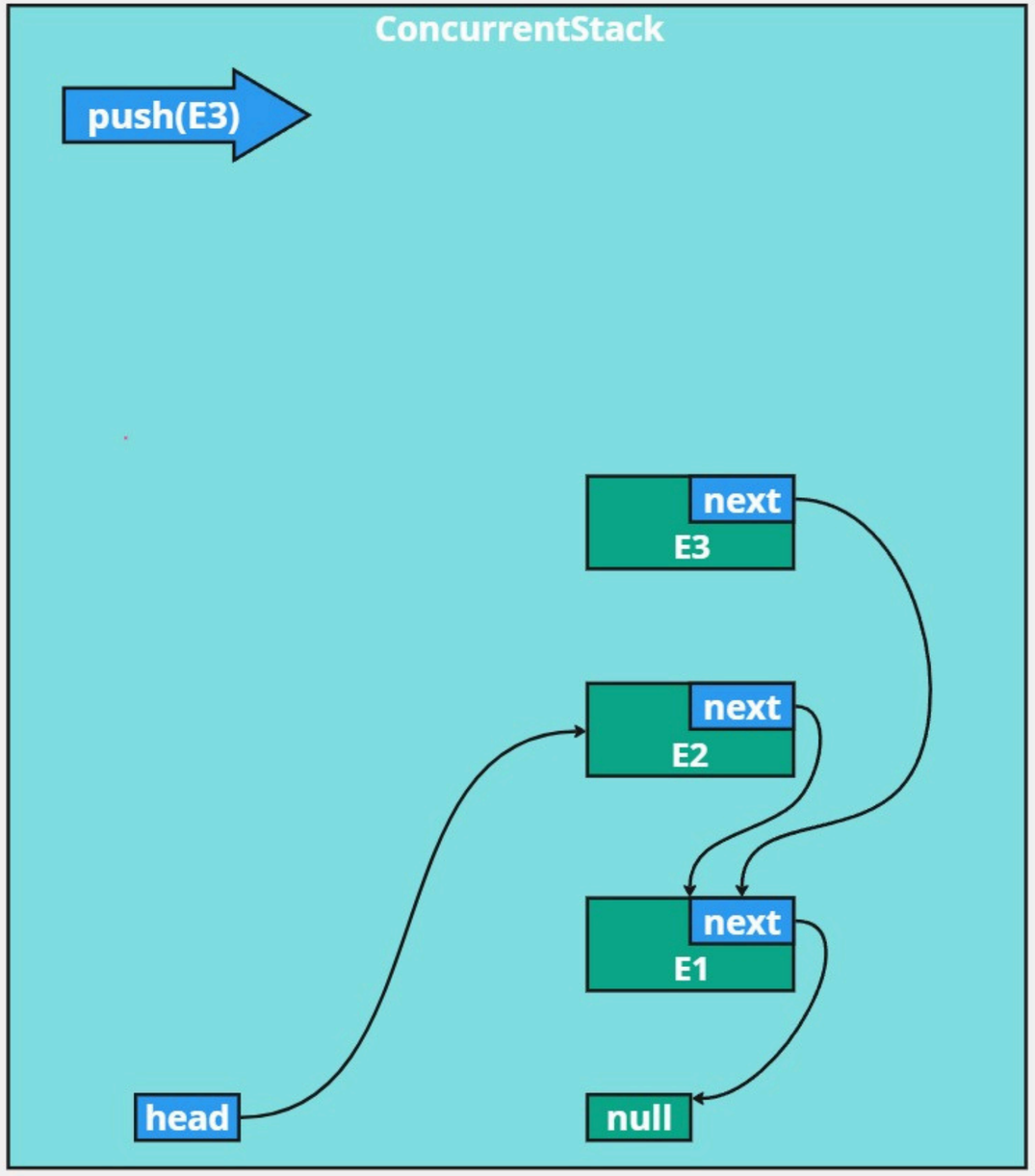
push(E2)




```java
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```


push(E3)

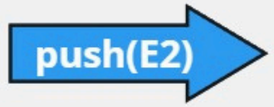
```java
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```



push(E1)


```java
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

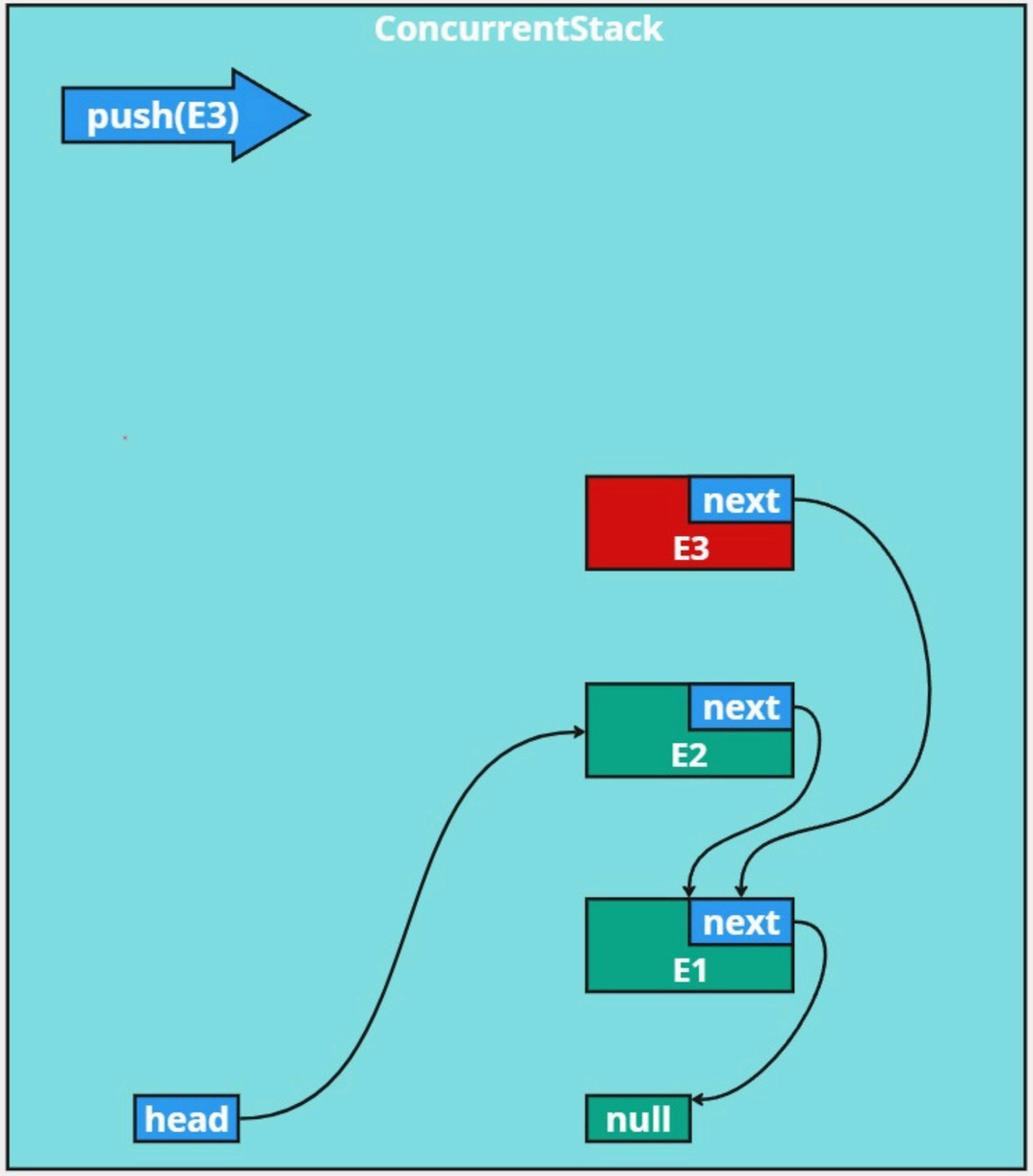
push(E2)




```java
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```java
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```



push(E1)

push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

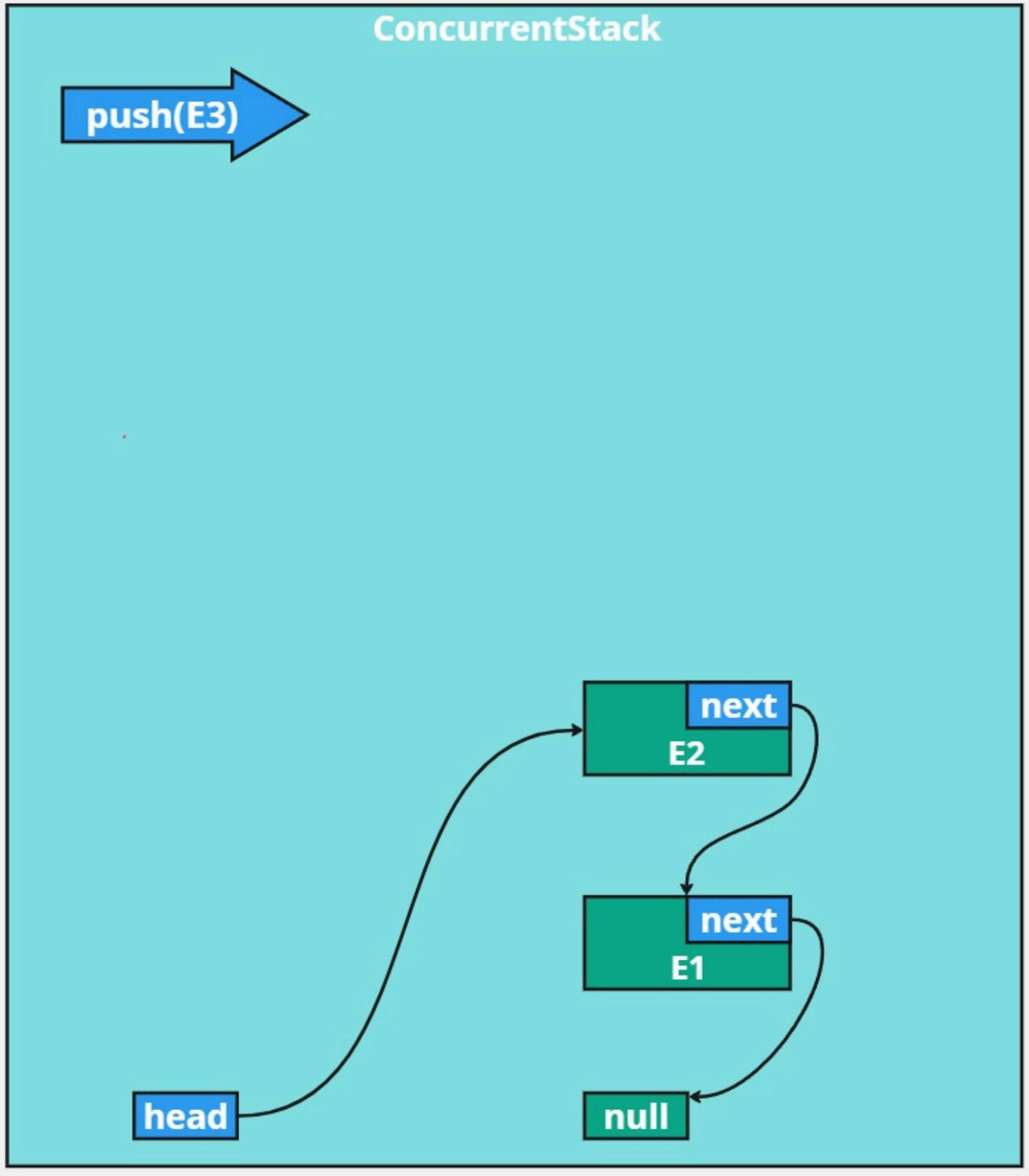
push(E2)

push(E2)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```



push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

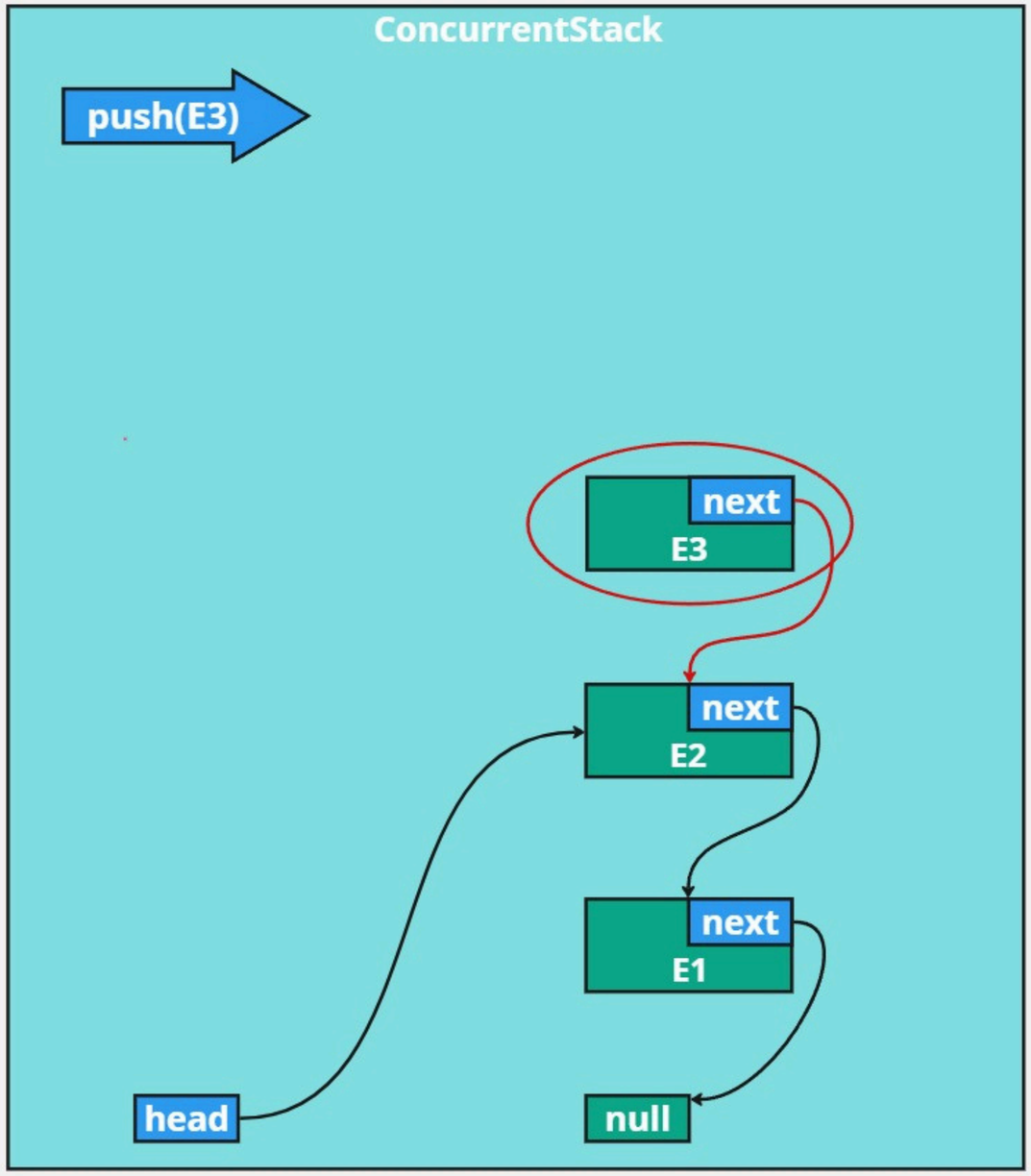
push(E2)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next
E2



push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

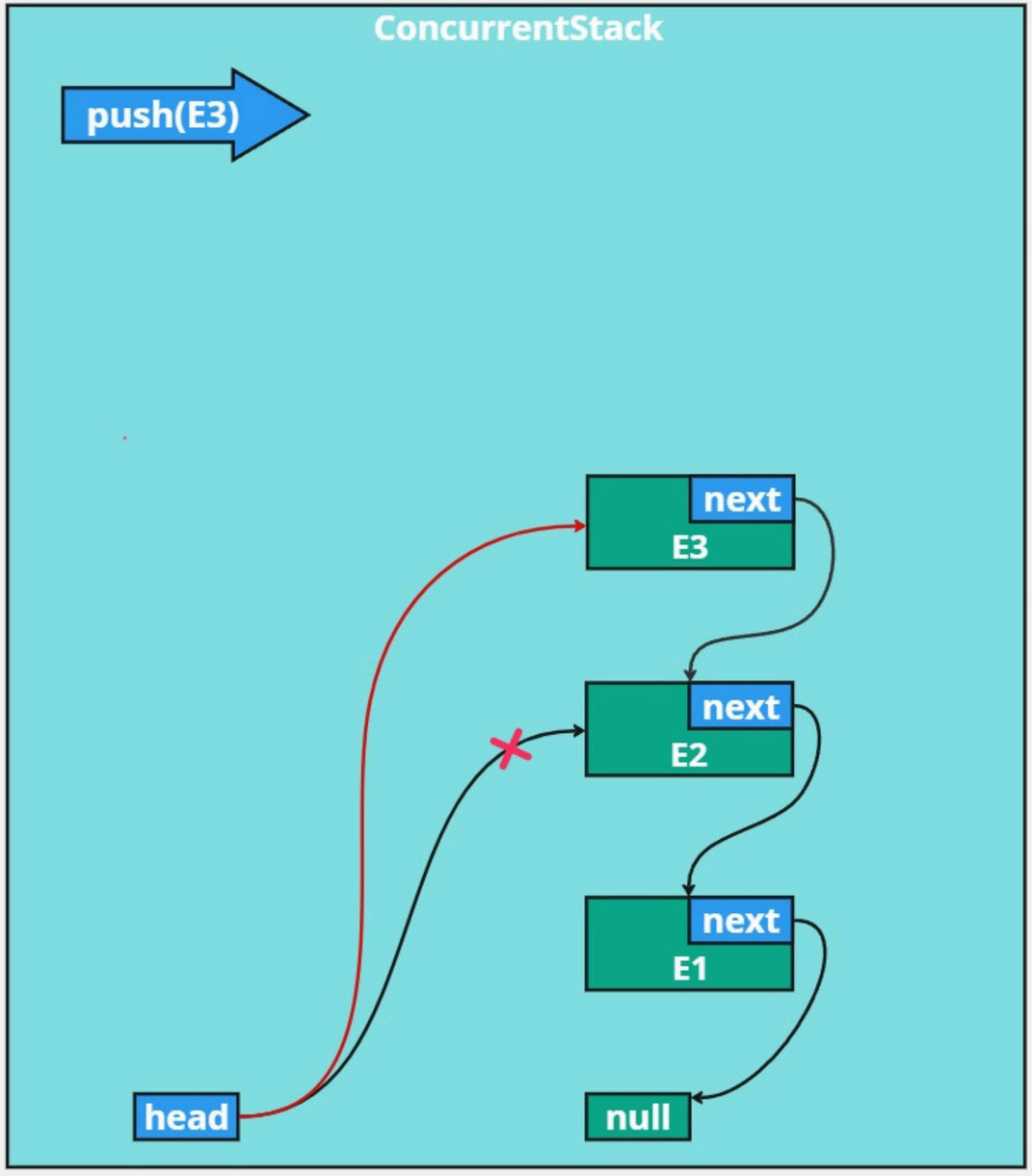
push(E2)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E2



push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

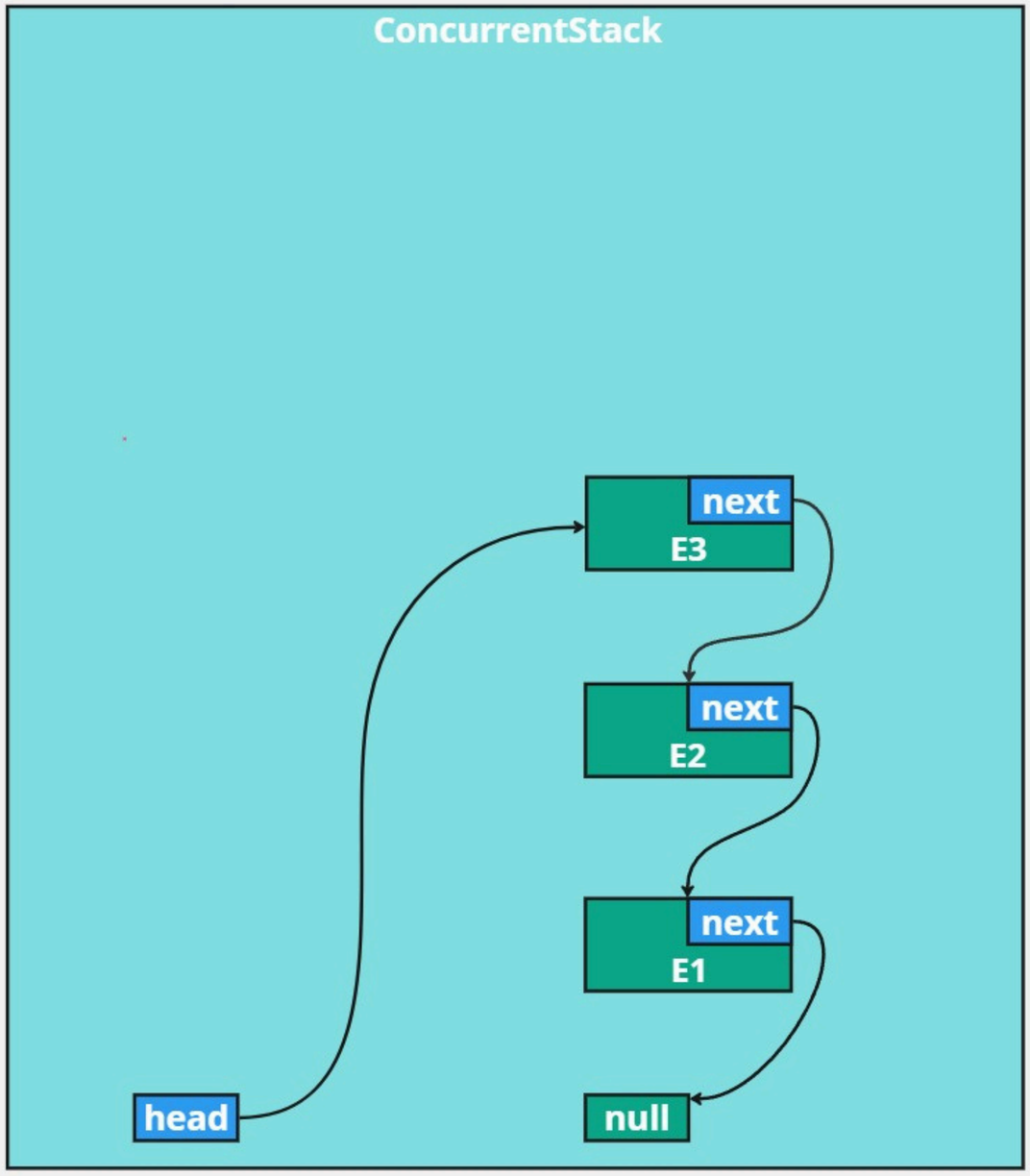
push(E2)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

next E2



push(E1)

push(E1)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E2)

push(E2)

