

Lesson 02 Demo 09

Merging Two Sorted Linked Lists

Objective: To write a function that merges two sorted linked lists into a single sorted linked list by splicing together their nodes, which helps understand merging techniques used in algorithms for efficient list manipulation

Tools required: Visual Studio Code (VS Code) and JavaScript

Prerequisites: Completion of Lesson 02 Demo 01

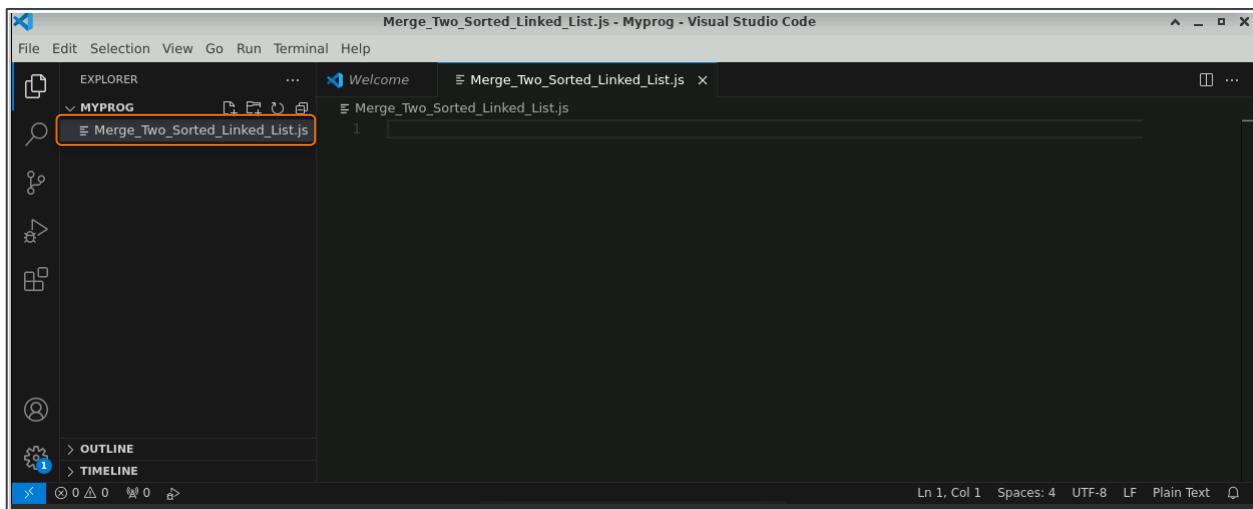
Steps to be followed:

1. Create a JavaScript file and execute it

Step 1: Create a JavaScript file and execute it

1.1 Open the Visual Studio Code editor and create a JavaScript file named

Merge_Two_Sorted_Linked_List.js



1.2 Add the following code to the file:

```
class ListNode {  
    constructor(value) {  
        this.value = value;  
        this.next = null;  
    }  
}  
  
function mergeSortedLists(l1, l2) {  
    let dummyHead = new ListNode(0);  
    let current = dummyHead;  
  
    while (l1 !== null && l2 !== null) {  
        if (l1.value < l2.value) {  
            current.next = l1;  
            l1 = l1.next;  
        } else {  
            current.next = l2;  
            l2 = l2.next;  
        }  
        current = current.next;  
    }  
  
    // At least one of l1 and l2 can still have nodes at this point, so connect  
    // the non-null list to the end of the merged list.  
    current.next = l1 === null ? l2 : l1;  
  
    return dummyHead.next;  
}
```

