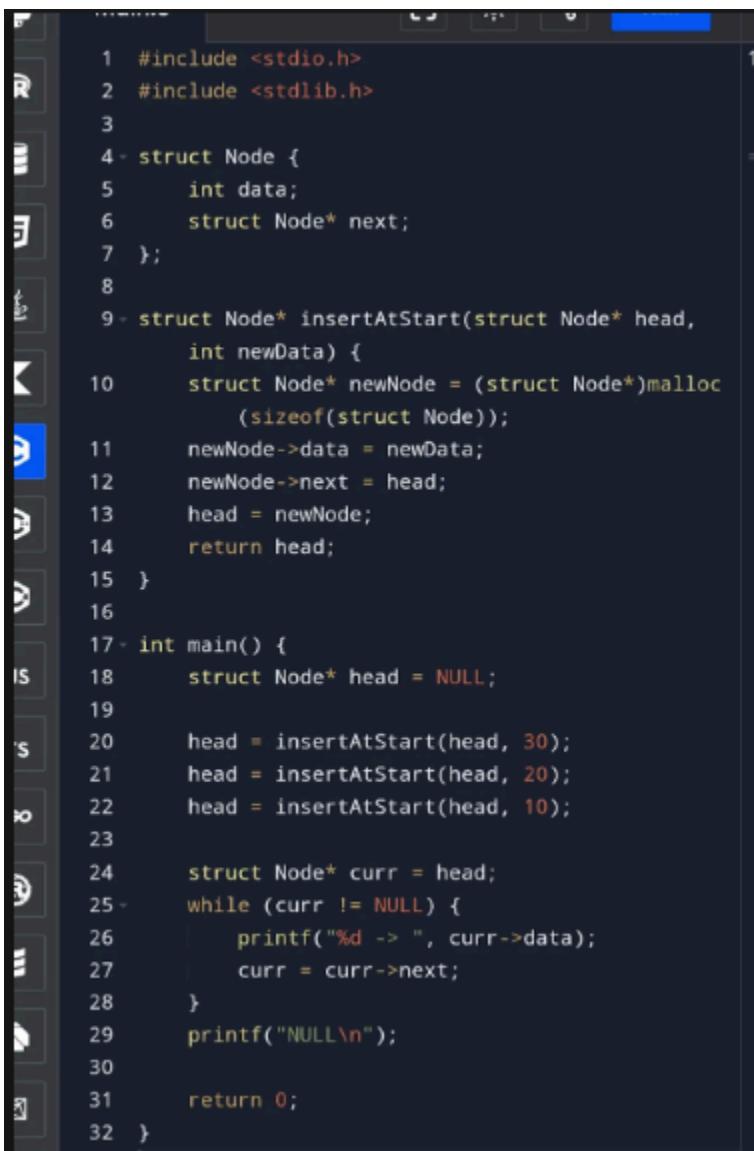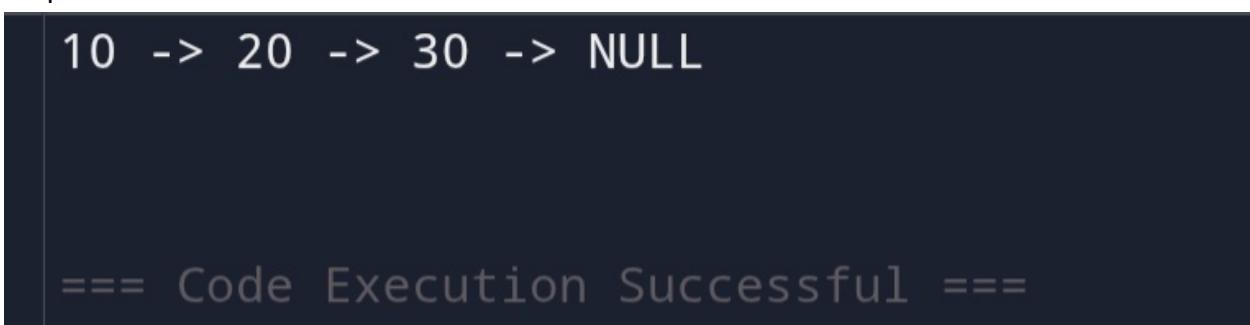Insertion at start:
Input:

```c
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node* next;
};

struct Node* insertAtStart(struct Node* head,
        int newData) {
    struct Node* newNode = (struct Node*)malloc
        (sizeof(struct Node));
    newNode->data = newData;
    newNode->next = head;
    head = newNode;
    return head;
}

int main() {
    struct Node* head = NULL;

    head = insertAtStart(head, 30);
    head = insertAtStart(head, 20);
    head = insertAtStart(head, 10);

    struct Node* curr = head;
    while (curr != NULL) {
        printf("%d -> ", curr->data);
        curr = curr->next;
    }
    printf("NULL\n");

    return 0;
}
```