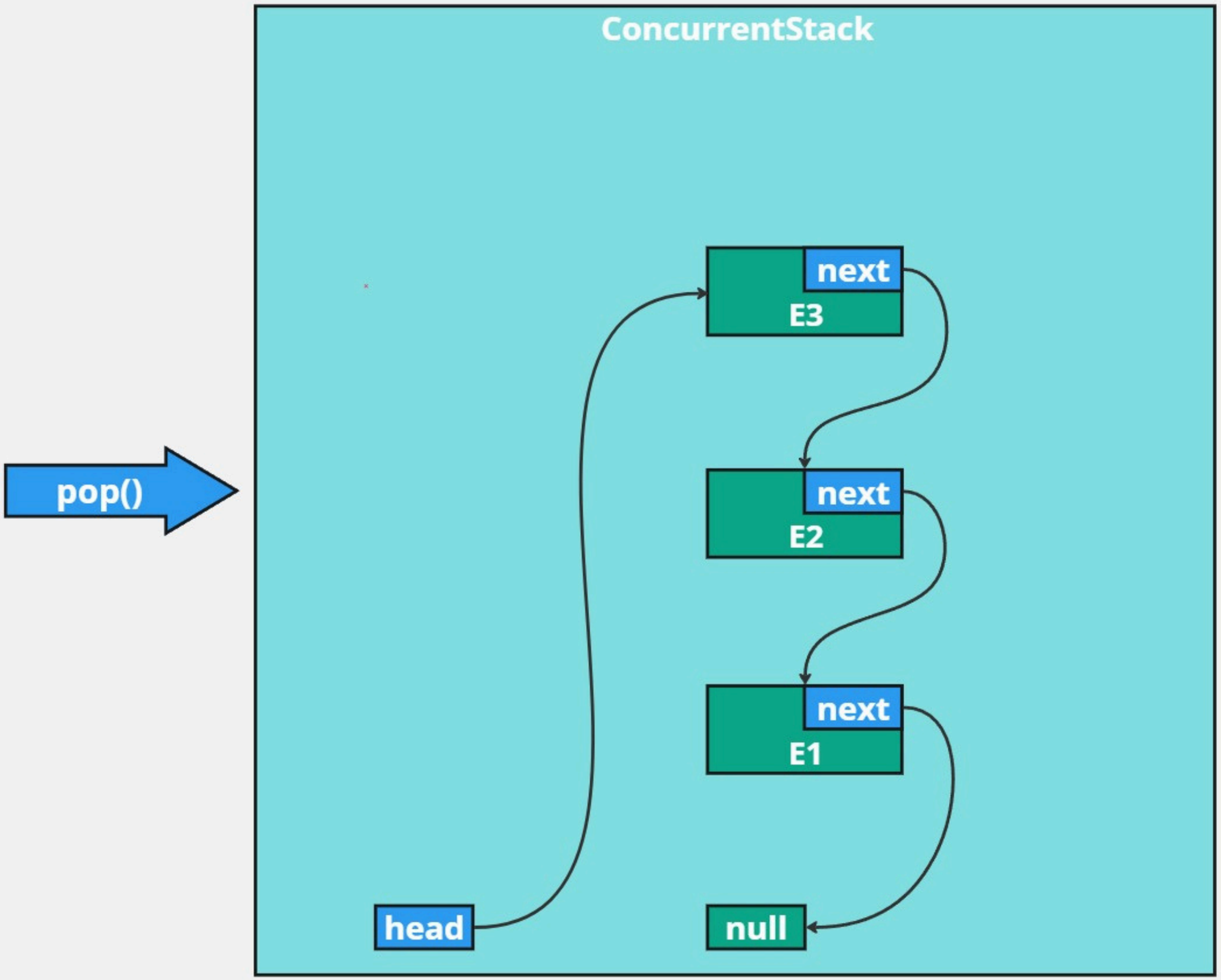
```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

push(E3)

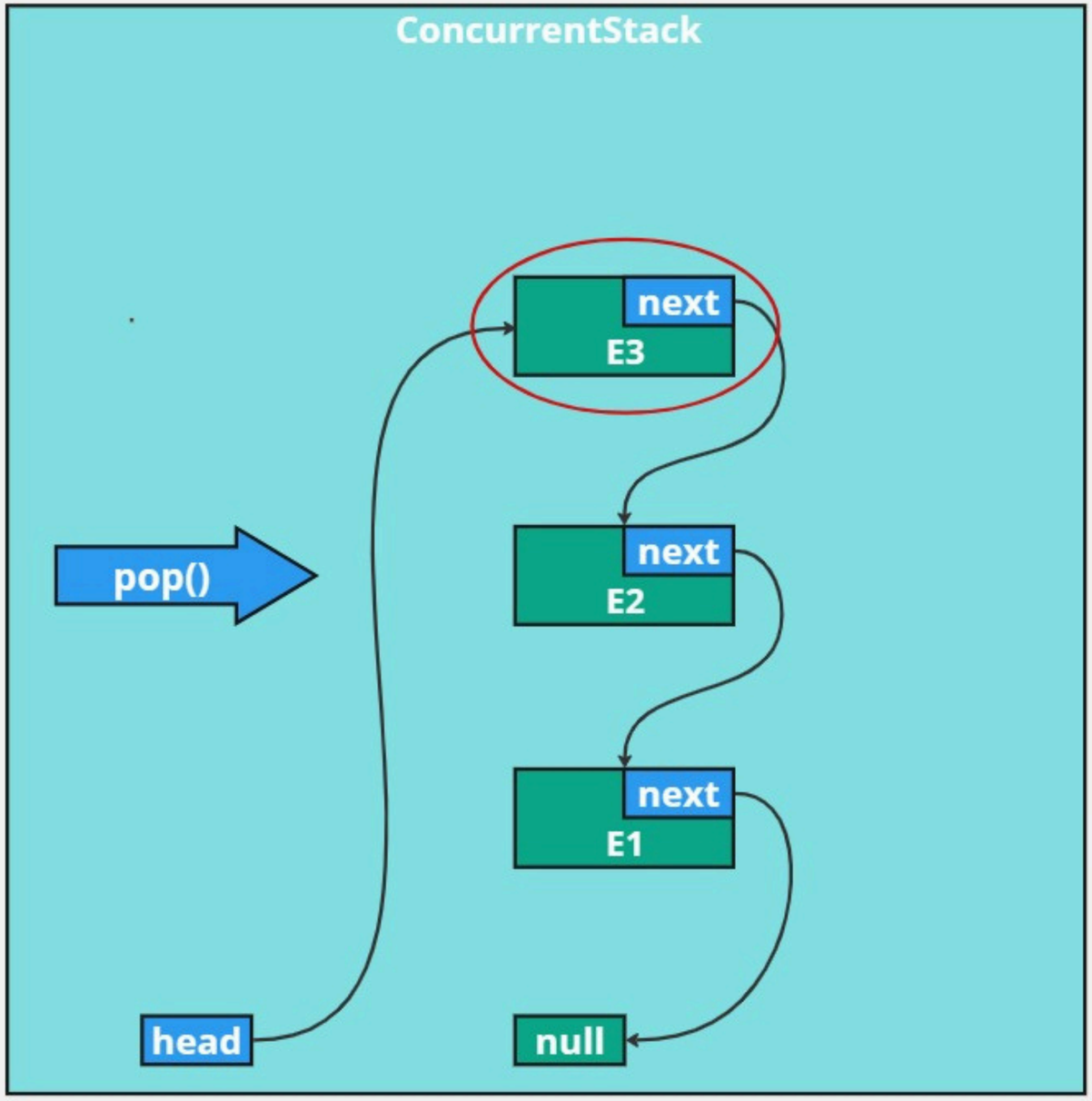
push(E3)

```
public void push(final T element) {
    while (true) {
        final Node<T> previousHead = head.get();
        final Node<T> newNode = new Node<>(element, previousHead);
        if (head.compareAndSet(previousHead, newNode)) {
            return;
        }
    }
}
```

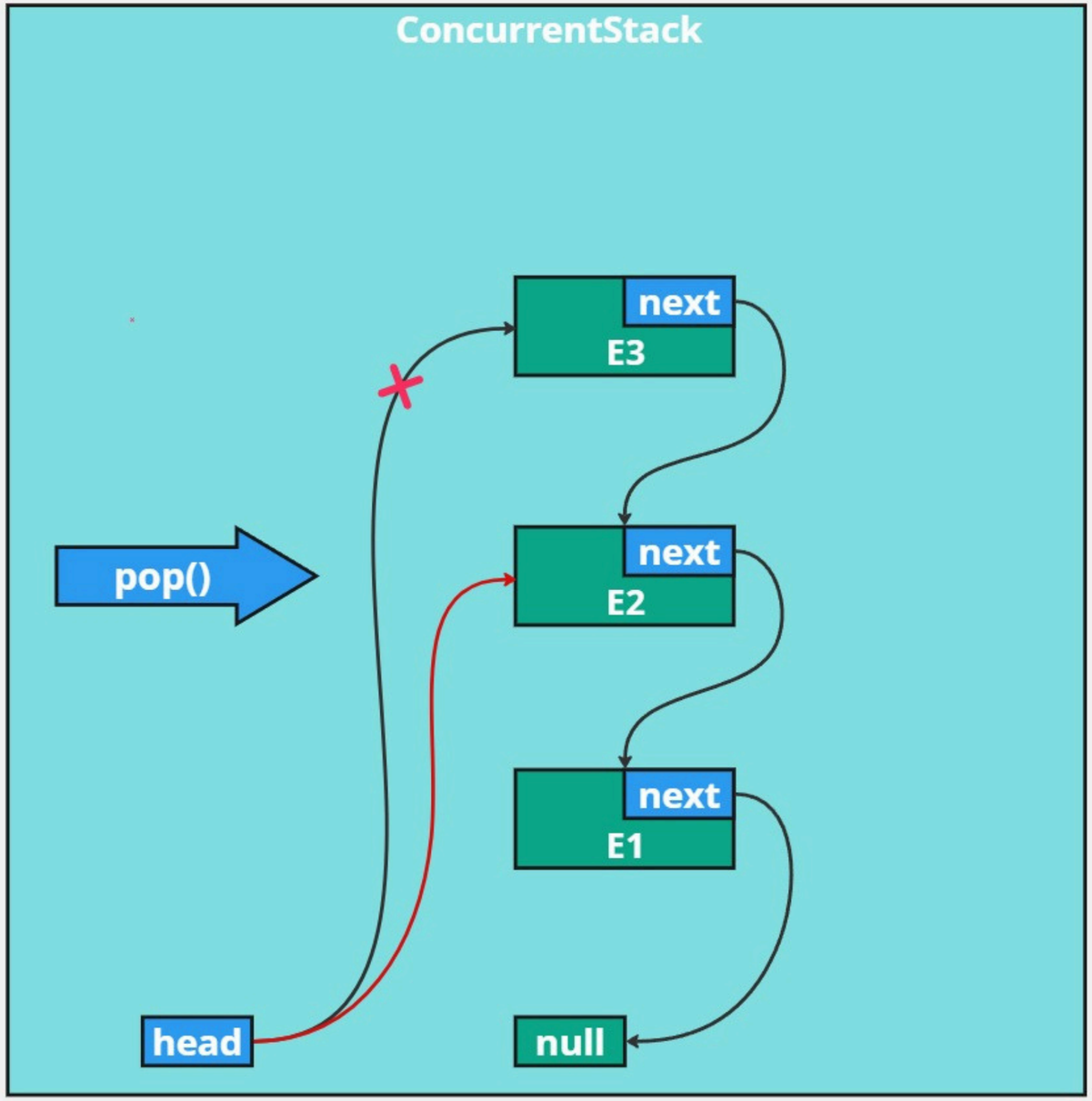
next
E2



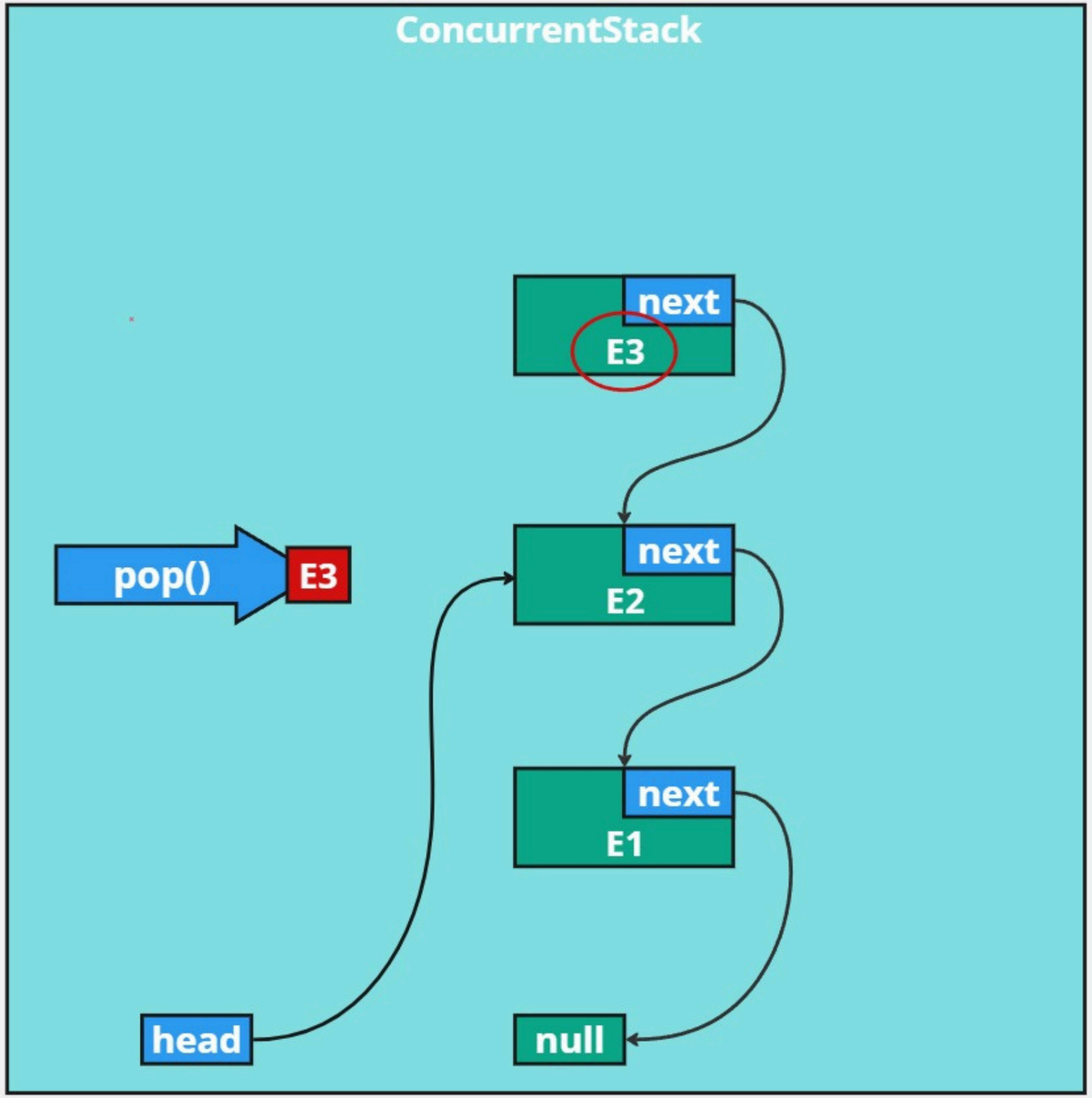