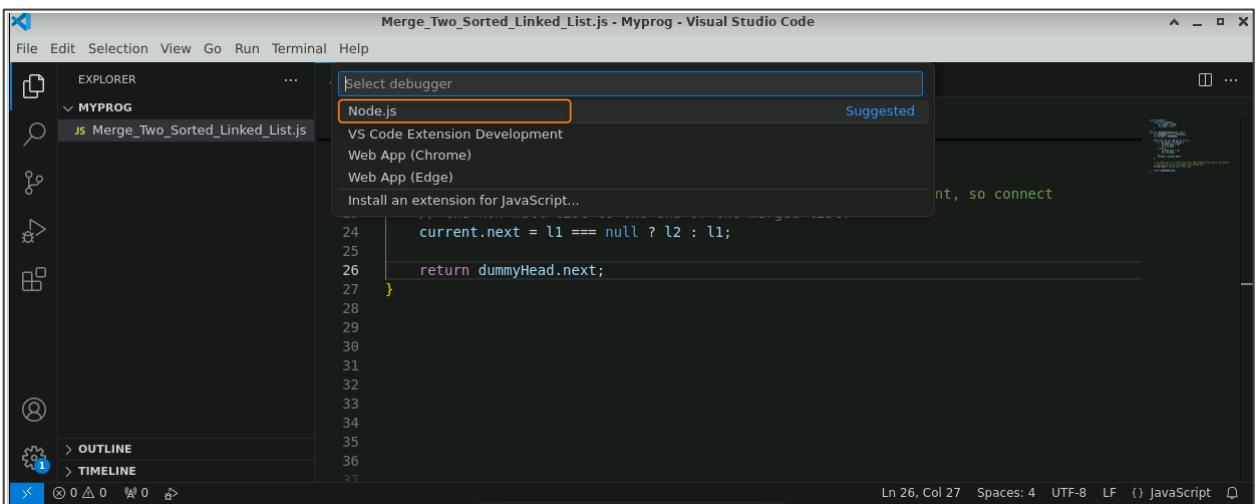
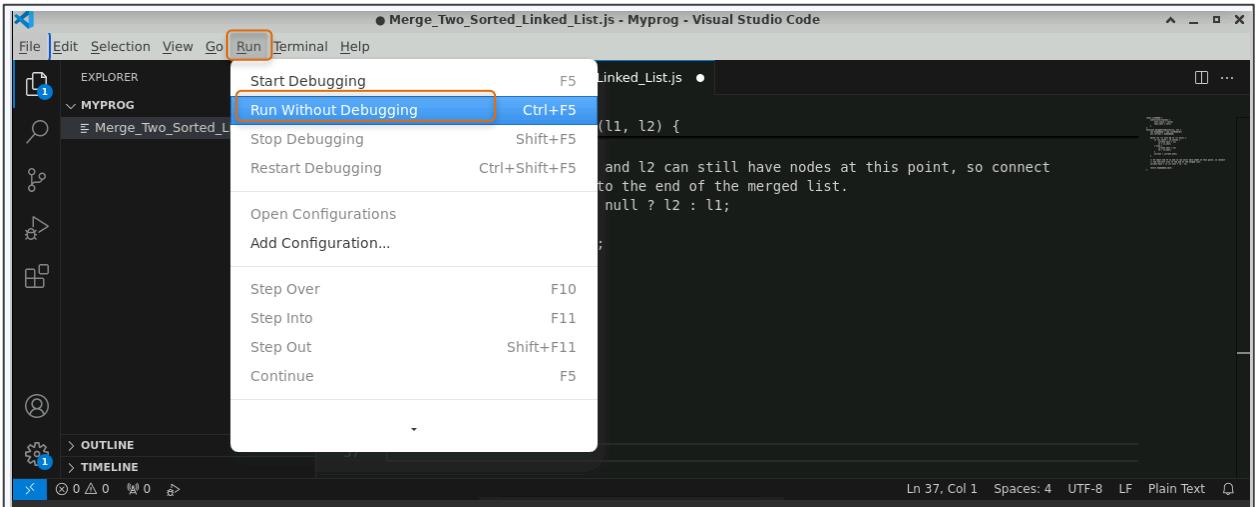
A screenshot of Visual Studio Code showing the file `Merge_Two_Sorted_Linked_List.js`. The code defines a `ListNode` class and a `mergeSortedLists` function. The `ListNode` constructor initializes a node with a value and a `next` pointer. The `mergeSortedLists` function creates a dummy head node and iterates through both lists, connecting nodes from the smaller list to the merged list until one list is exhausted. A red oval highlights the first few lines of the function definition.

```
1 class ListNode {
2     constructor(value) {
3         this.value = value;
4         this.next = null;
5     }
6 }
7 function mergeSortedLists(l1, l2) {
8     let dummyHead = new ListNode(0);
9     let current = dummyHead;
10
11     while (l1 !== null && l2 !== null) {
12         if (l1.value < l2.value) {
13             current.next = l1;
14             l1 = l1.next;
15         } else {
16             current.next = l2;
17             l2 = l2.next;
18         }
19         current = current.next;
}
```

