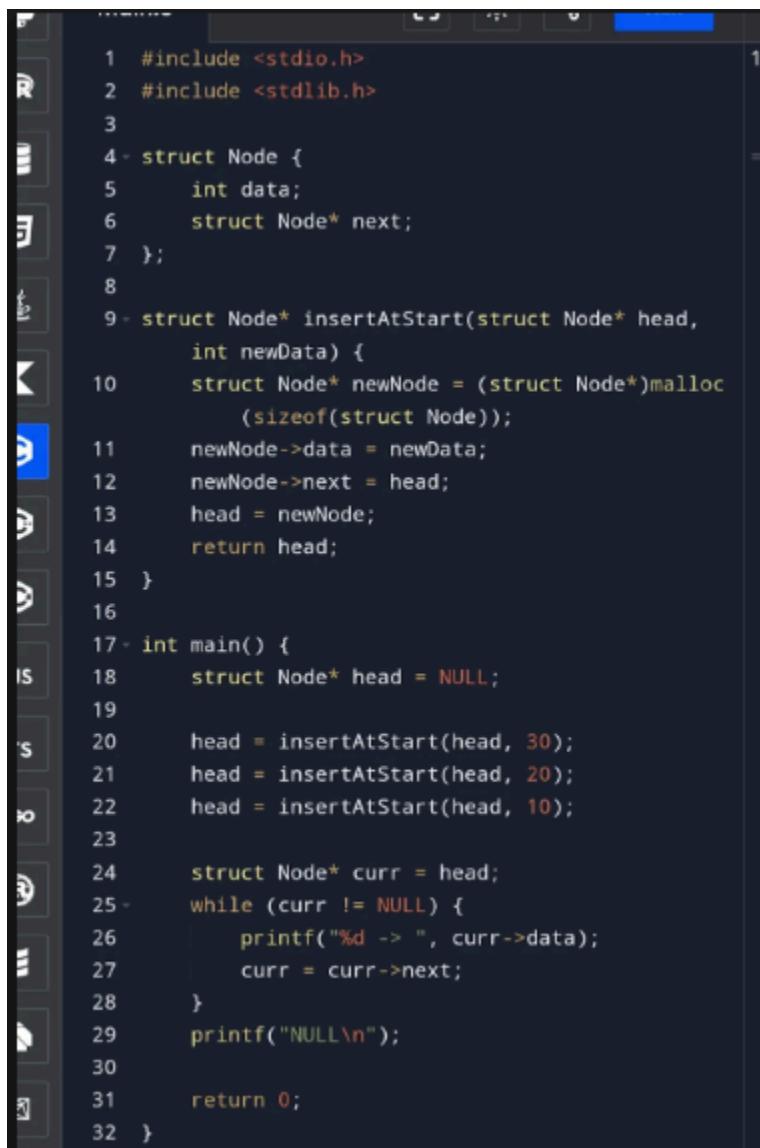


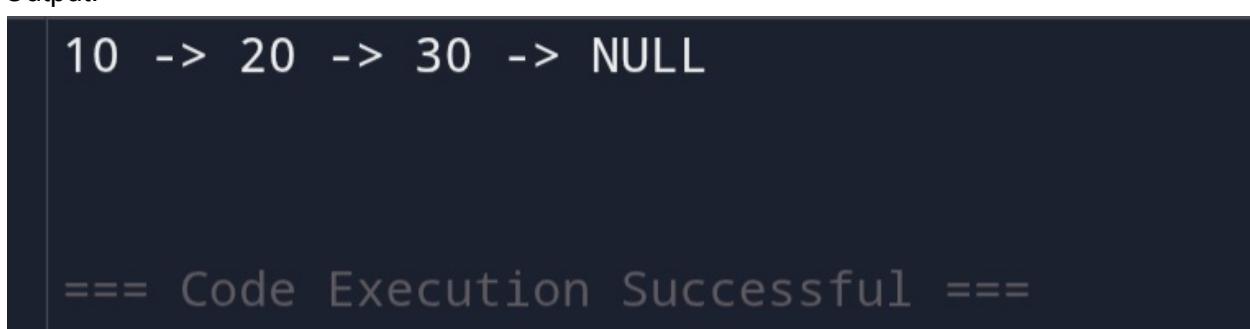
Insertion at start:

Input:



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

Output:



```
10 -> 20 -> 30 -> NULL
=====
==== Code Execution Successful ===
```

Insertion at end:

Input:

```
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
13         (sizeof(struct Node));
14     newNode->data = newData;
15     newNode->next = NULL;
16
17     if (head == NULL) {
18         head = newNode;
19         return;
20     }
21
22     struct Node* temp = head;
23     while (temp->next != NULL) {
24         temp = temp->next;
25     }
26     temp->next = newNode;
27 }
28
29 int main() {
30     insertAtEnd(10);
31     insertAtEnd(20);
32     insertAtEnd(30);
33
34     struct Node* curr = head;
35     while (curr != NULL) {
36         printf("%d -> ", curr->data);
37         curr = curr->next;
38     }
39     printf("NULL\n");
40
41 }
```