```
public T pop() {
    //memorize previous head
    //head should be assigned previous head's next node
    //extract previous head's value
    //return extracted value
}
```



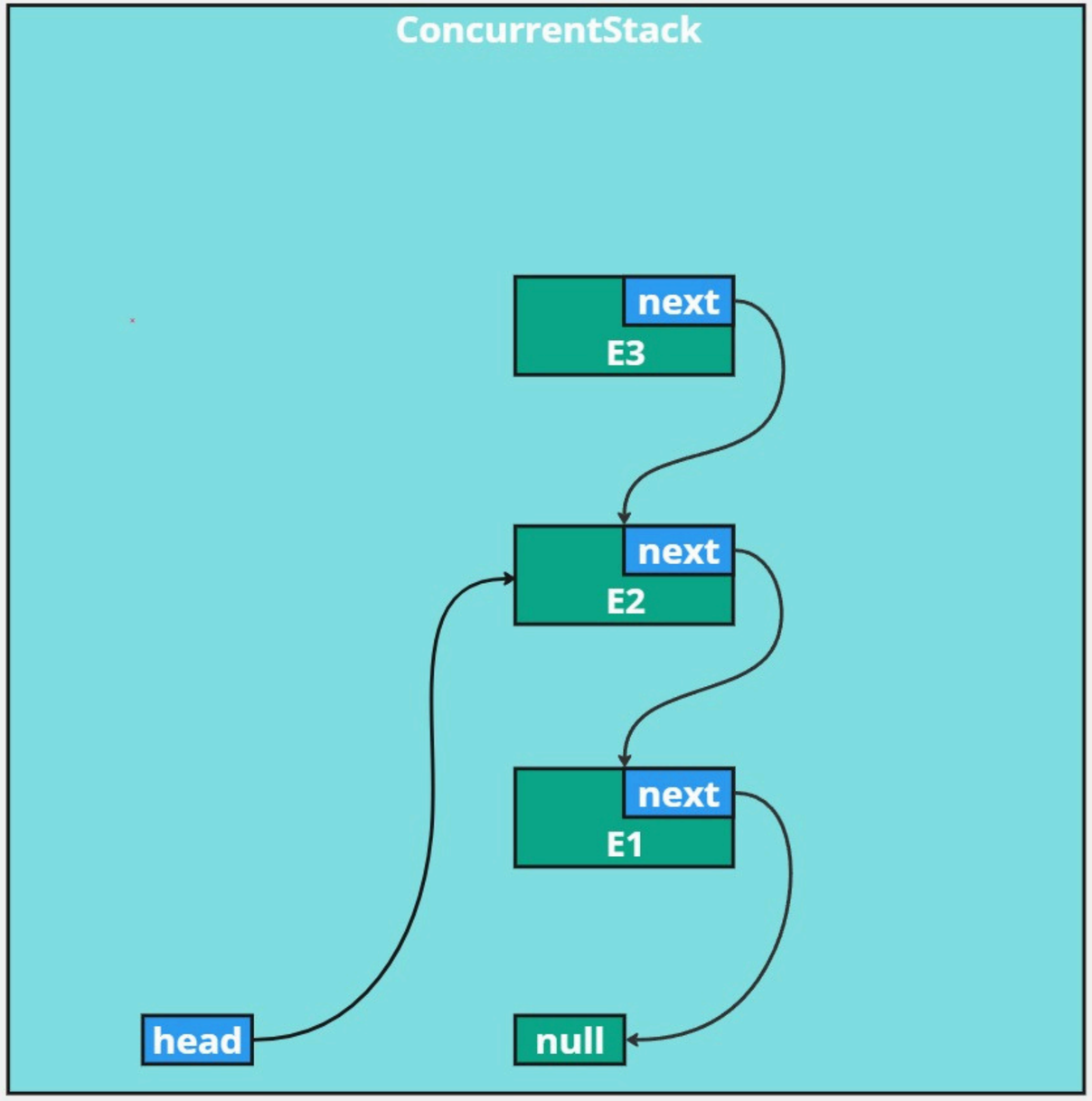

```
public T pop() {
    //memorize previous head
    //head should be assigned previous head's next node
    //extract previous head's value
    //return extracted value
}
```



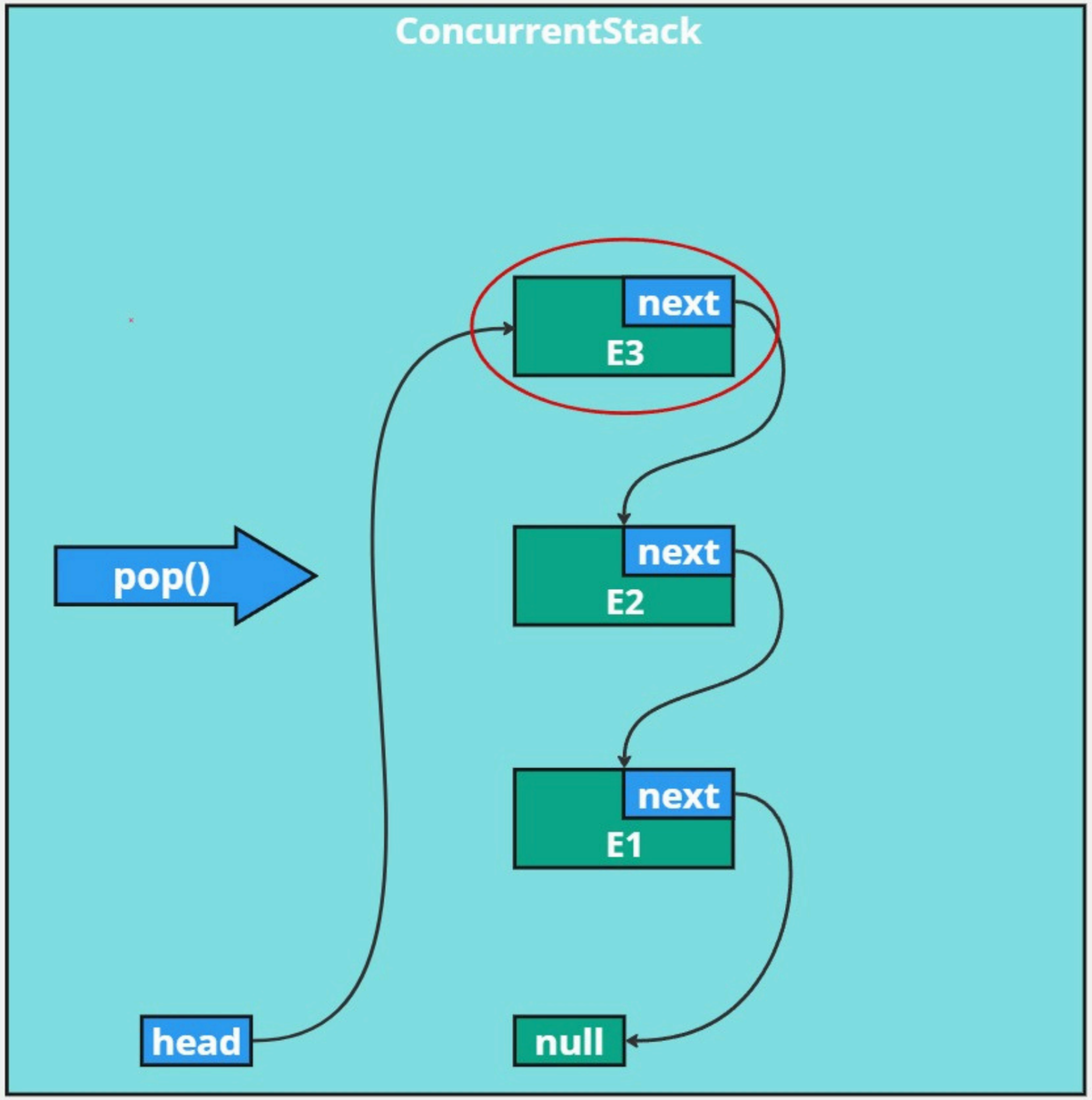

```
public T pop() {
    //memorize previous head
    //head should be assigned previous head's next node
    //extract previous head's value
    //return extracted value
}
```



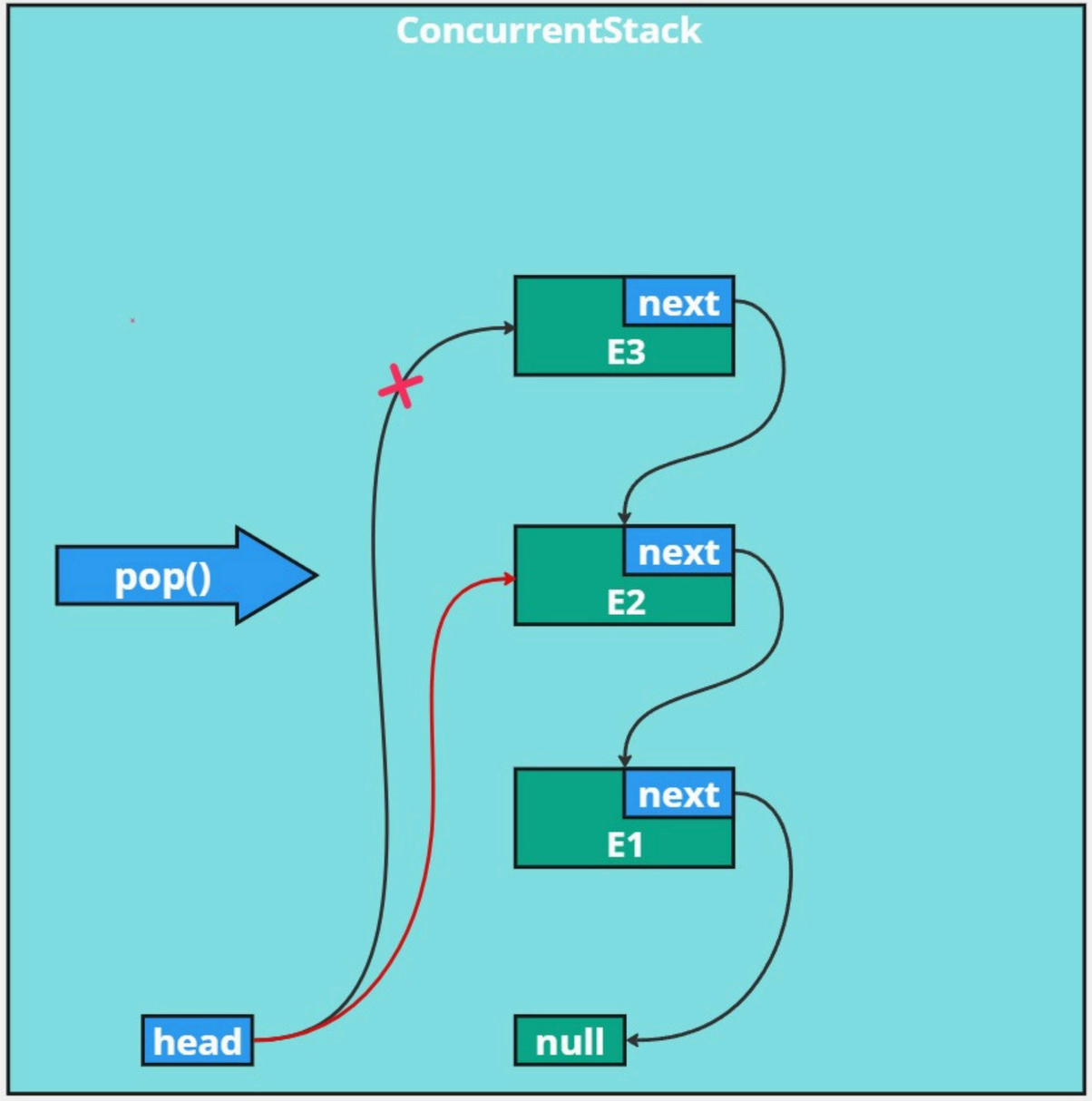

```
public T pop() {
    //memorize previous head
    //head should be assigned previous head's next node
    //extract previous head's value
    //return extracted value
}
```



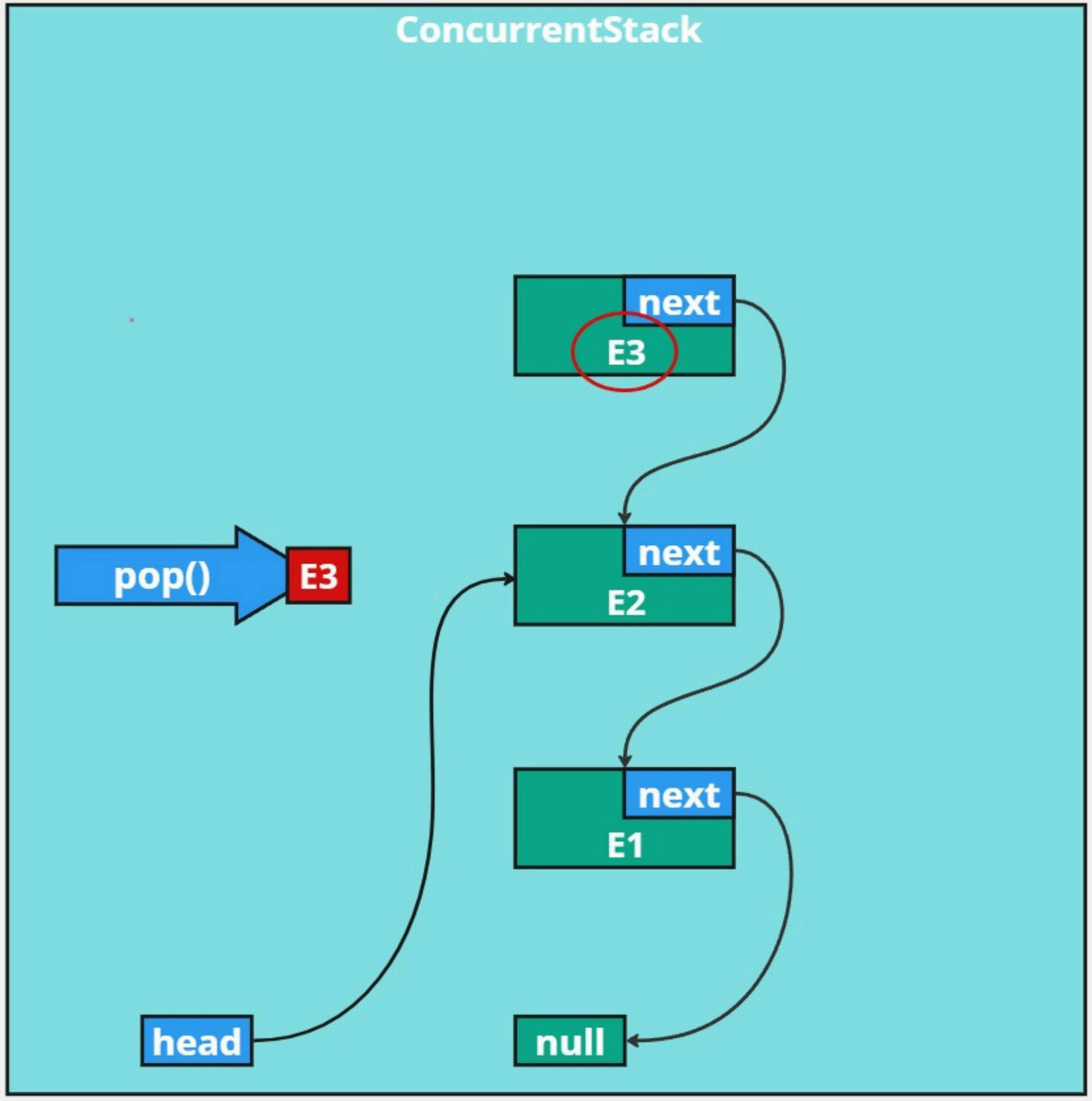

```
public T pop() {
    //memorize previous head
    //head should be assigned previous head's next node
    //extract previous head's value
    //return extracted value
}
```



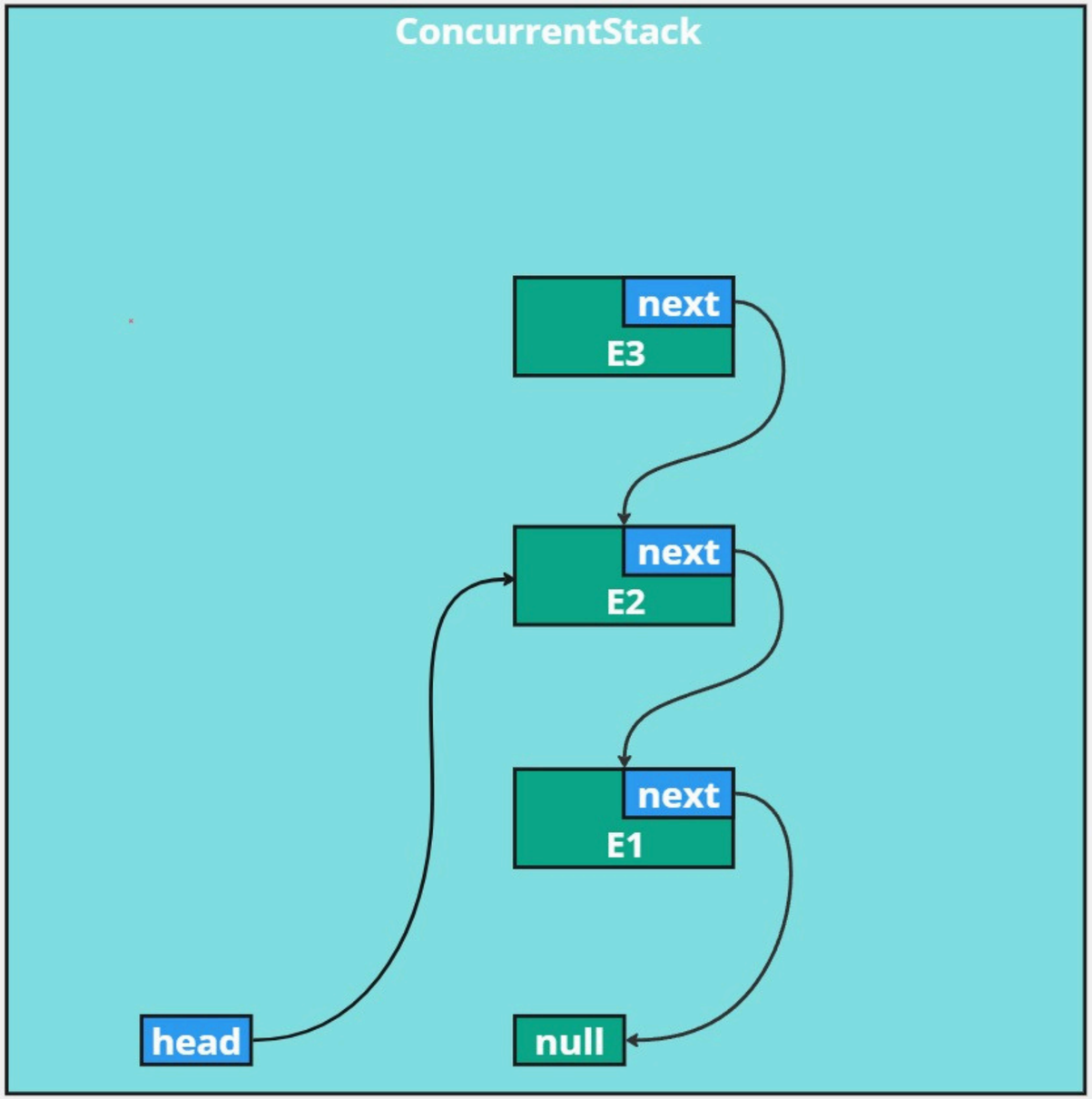

```
public T pop() {
    final Node<T> previousHead = head.get();
    //head should be assigned previous head's next node
    //extract previous head's value
    //return extracted value
}
```



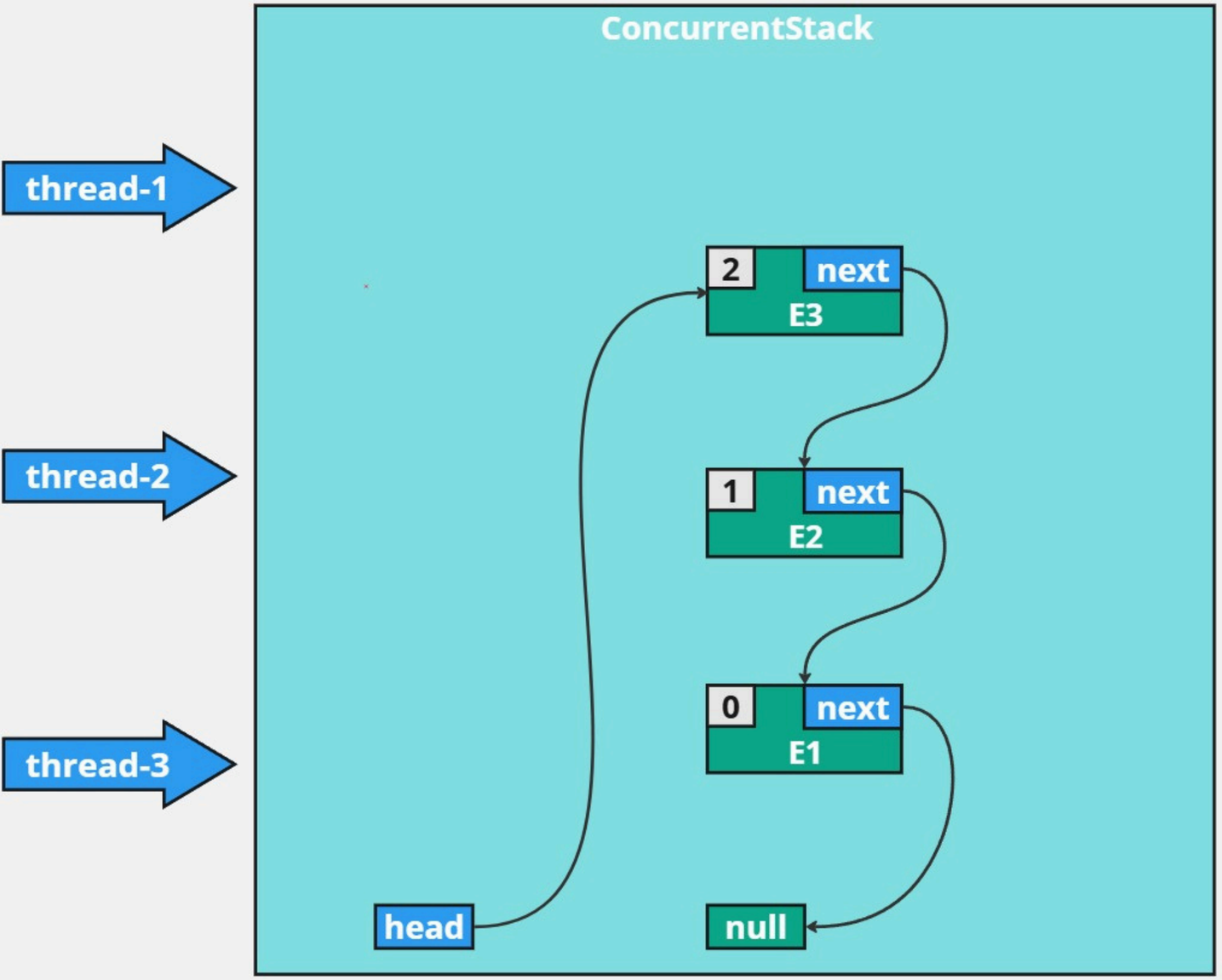