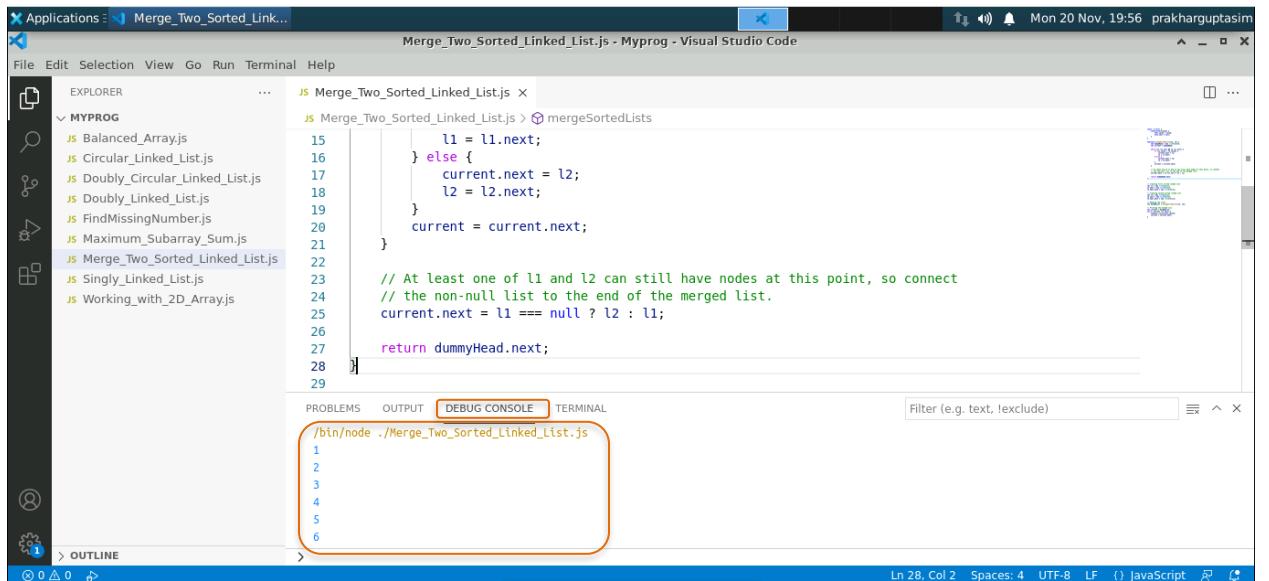
A screenshot of Visual Studio Code showing the same file `Merge_Two_Sorted_Linked_List.js`. The `mergeSortedLists` function now includes a final block of code to handle the case where one list is empty. It checks if `l1` is null and connects the remaining nodes of `l2` to the end of the merged list. A red oval highlights this final block of code.

```
1 class ListNode {
2     constructor(value) {
3         this.value = value;
4         this.next = null;
5     }
6 }
7 function mergeSortedLists(l1, l2) {
8     let dummyHead = new ListNode(0);
9     let current = dummyHead;
10
11     while (l1 !== null && l2 !== null) {
12         if (l1.value < l2.value) {
13             current.next = l1;
14             l1 = l1.next;
15         } else {
16             current.next = l2;
17             l2 = l2.next;
18         }
19         current = current.next;
20
21     // At least one of l1 and l2 can still have nodes at this point, so connect
22     // the non-null list to the end of the merged list.
23     current.next = l1 === null ? l2 : l1;
24
25     return dummyHead.next;
26
27 }
```

1.3 Click **Run** and then **Run Without Debugging**. Select **Node.js** to check the output in the DEBUG CONSOLE.



1.4 View the output in the **DEBUG CONSOLE** as shown below:



The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** Applications - Merge_Two_Sorted_Linked_List.js - Myprog - Visual Studio Code, Mon 20 Nov, 19:56 prakharguptasim
- File Explorer:** Shows a folder named "MYPROG" containing several JavaScript files: Balanced_Array.js, Circular_Linked_List.js, Doubly_Circular_Linked_List.js, Doubly_Linked_List.js, FindMissingNumber.js, Maximum_Subarray_Sum.js, Merge_Two_Sorted_Linked_List.js, Singly_Linked_List.js, and Working_with_2D_Arrays.js.
- Editor:** The file "Merge_Two_Sorted_Linked_List.js" is open, showing the code for merging two sorted linked lists. The code uses pointers l1 and l2 to traverse the lists and a current pointer to build the merged list.
- Output Tab:** The "DEBUG CONSOLE" tab is selected, showing the output of the program. The command run is "/bin/node ./Merge_Two_Sorted_Linked_List.js". The output shows the merged list: 1, 2, 3, 4, 5, 6.
- Status Bar:** Ln 28, Col 2, Spaces: 4, UTF-8, LF, JavaScript

By following these steps, you have efficiently combined two sorted linked lists into a single sorted list, which helps understand merging techniques used in algorithms for efficient list manipulation. This implementation is concise and leverages the existing order of the lists to minimize operations.