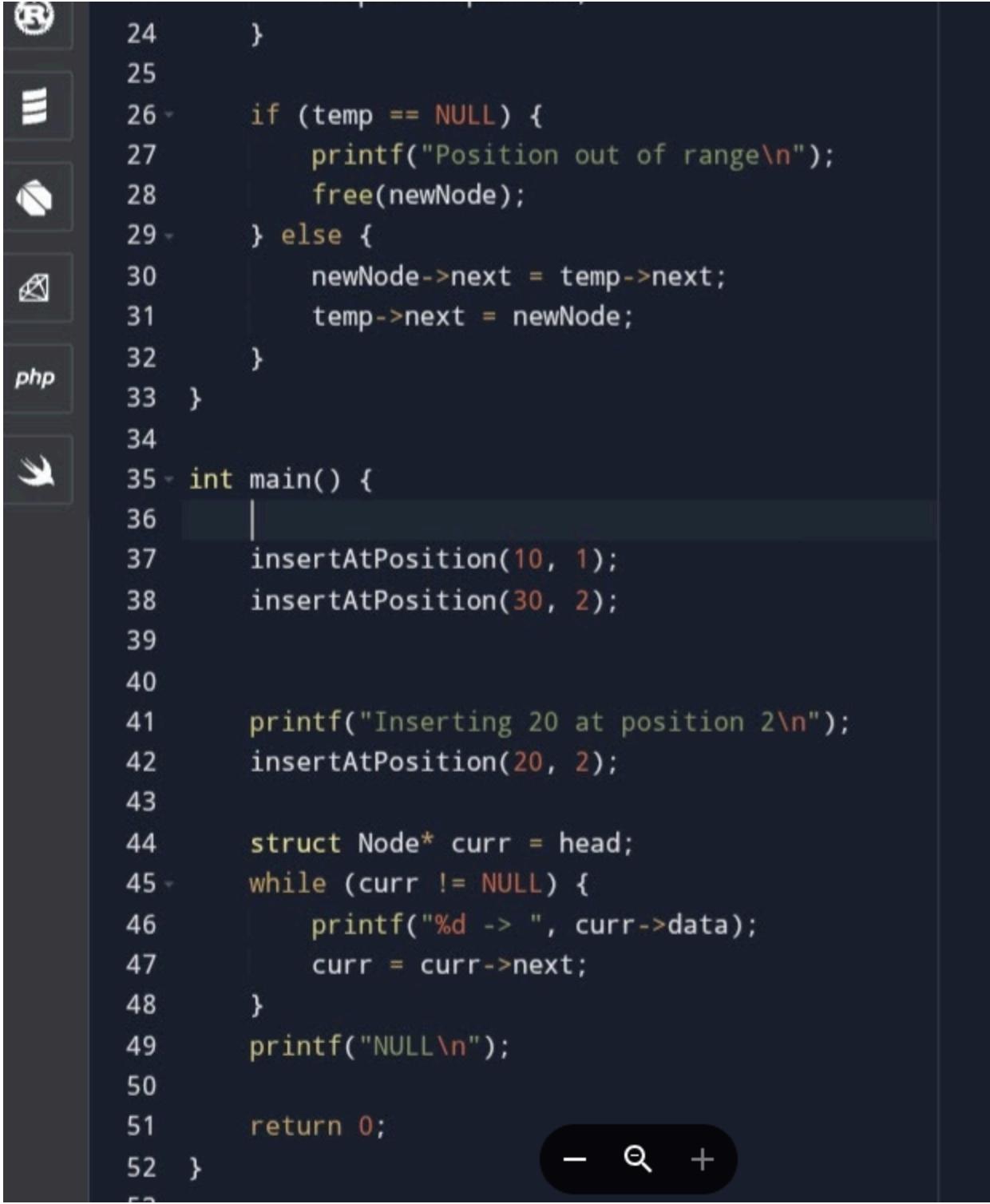
Output:

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

Insertion at position

Input:

```
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 ) {
13     struct Node* newNode = (struct Node*)malloc
14         (sizeof(struct Node));
15     newNode->data = newData;
16
17     if (position == 1) {
18         newNode->next = head;
19         head = newNode;
20         return;
21     }
22
23     struct Node* temp = head;
24     for (int i = 1; i < position - 1 && temp !=
25         NULL; i++) {
26         temp = temp->next;
27     }
```



The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for file operations (New, Open, Save, Find, Replace, Cut, Copy, Paste, Select All, Undo, Redo) and a PHP icon. The main area displays the following C code:

```
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 }
```

A status bar at the bottom right contains icons for zoom in/out, search, and other functions.

Output:

Run	Output	Clear
	<pre>Inserting 20 at position 2 10 -> 20 -> 30 -> NULL ==== Code Execution Successful ====</pre>	