```java
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    //extract previous head's value
    //return extracted value
}
```




```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```




```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```



thread-1

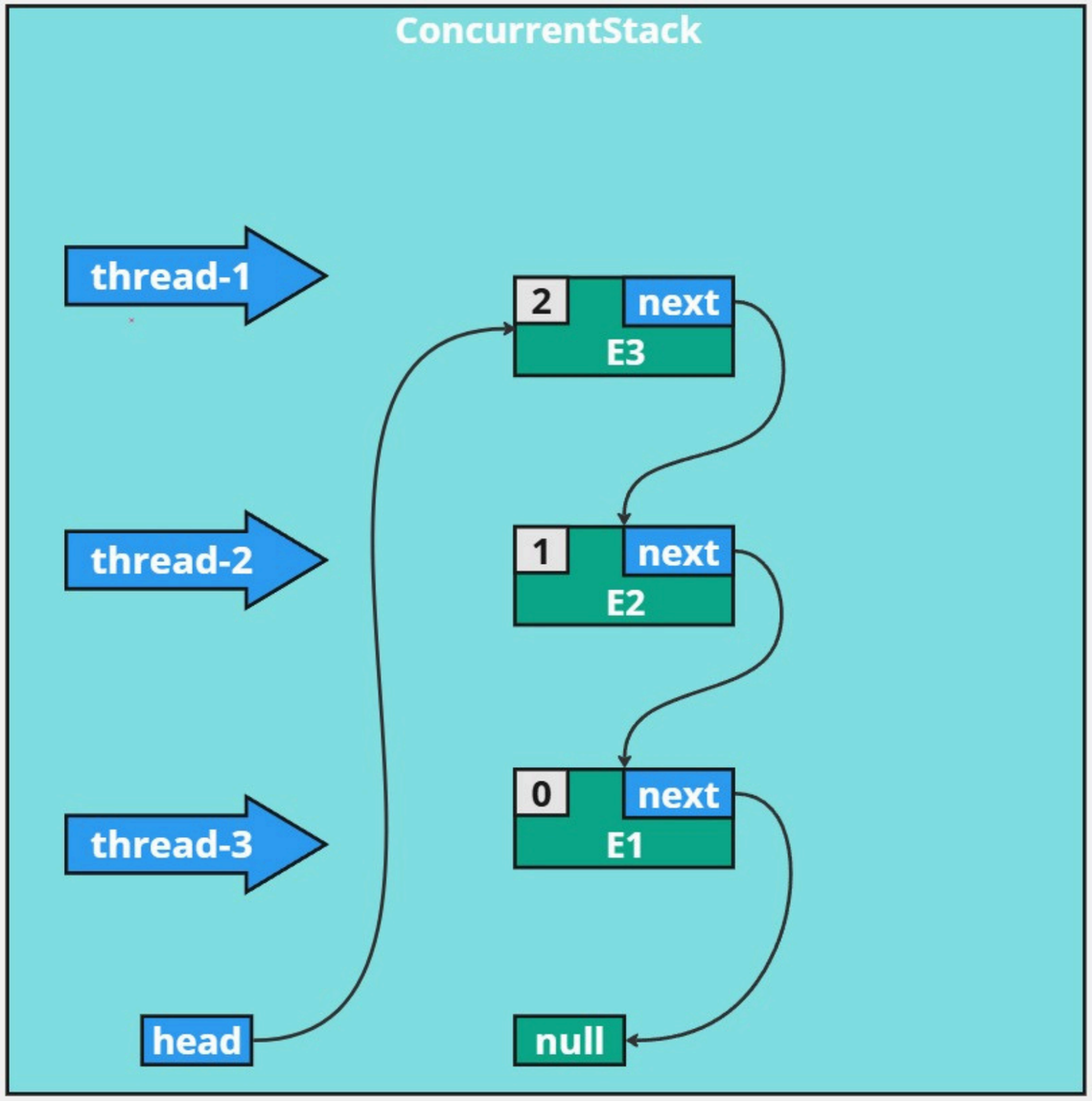
```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-3

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```



thread-1

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

thread-2

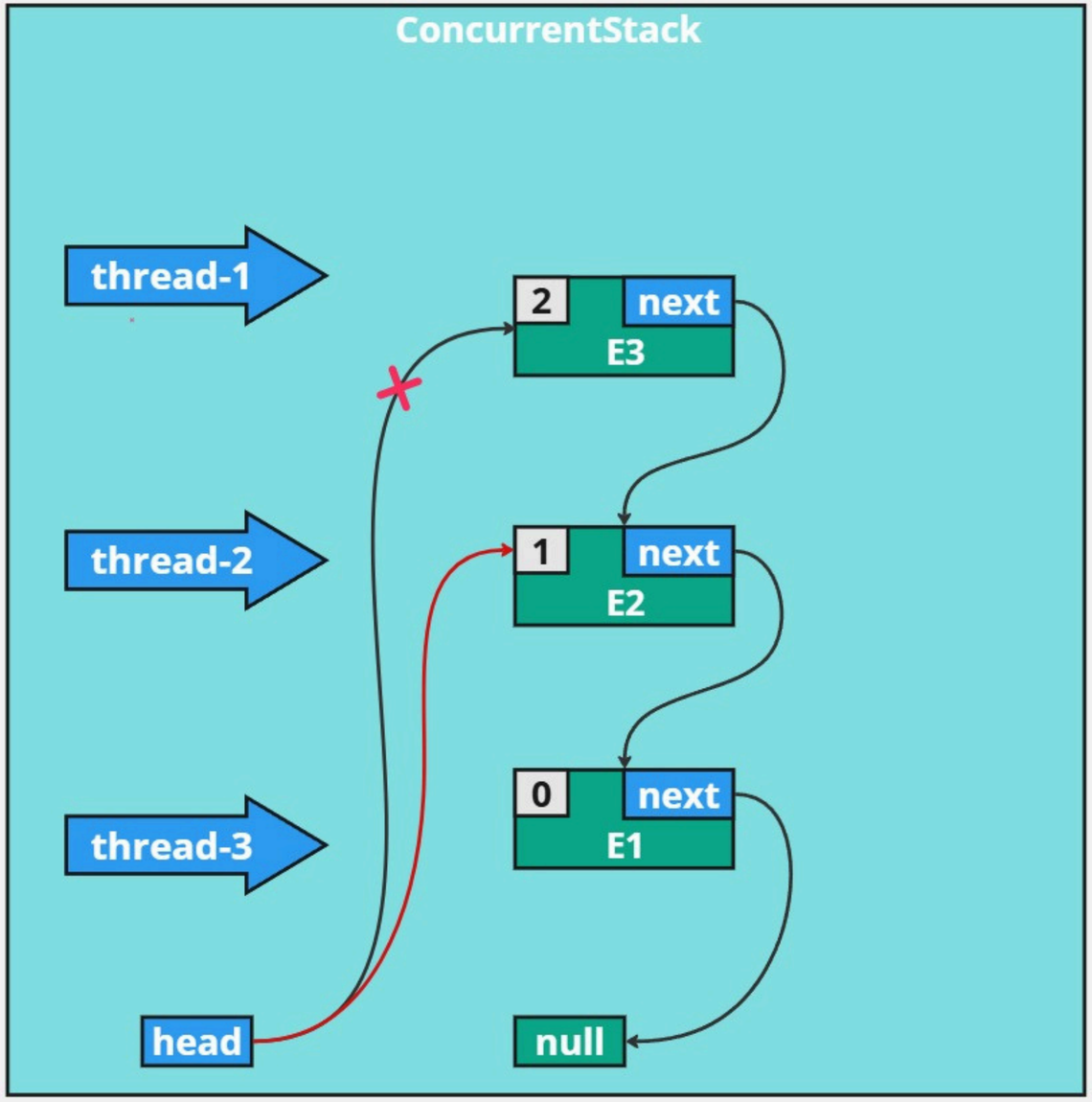
```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

thread-3

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2



thread-1

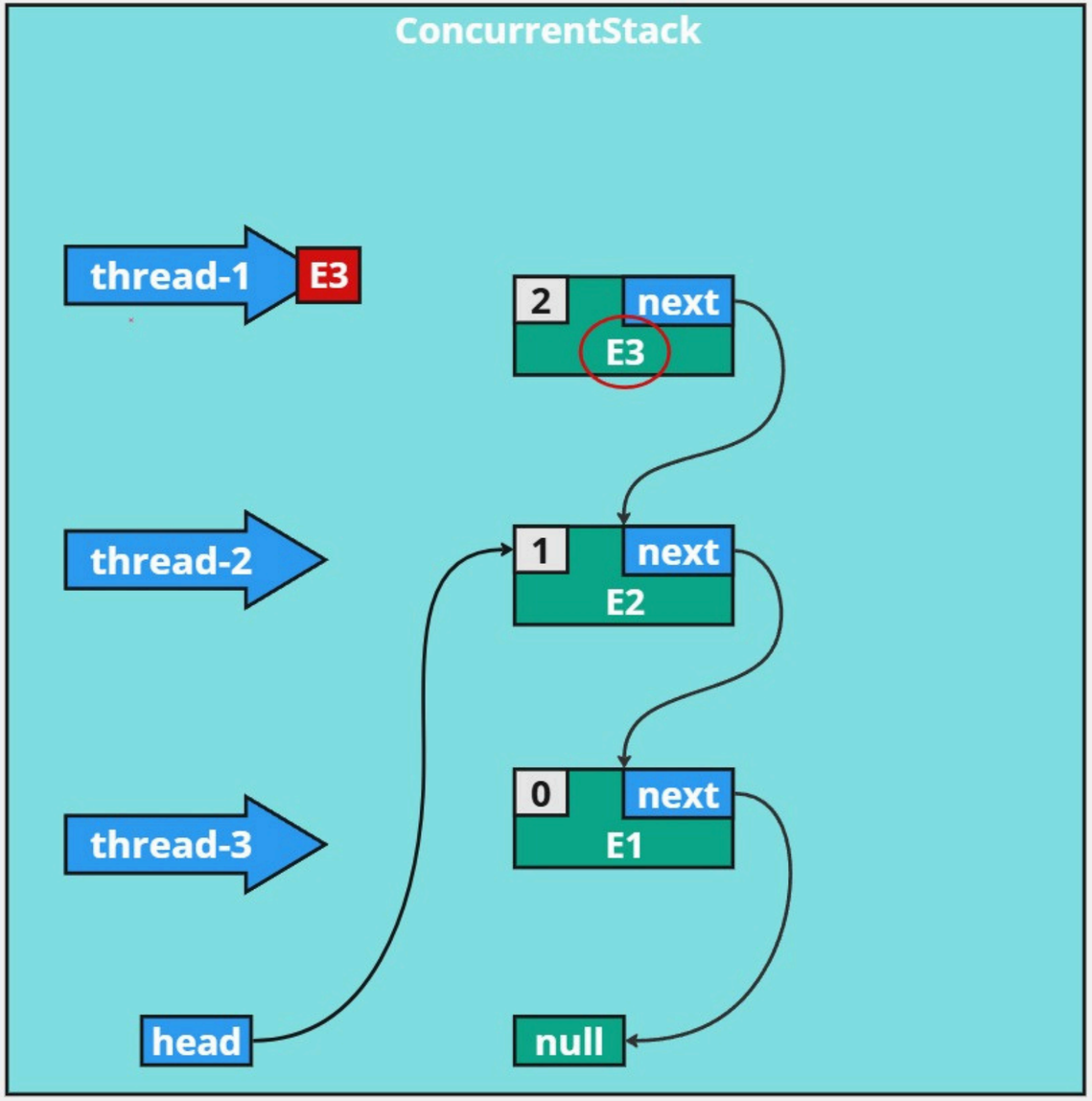
```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-3

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```



thread-1

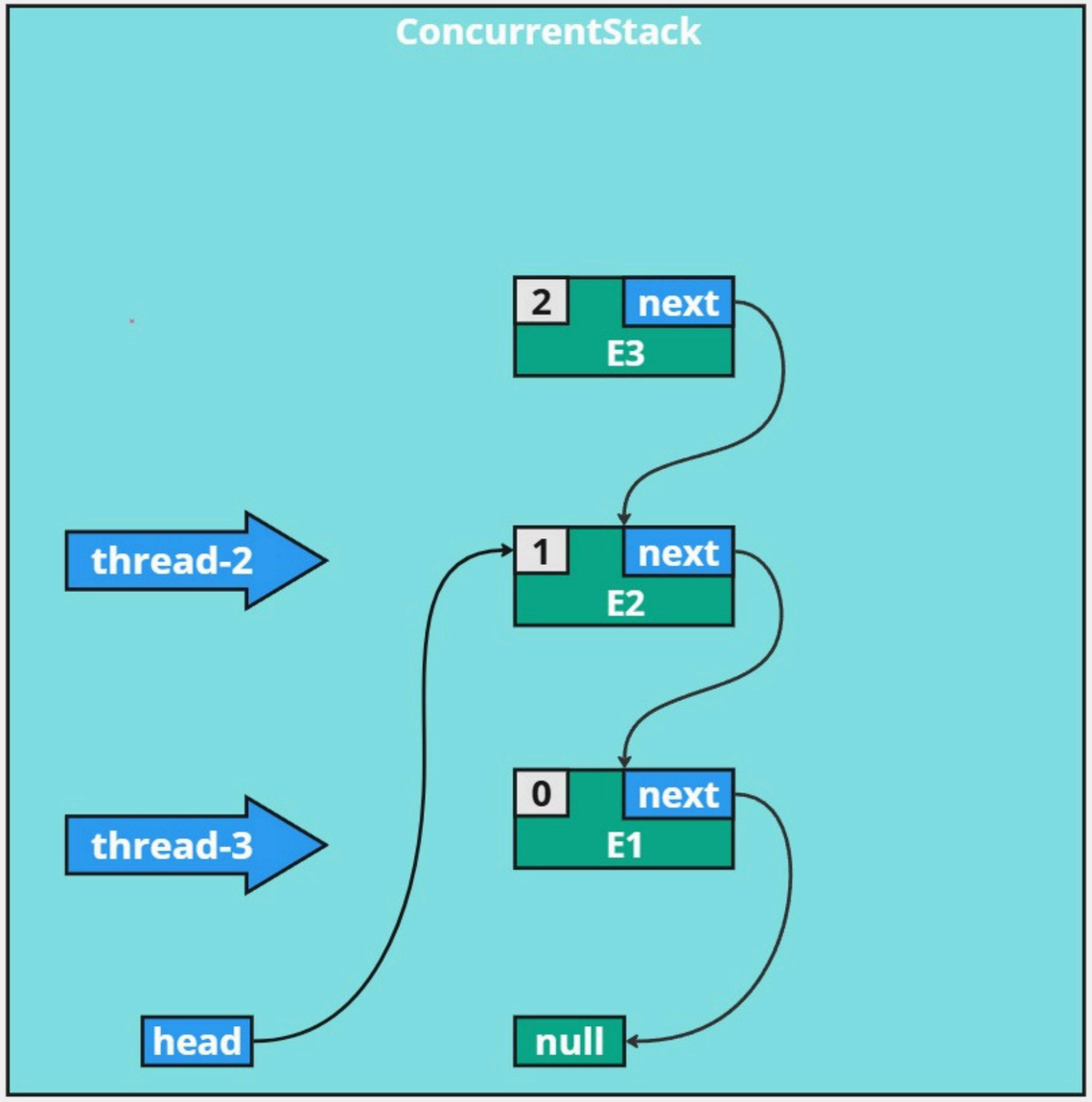
```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-3

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```



thread-1

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

thread-2

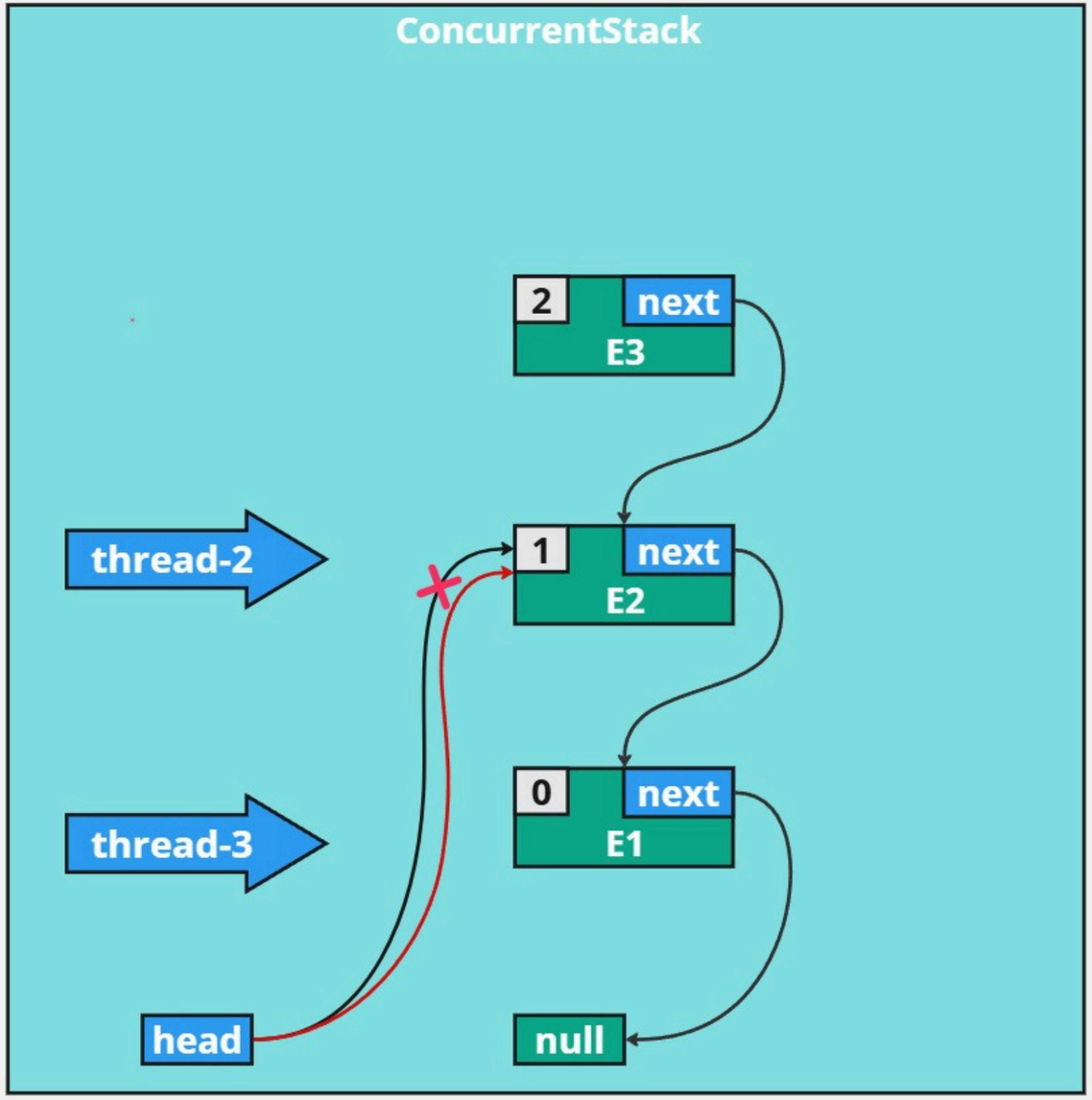
```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