Output:

```
10 -> 20 -> 30 -> NULL



=== Code Execution Successful ===
```

Insertion at end:
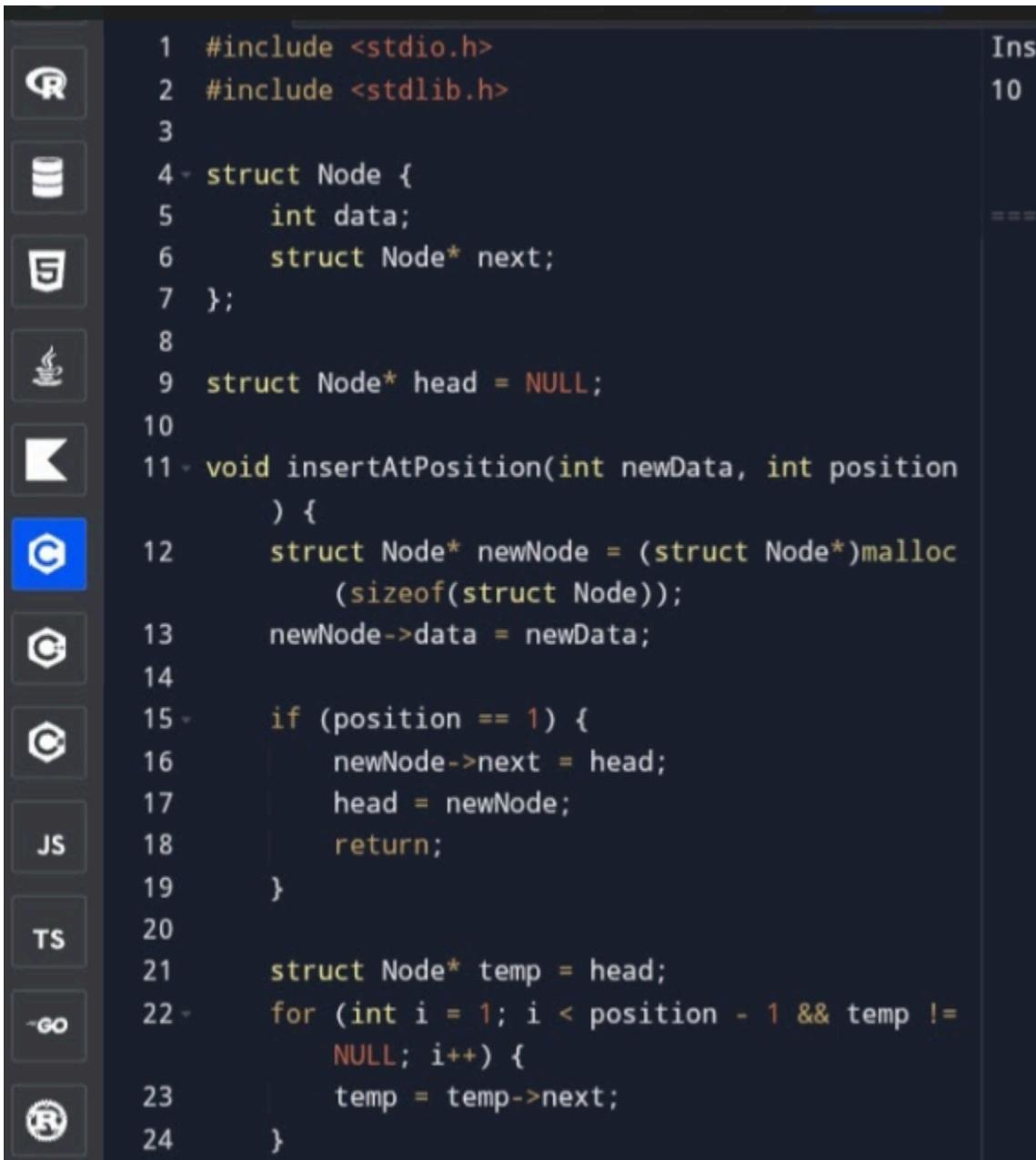
Input:

```c
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   struct Node {
5       int data;
6       struct Node* next;
7   };
8
9   struct Node* head = NULL;
10
11  void insertAtEnd(int newData) {
12      struct Node* newNode = (struct Node*)malloc
            (sizeof(struct Node));
13      newNode->data = newData;
14      newNode->next = NULL;
15
16      if (head == NULL) {
17          head = newNode;
18          return;
19      }
20
21      struct Node* temp = head;
22      while (temp->next != NULL) {
23          temp = temp->next;
24      }
25      temp->next = newNode;
26  }
27
28  int main() {
29      insertAtEnd(10);
30      insertAtEnd(20);
31      insertAtEnd(30);
32
33      struct Node* curr = head;
34      while (curr != NULL) {
35          printf("%d -> ", curr->data);
36          curr = curr->next;
37      }
38      printf("NULL\n");
39
40      return 0;
41  }
```

Output:

```
10 -> 20 -> 30 -> NULL
```

Insertion at position

Input:

```c
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   struct Node {
5       int data;
6       struct Node* next;
7   };
8
9   struct Node* head = NULL;
10
11  void insertAtPosition(int newData, int position
        ) {
12      struct Node* newNode = (struct Node*)malloc
            (sizeof(struct Node));
13      newNode->data = newData;
14
15      if (position == 1) {
16          newNode->next = head;
17          head = newNode;
18          return;
19      }
20
21      struct Node* temp = head;
22      for (int i = 1; i < position - 1 && temp !=
            NULL; i++) {
23          temp = temp->next;
24      }
```

Ins
10

===

```
24          }
25
26 ▾        if (temp == NULL) {
27              printf("Position out of range\n");
28              free(newNode);
29          } else {
30              newNode->next = temp->next;
31              temp->next = newNode;
32          }
33  }
34
35 ▾ int main() {
36          |
37      insertAtPosition(10, 1);
38      insertAtPosition(30, 2);
39
40
41      printf("Inserting 20 at position 2\n");
42      insertAtPosition(20, 2);
43
44      struct Node* curr = head;
45 ▾    while (curr != NULL) {
46          printf("%d -> ", curr->data);
47          curr = curr->next;
48      }
49      printf("NULL\n");
50
51      return 0;
52  }
```

Output:

```
Run        Output                                                    Clea

           Inserting 20 at position 2
           10 -> 20 -> 30 -> NULL


           === Code Execution Successful ===
```