thread-3

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2



thread-1

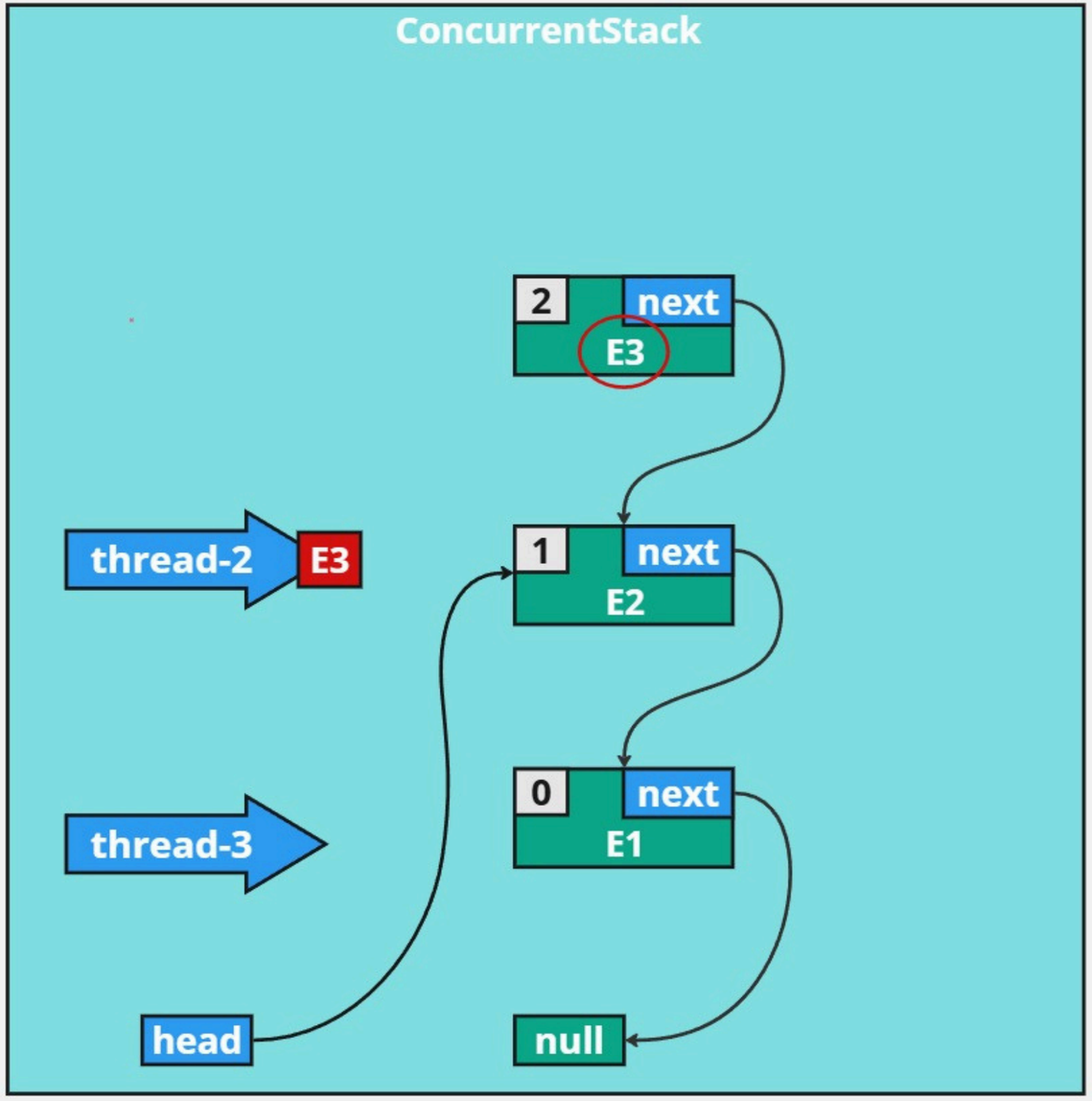
```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-3

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```



thread-1

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2

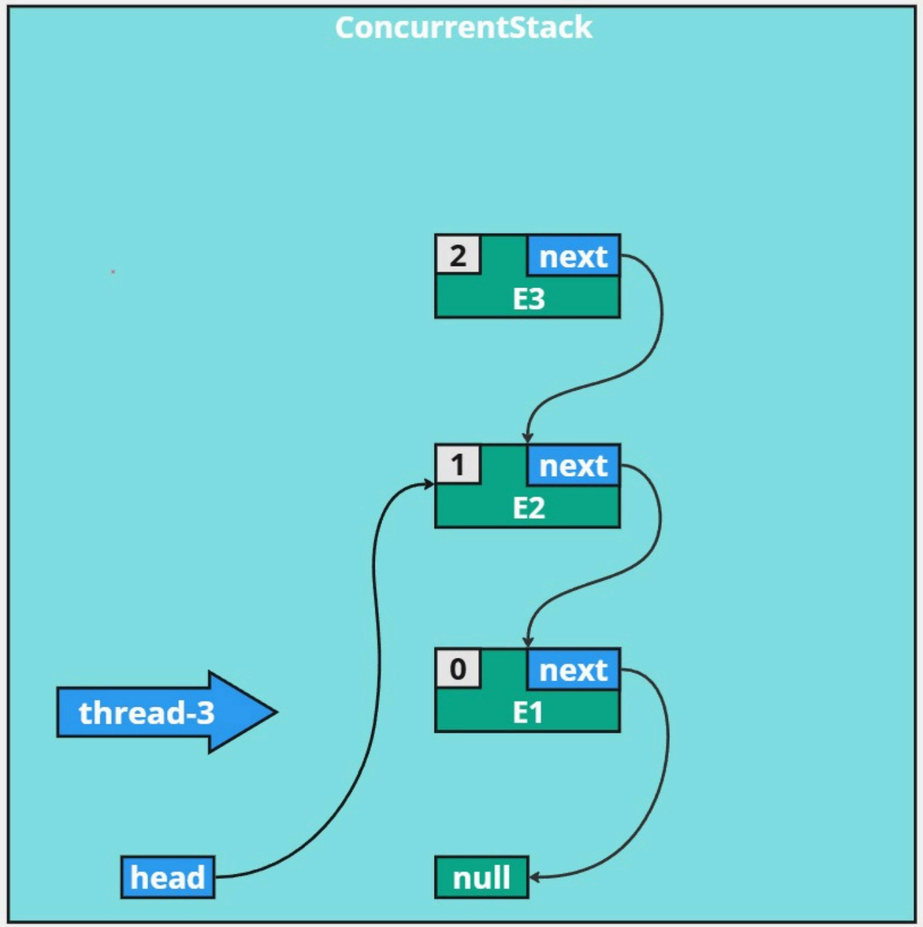
```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

thread-3

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2



thread-1

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2

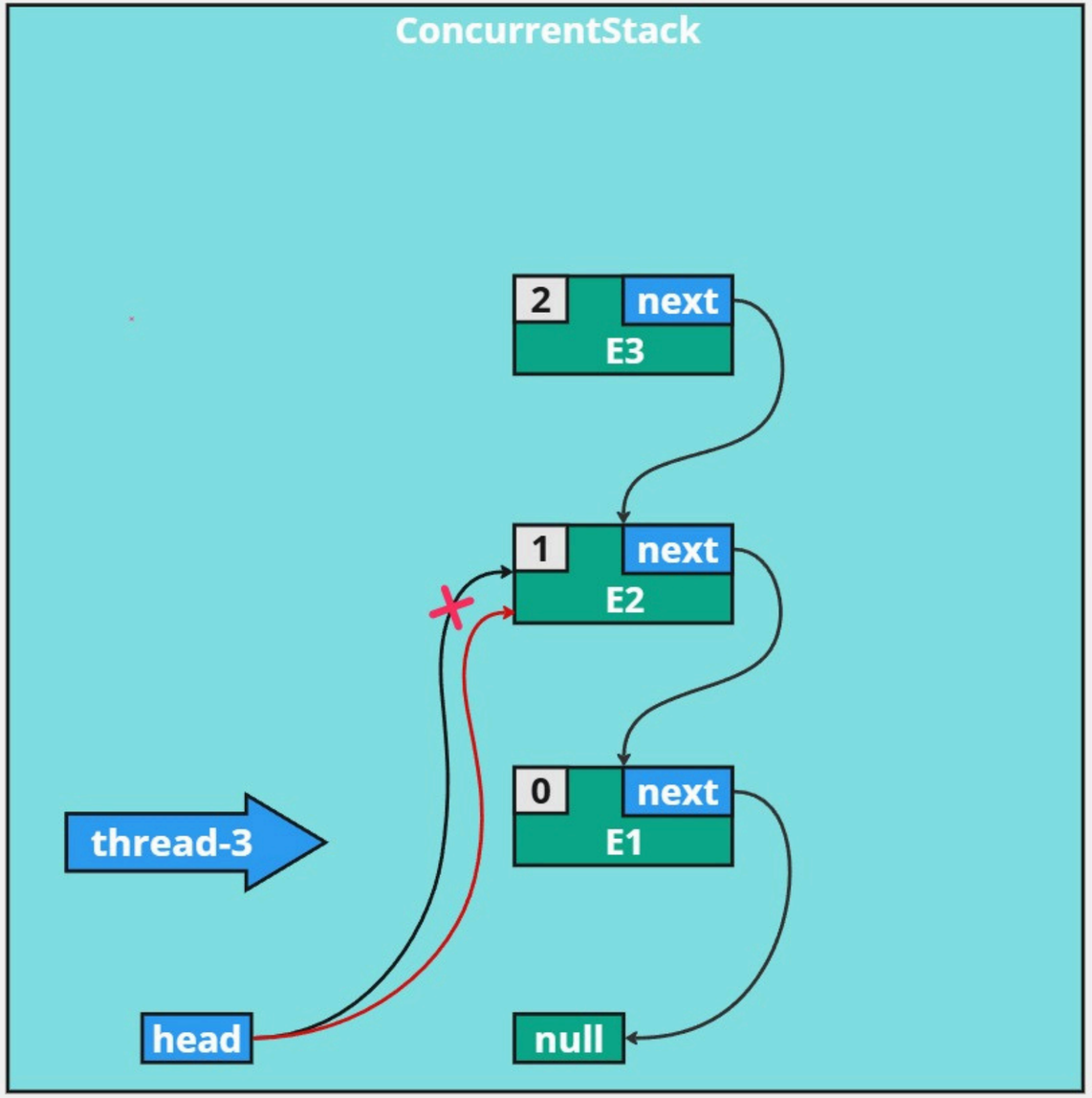
```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

thread-3

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2



thread-1

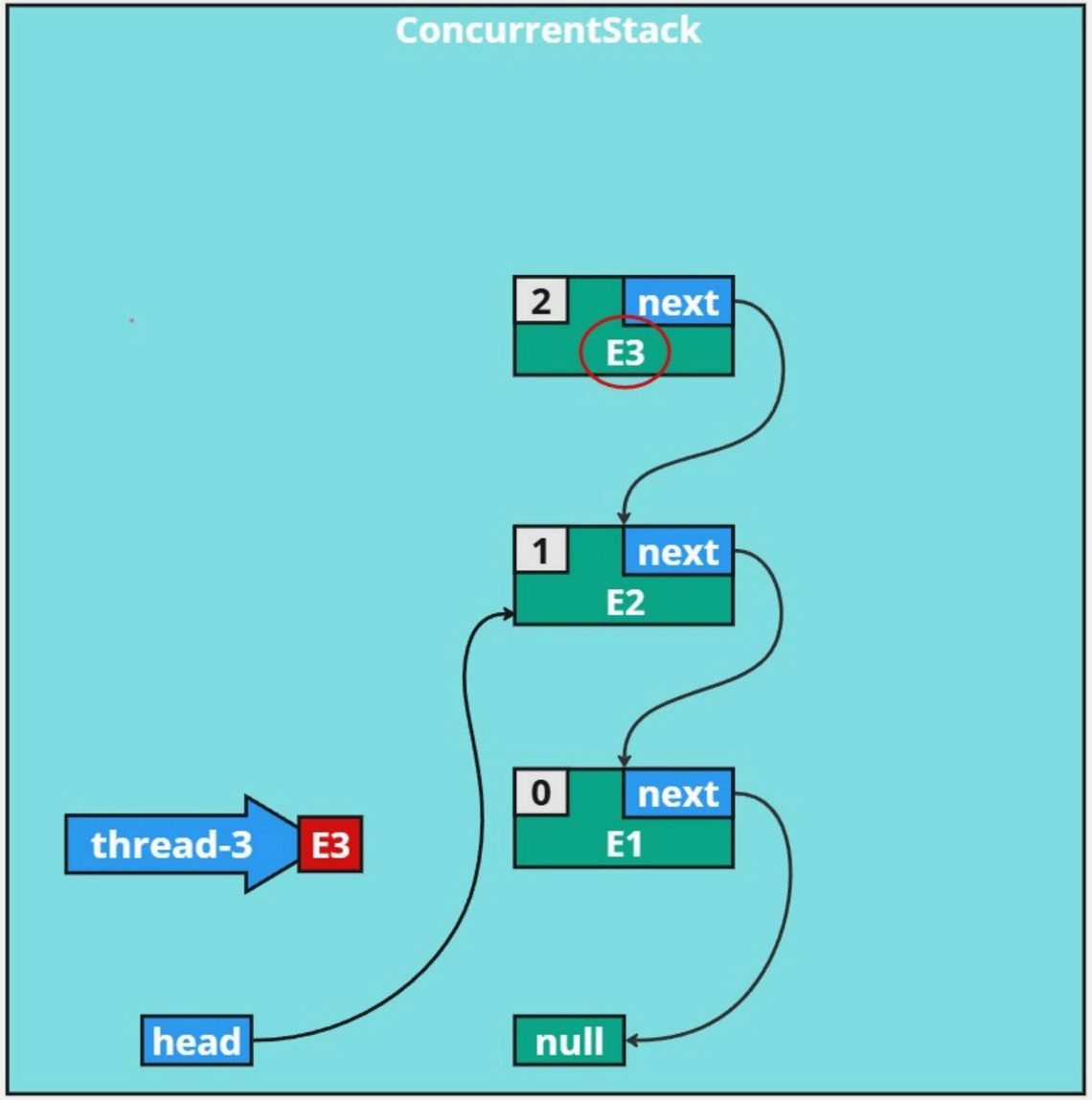

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2


```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-3

```
public T pop() {                                    2
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```



thread-1

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

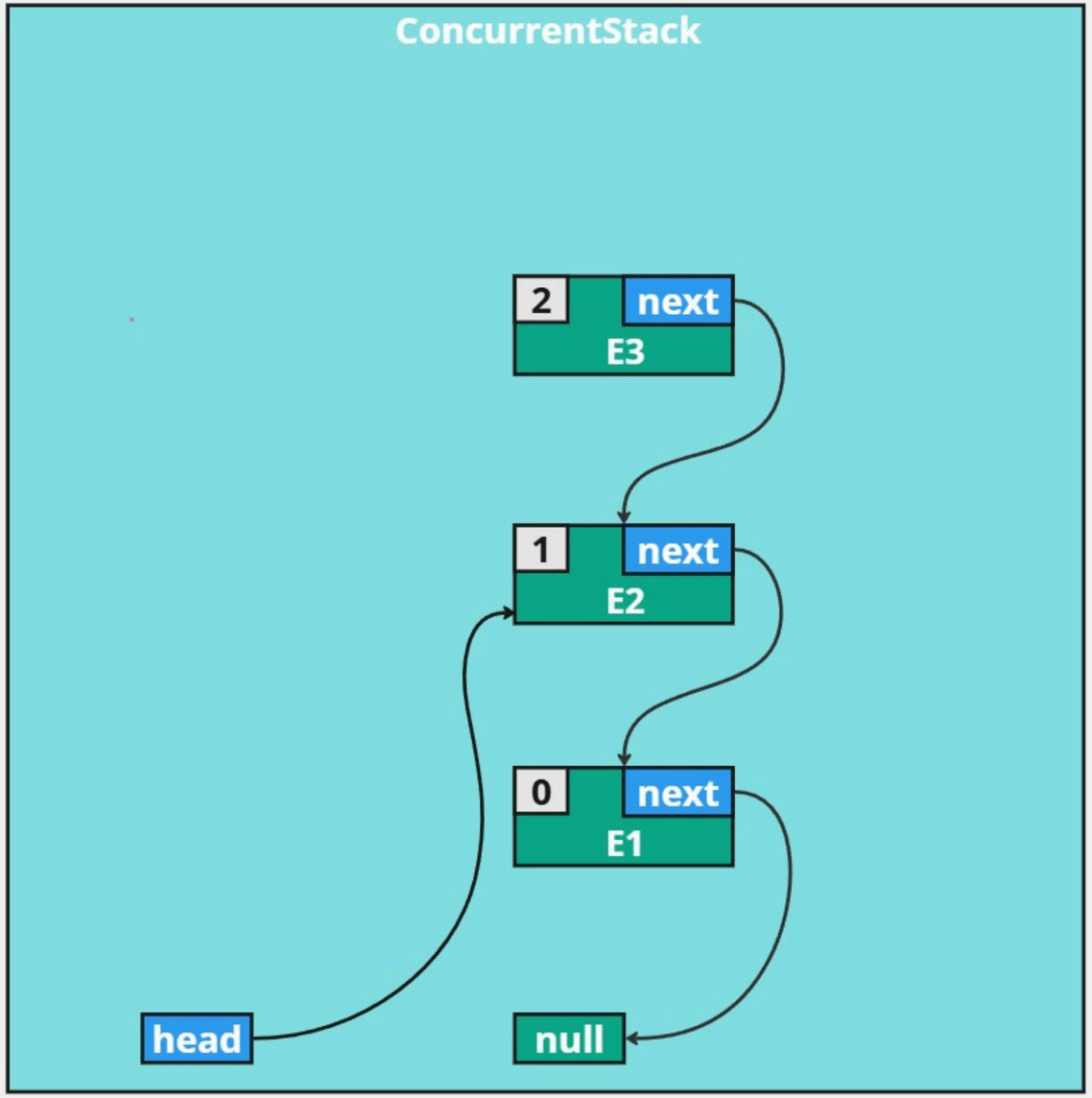
thread-2

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-3

```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2



thread-1


```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

thread-2


```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

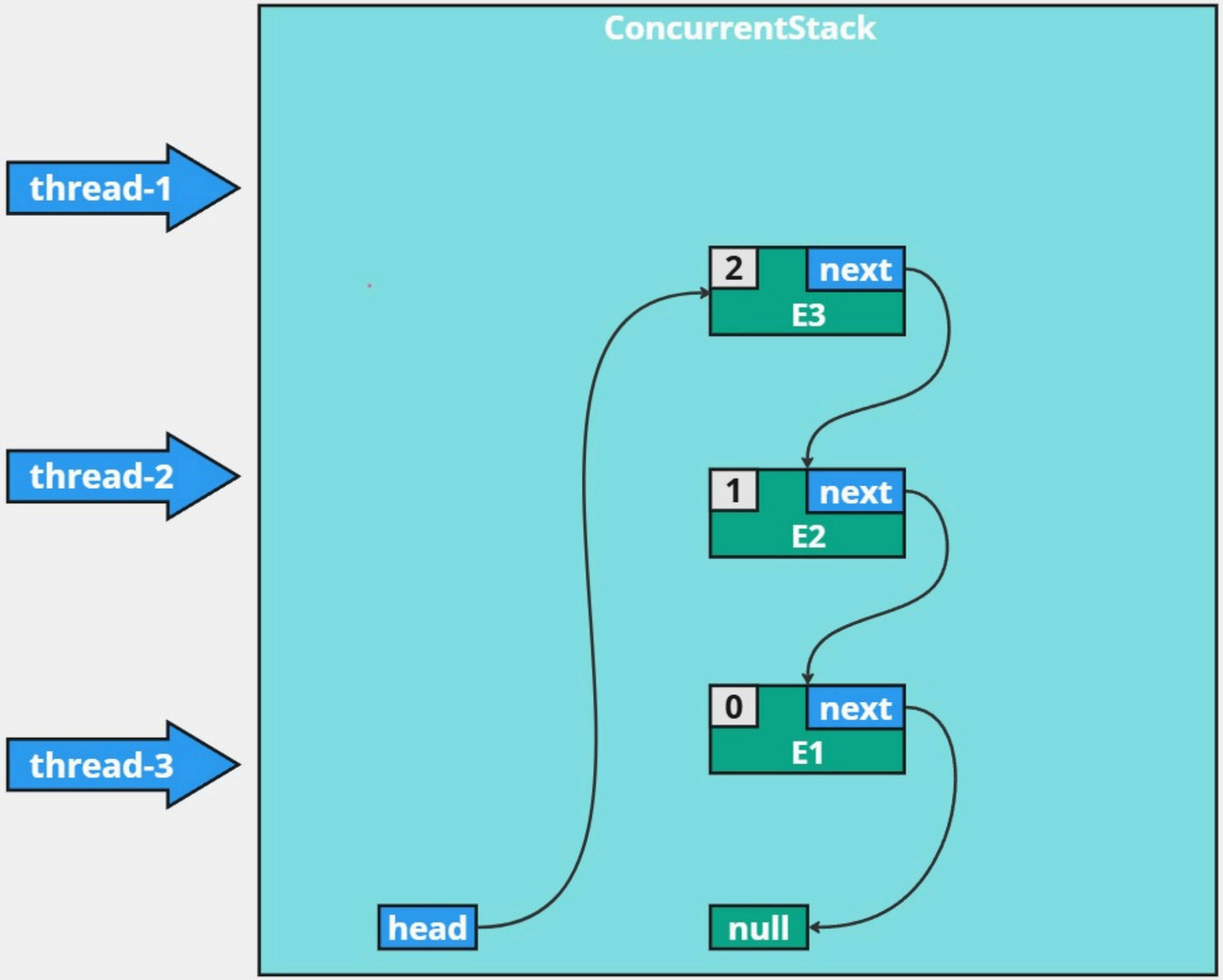
thread-3


```
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

2

```java
public T pop() {
    final Node<T> previousHead = head.get();
    head.set(previousHead.next);
    return previousHead.value;
}
```

```java
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```



thread-1

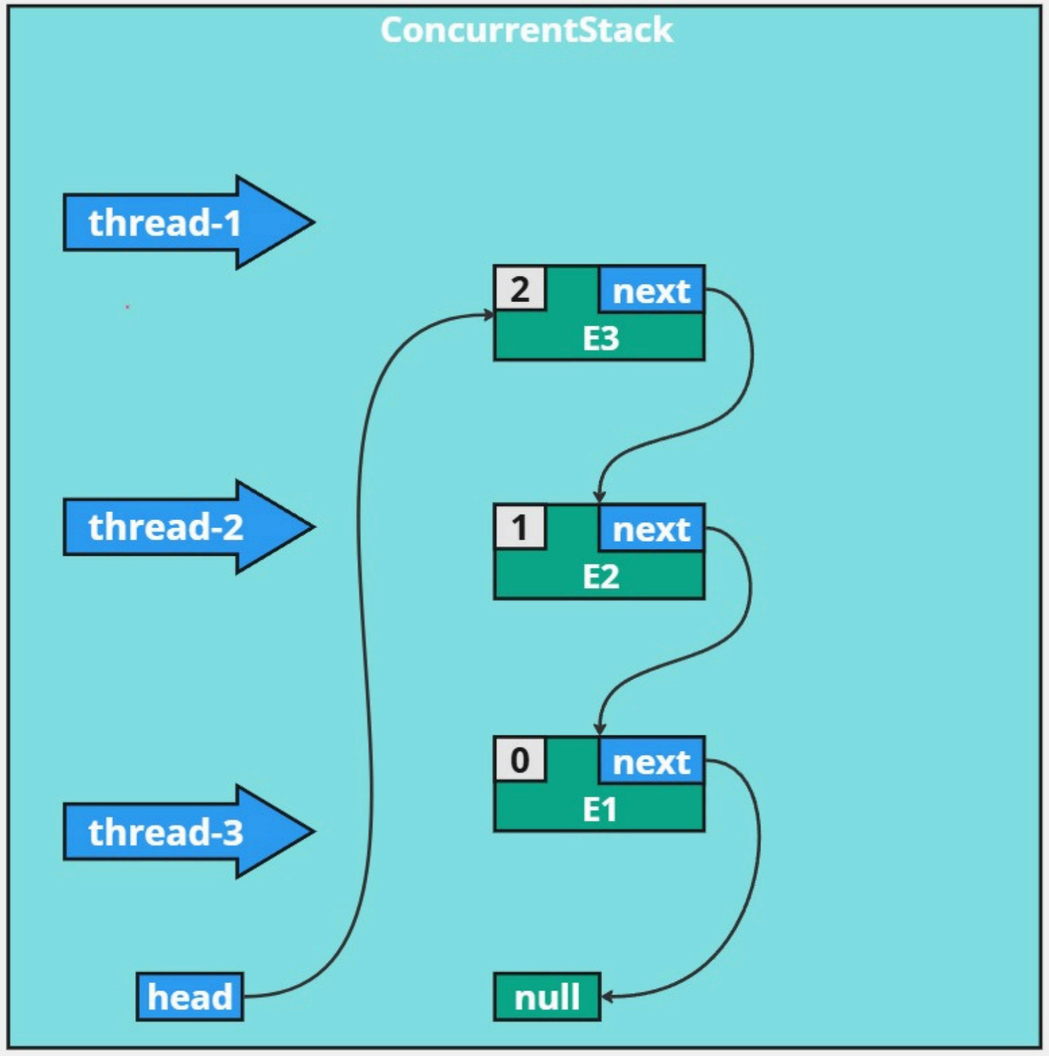
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```



thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2

thread-2

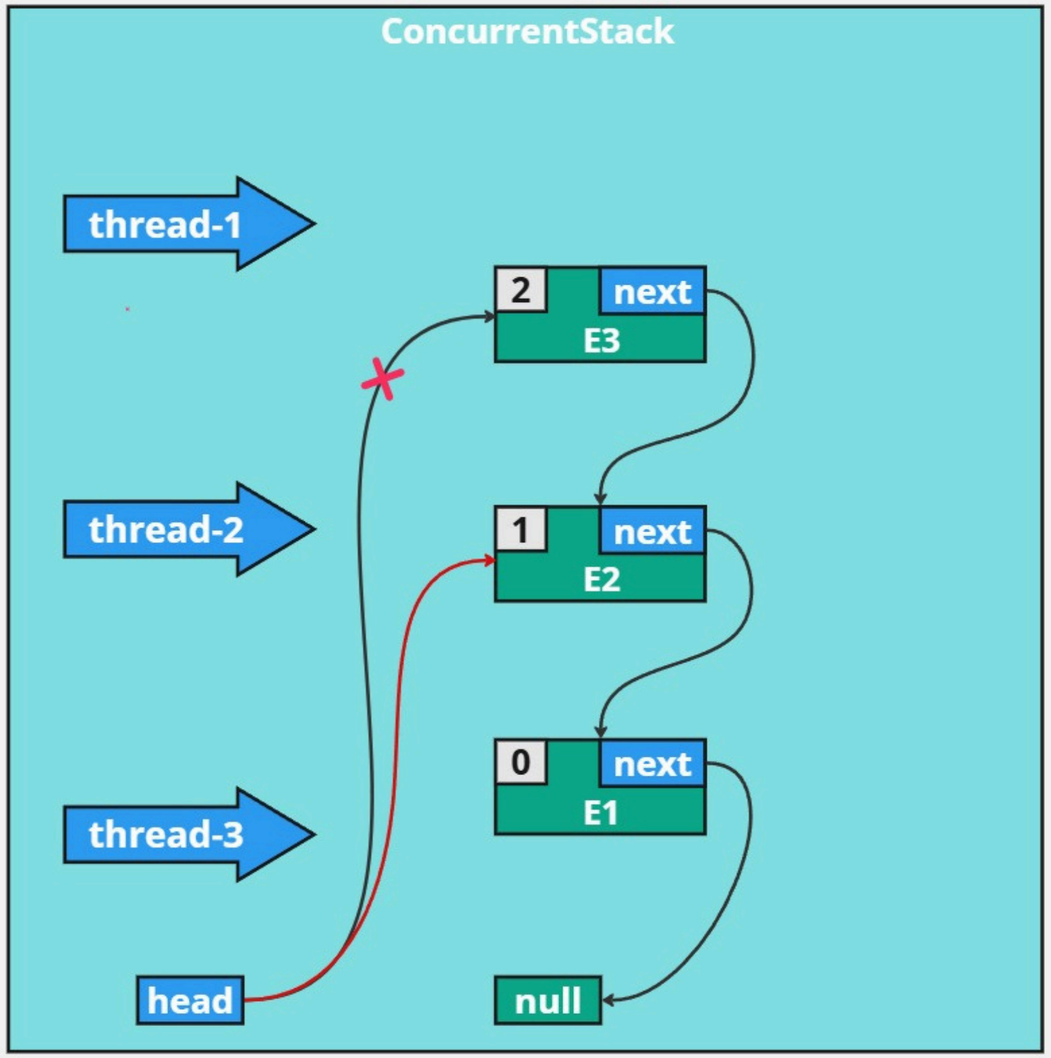
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2



thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2

thread-2

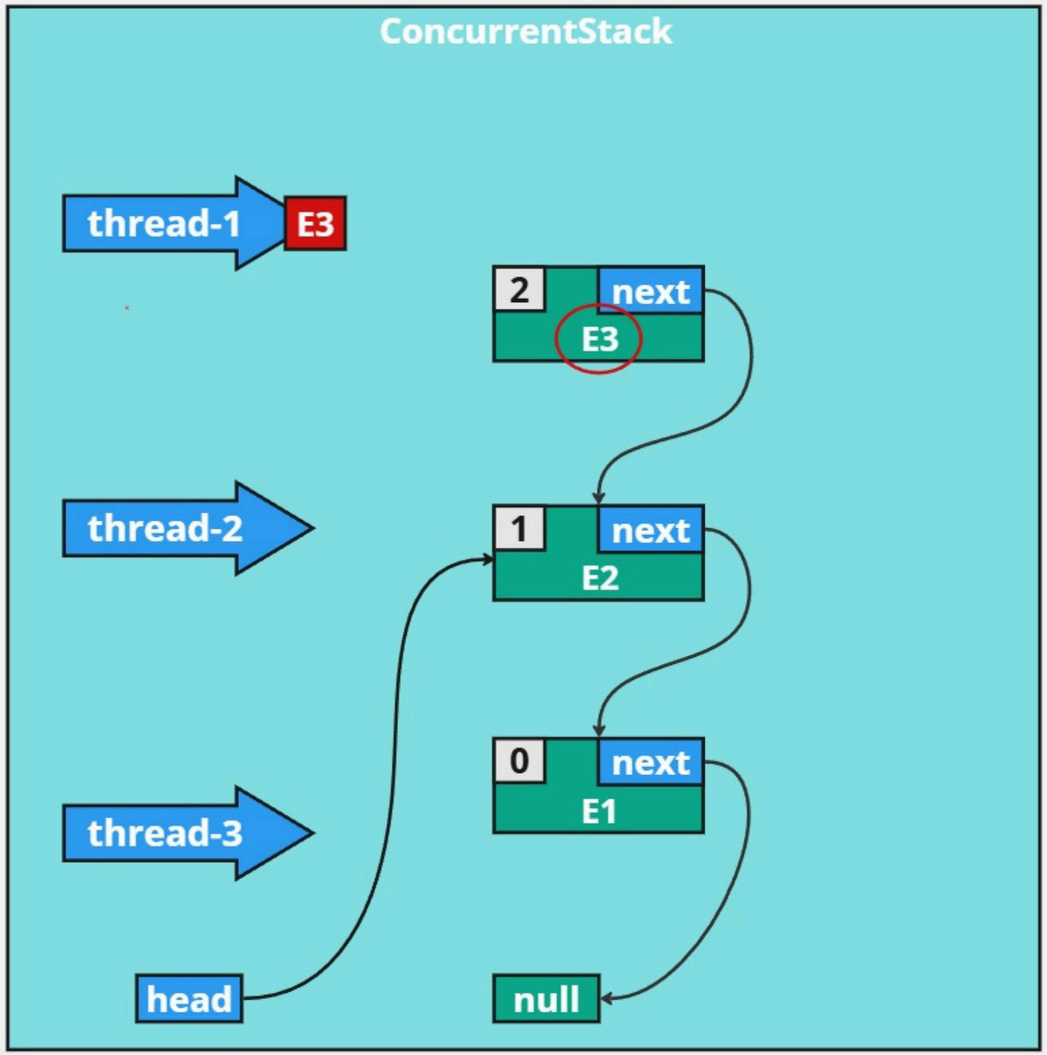
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2



thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2

thread-2

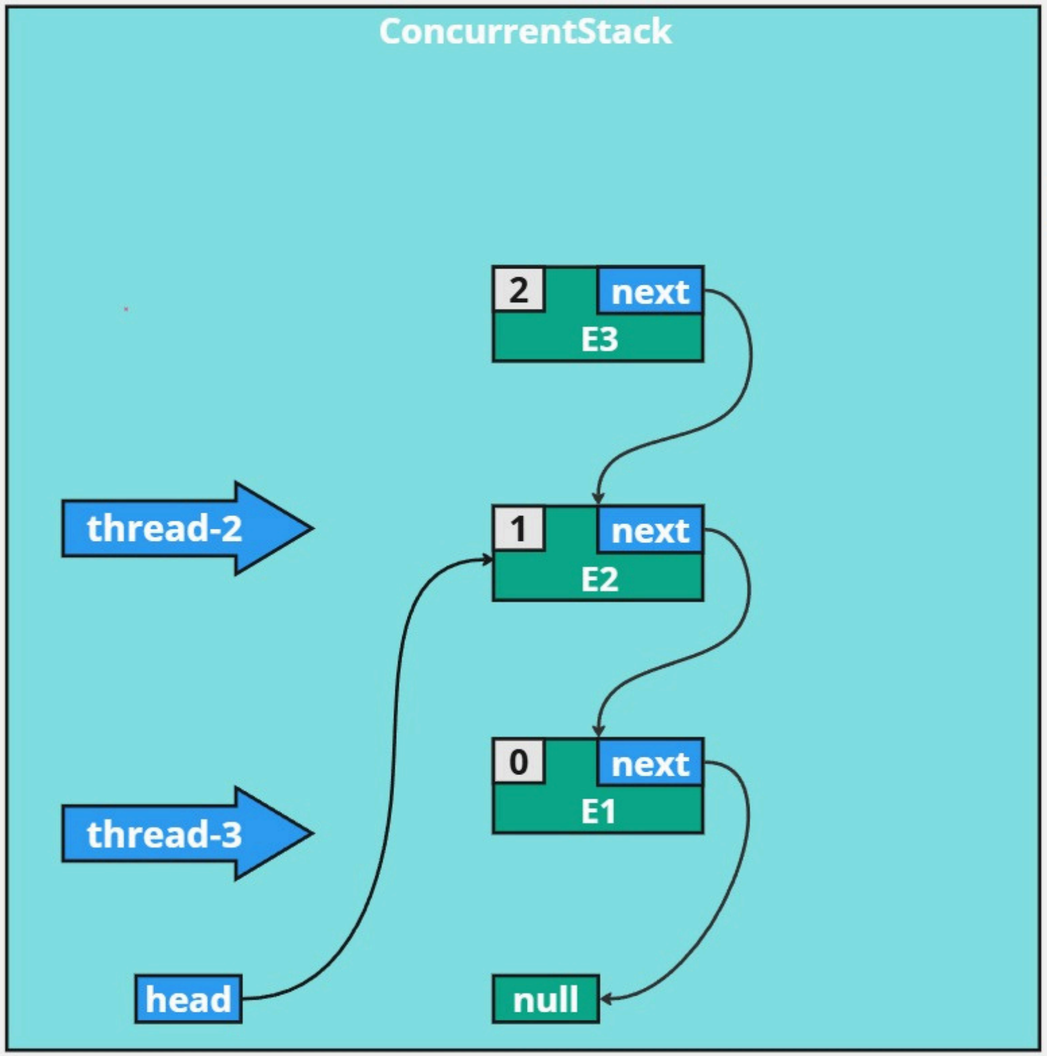
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

2



thread-1

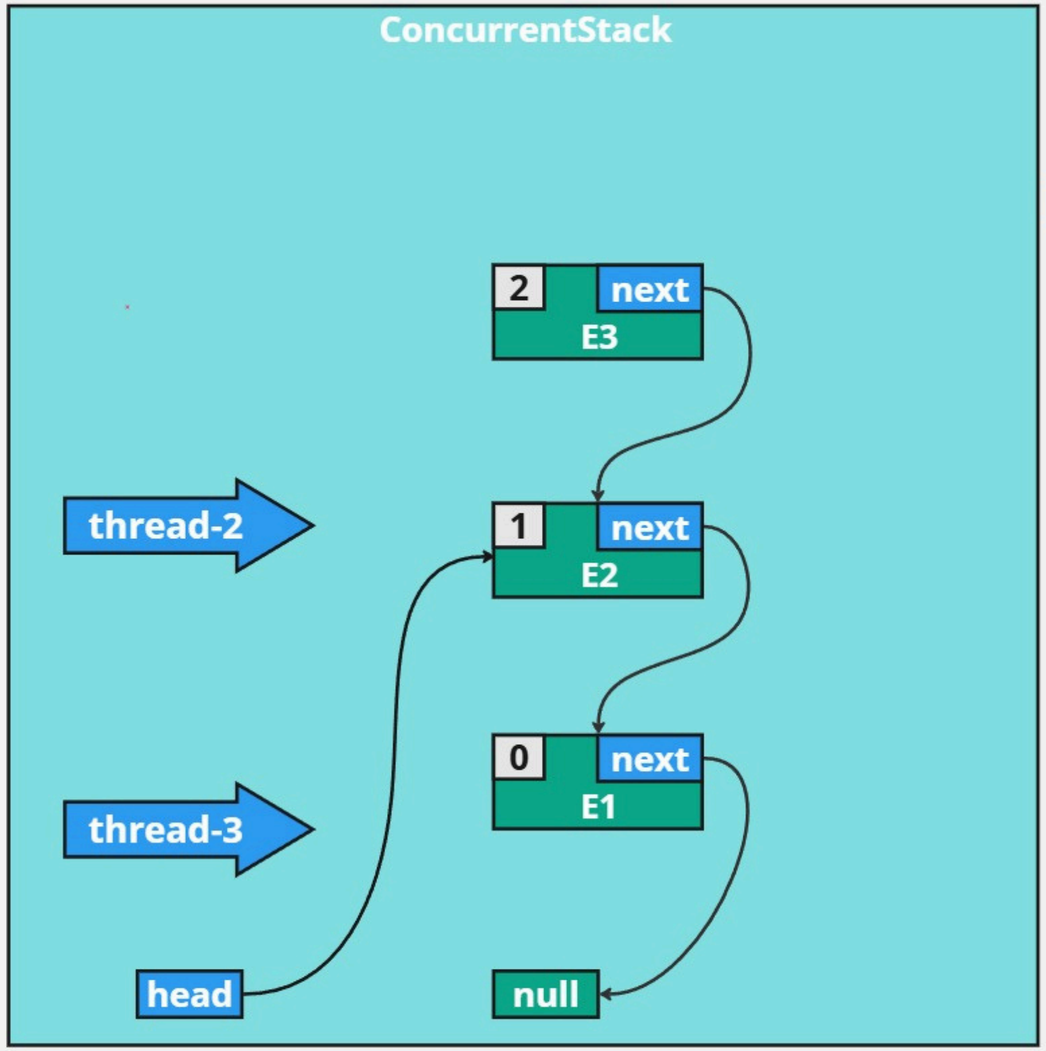
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```



thread-1 E3

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

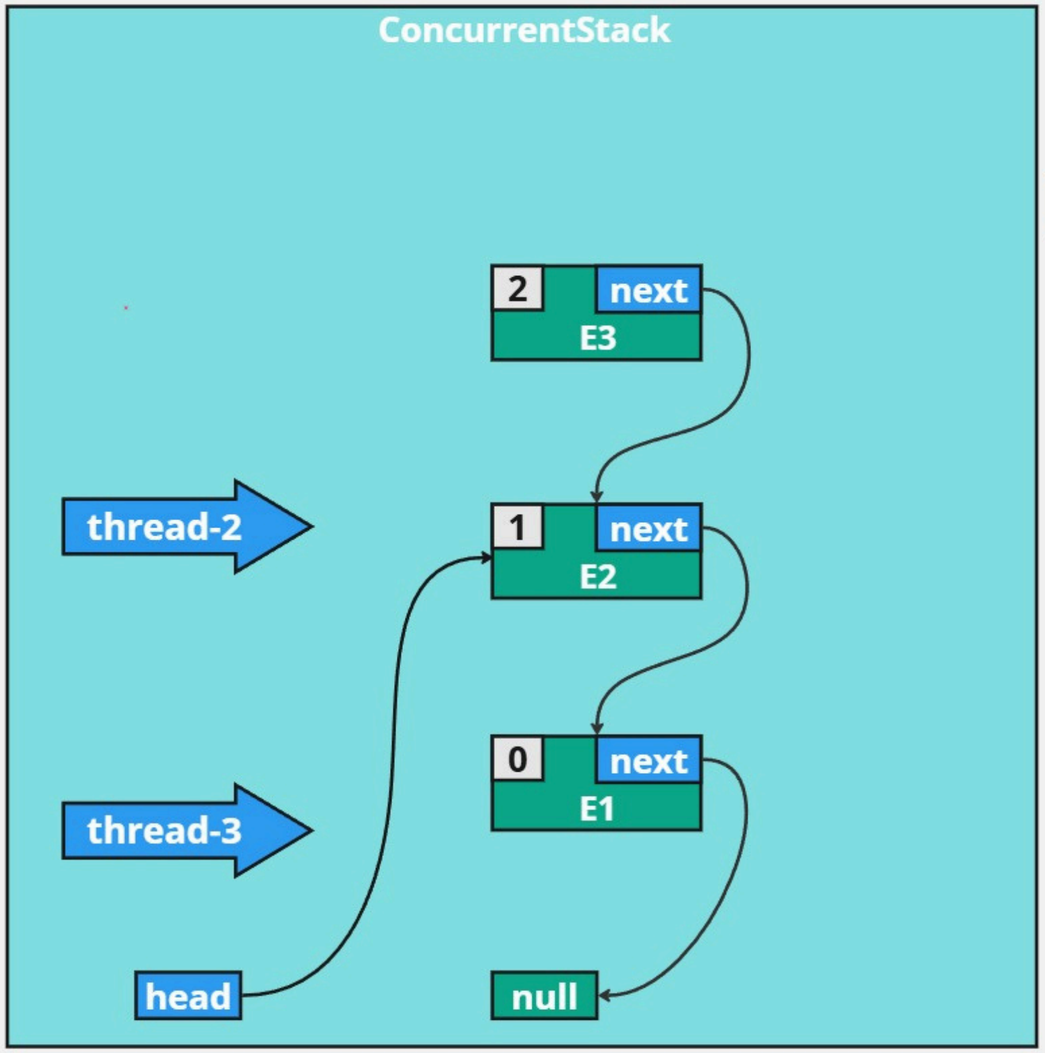
2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```



thread-1 E3

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

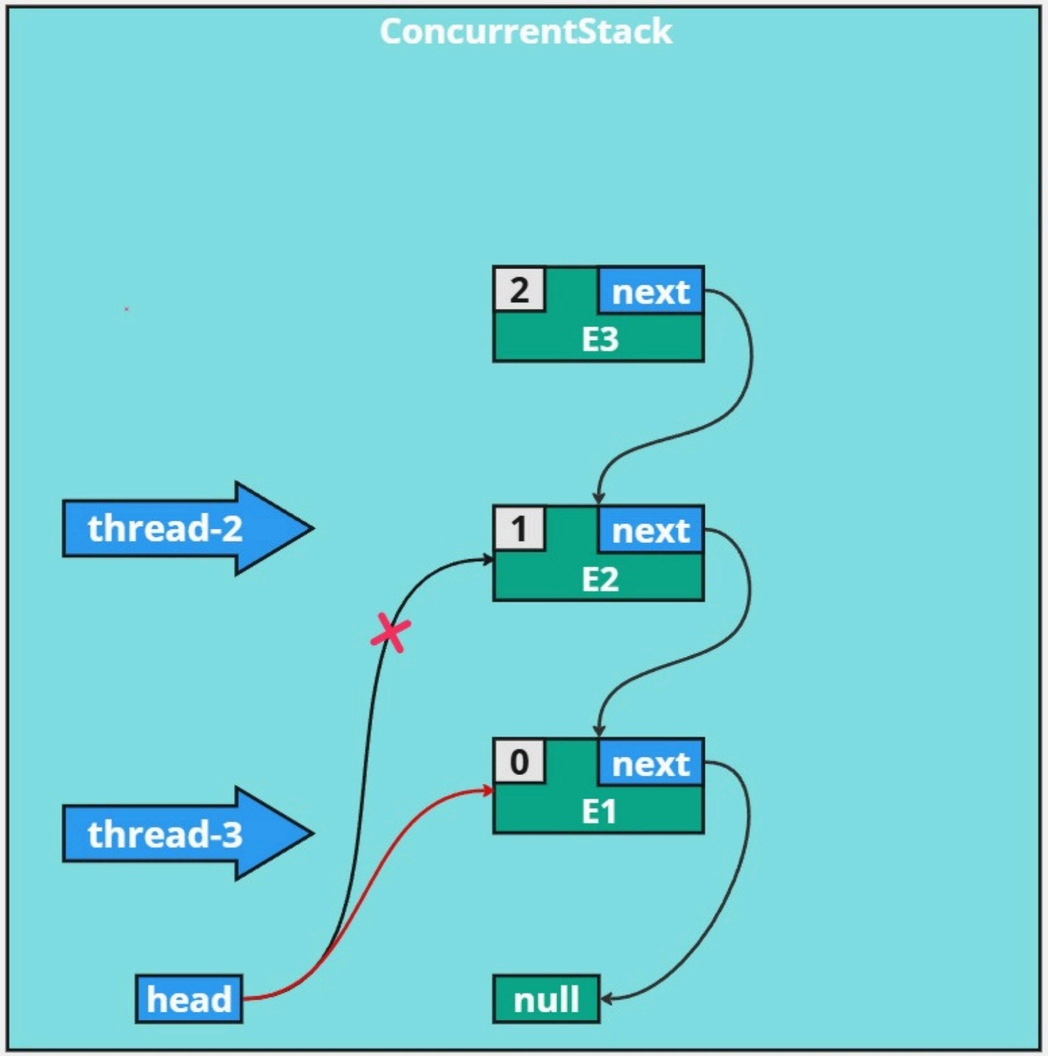
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1



thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

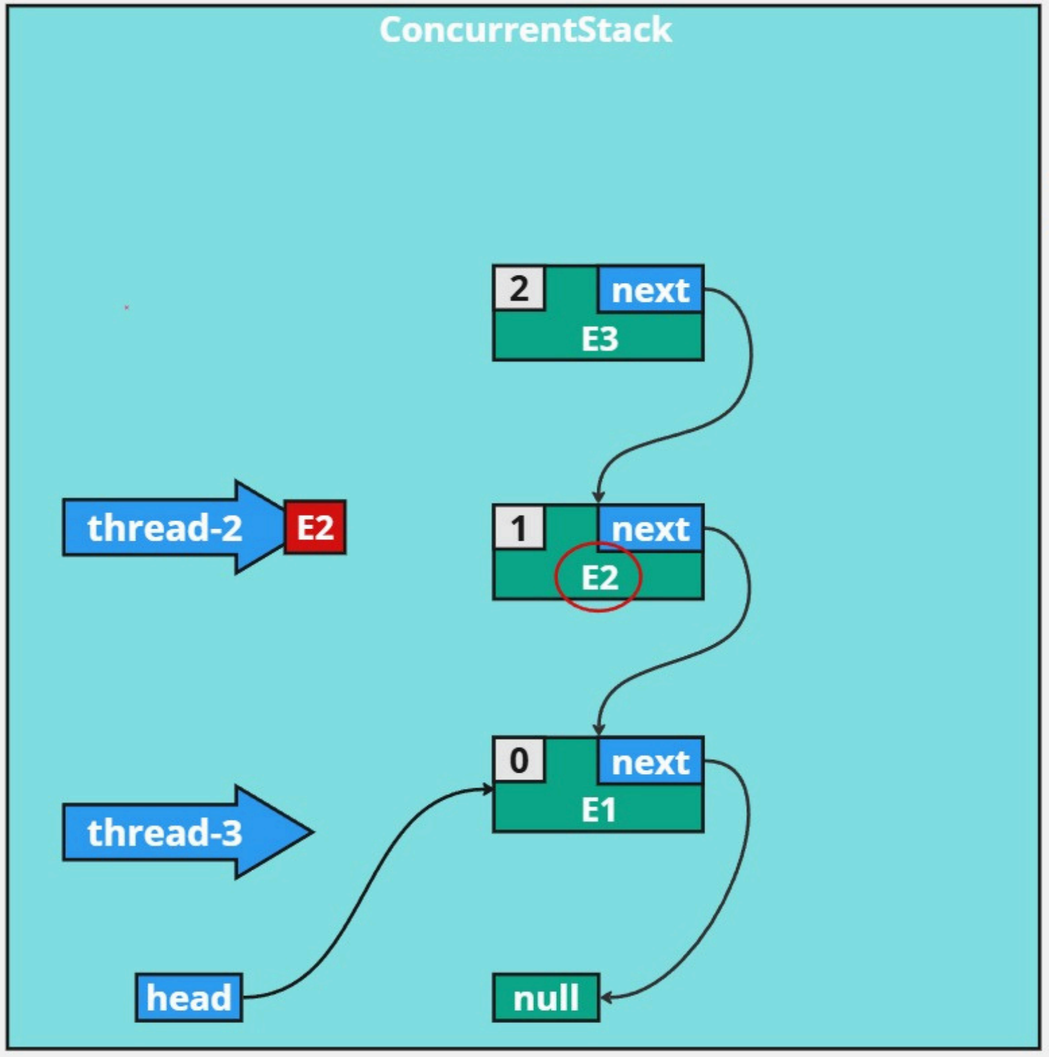
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1



thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

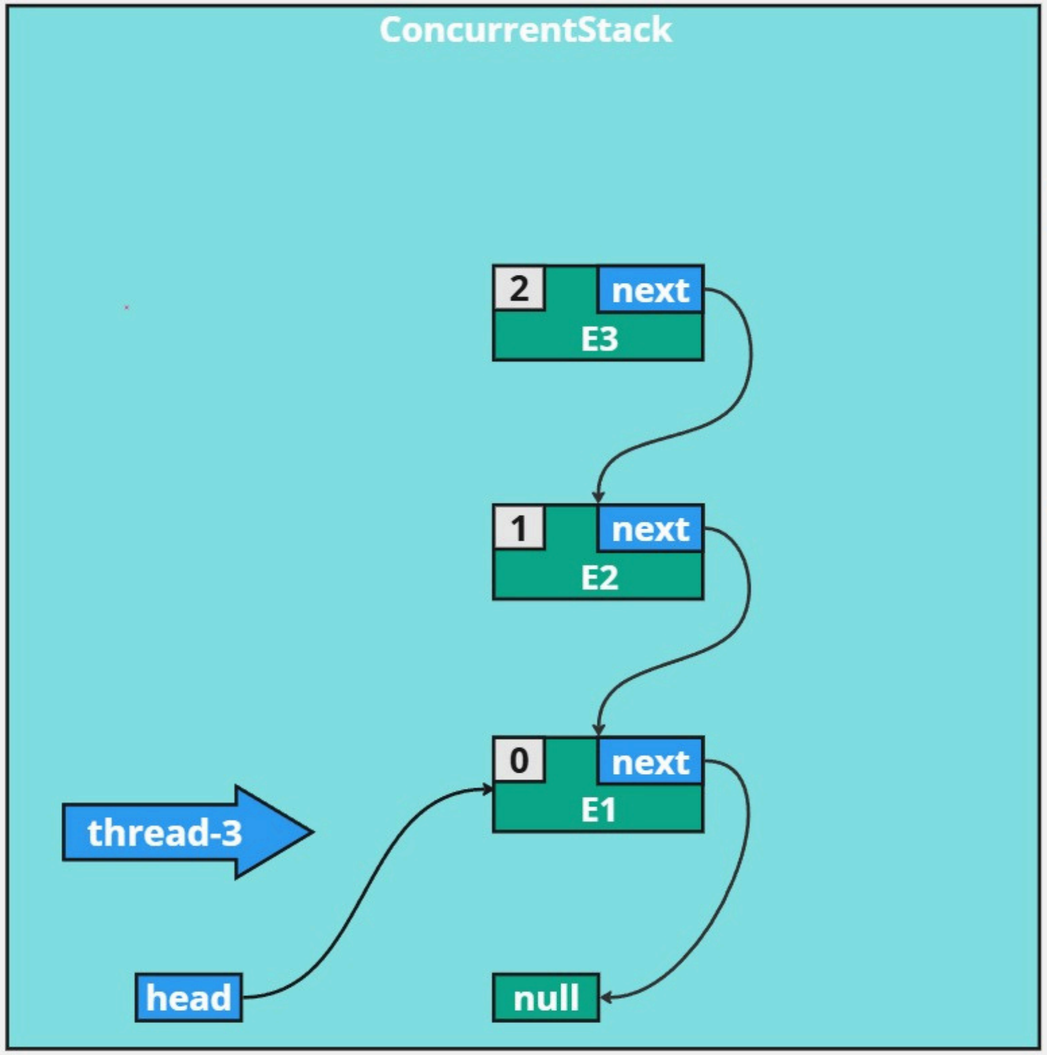
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1



thread-1 E3

thread-2 E2

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-2

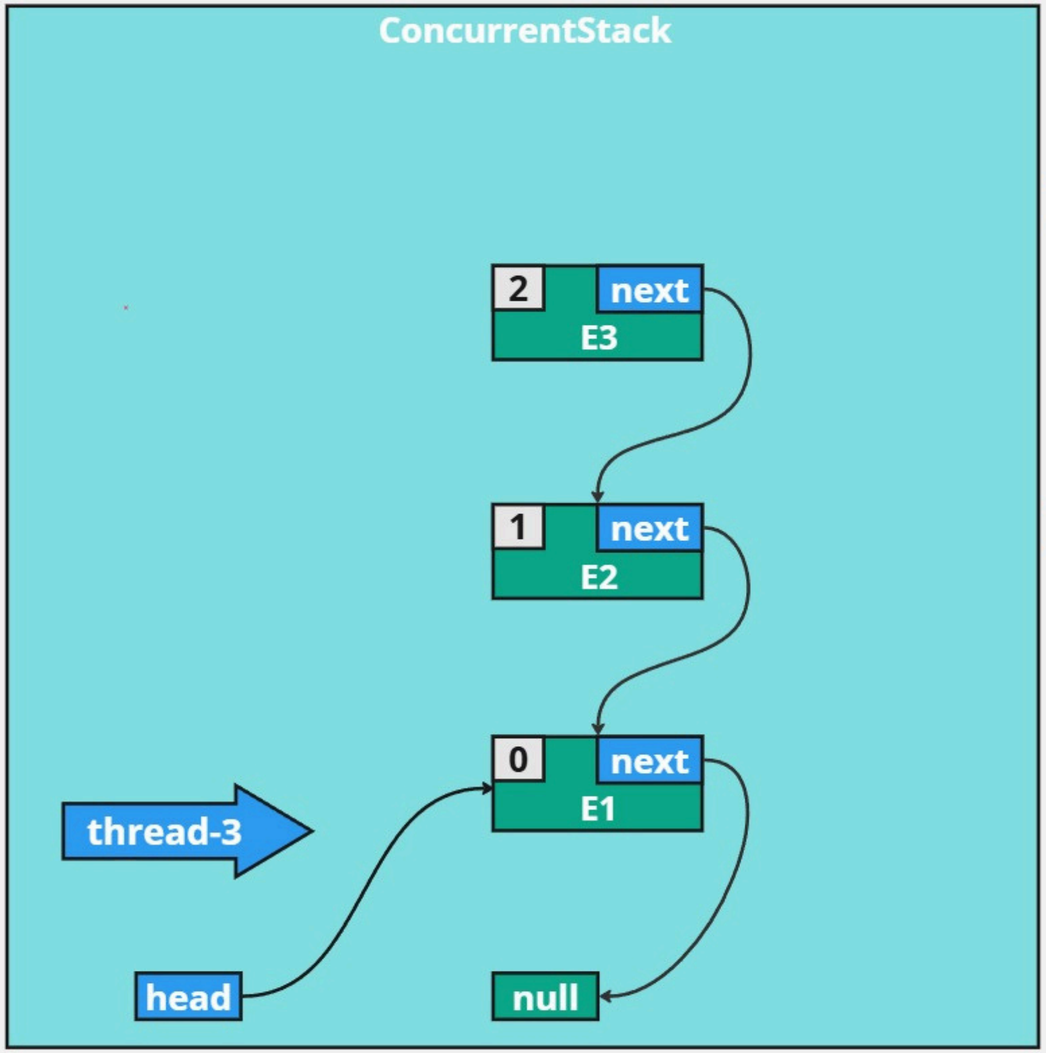
1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```



thread-1 E3

thread-2 E2

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

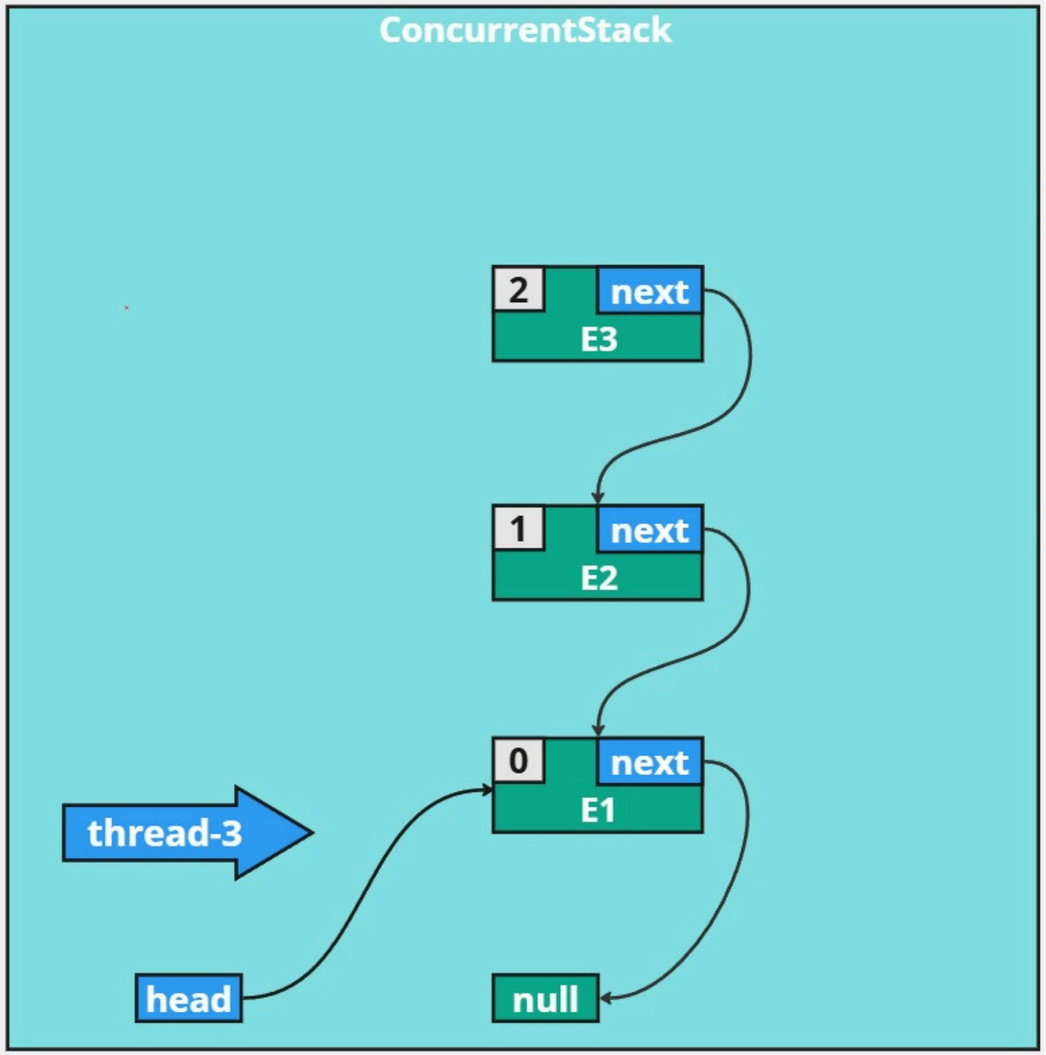
thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

1



thread-1 E3

thread-2 E2

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

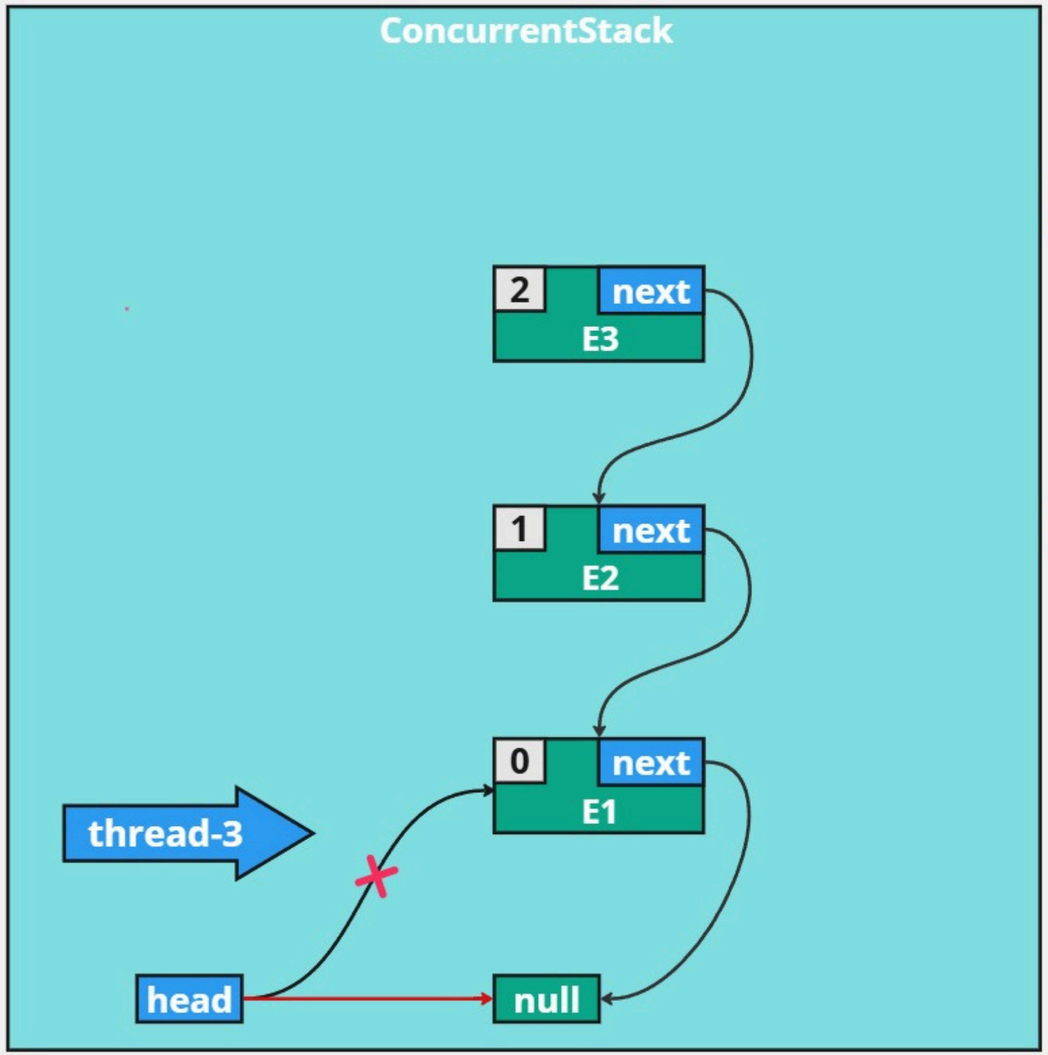
thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

0



thread-1 E3

thread-2 E2

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

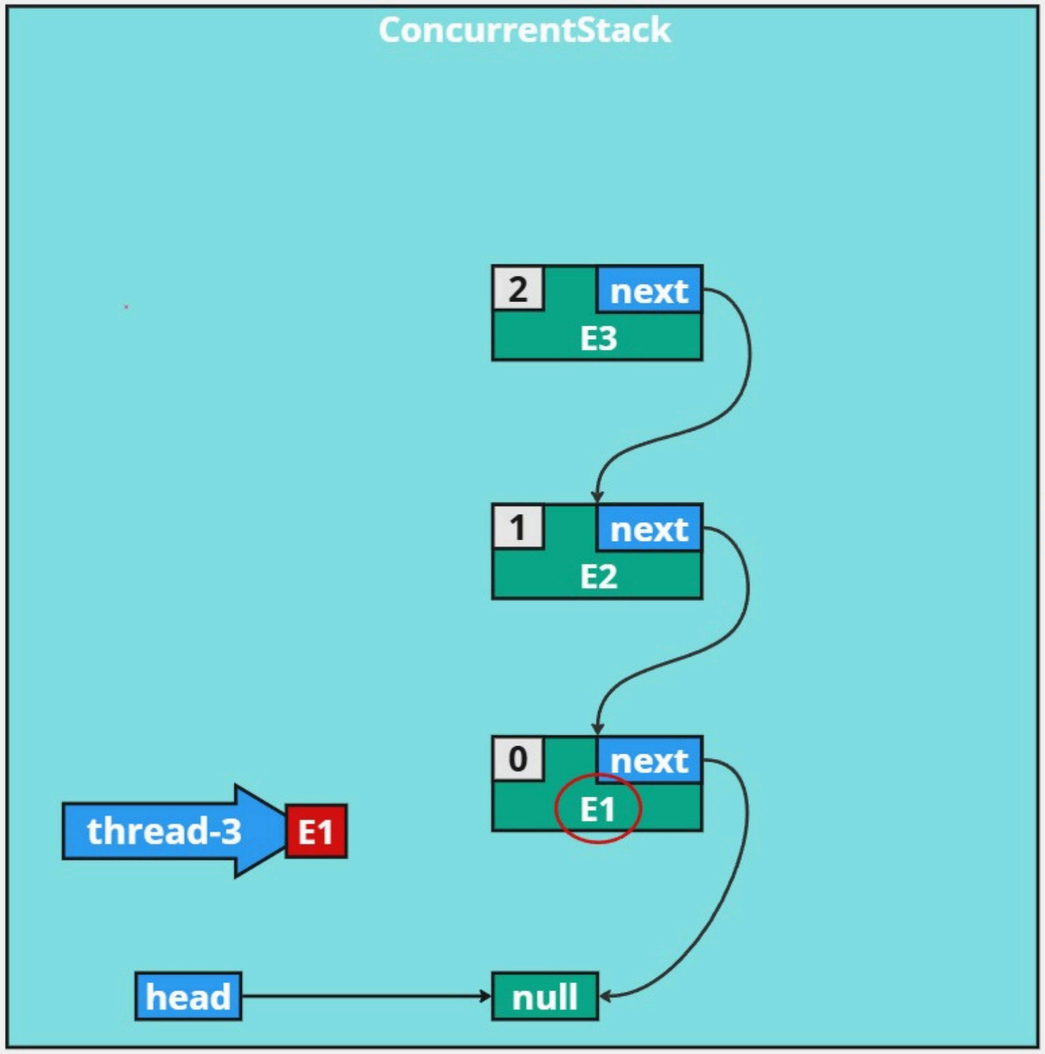
thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

0



thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

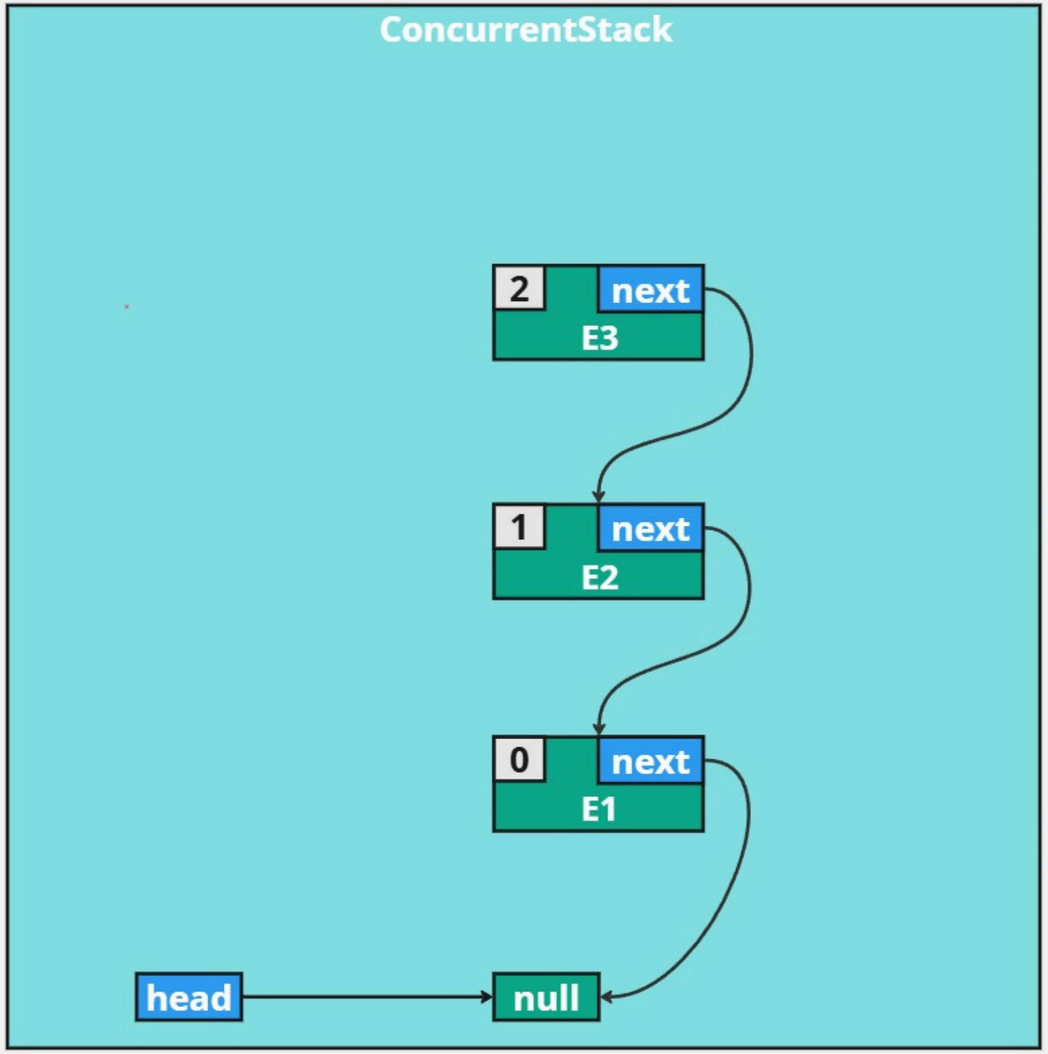
thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

0



thread-1 E3

thread-2 E2

thread-3 E1

thread-1

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

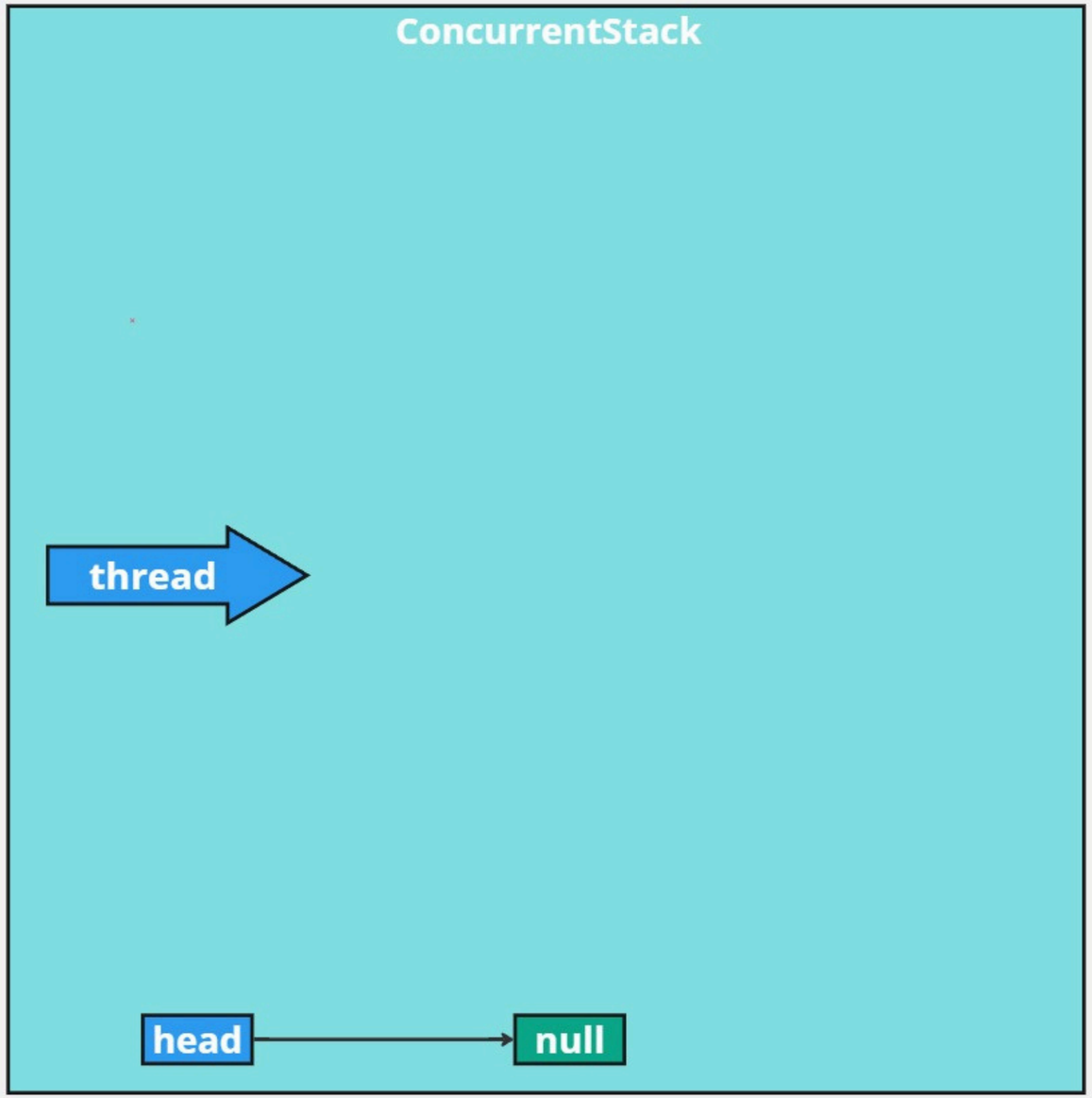
thread-2

```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

thread-3

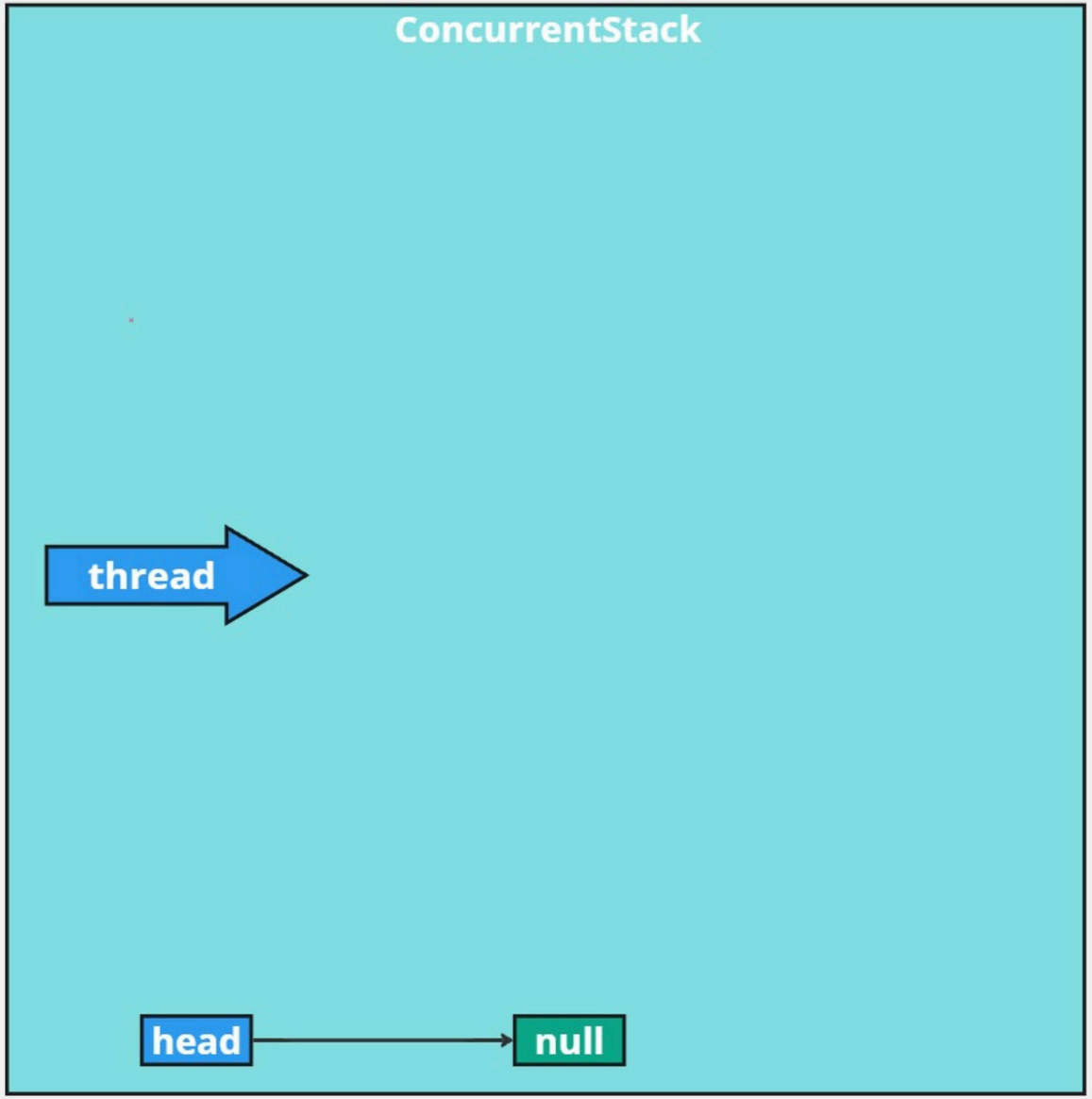
```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

0




```
public T pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return previousHead.value;
        }
    }
}
```

NullPointerException




```java
public Optional<T> pop() {
    while (true) {
        final Node<T> previousHead = head.get();
        if (previousHead == null) {
            return empty();
        }
        if (head.compareAndSet(previousHead, previousHead.next)) {
            return Optional.of(previousHead.value);
        }
    }
}
```