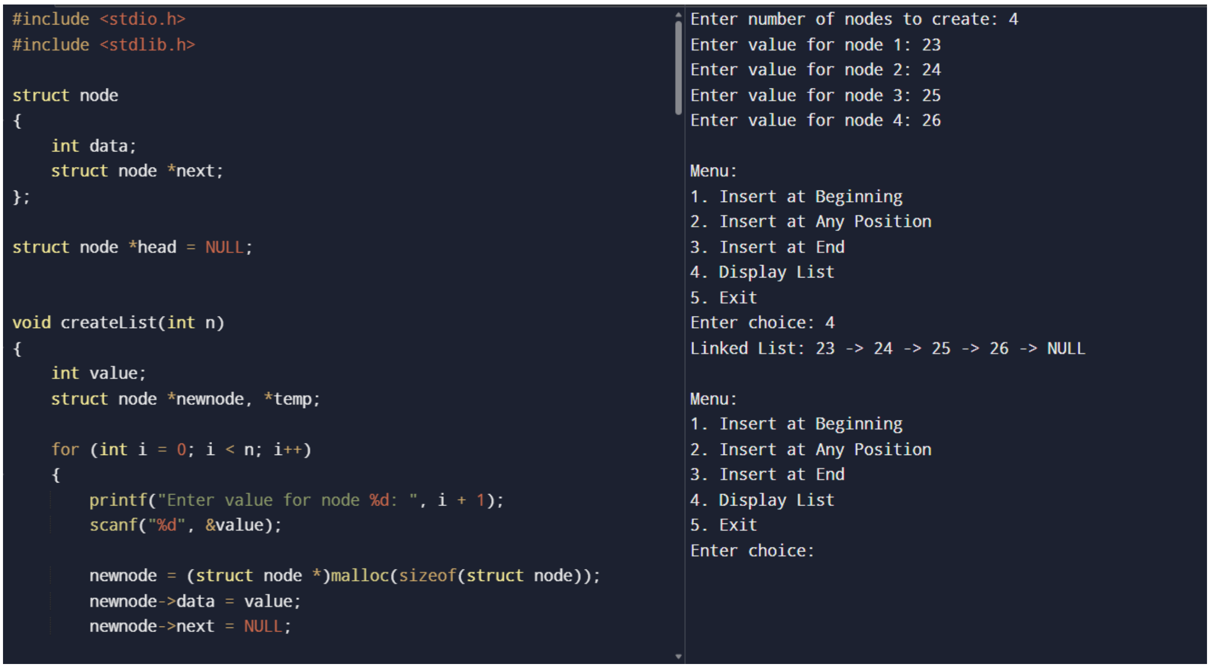
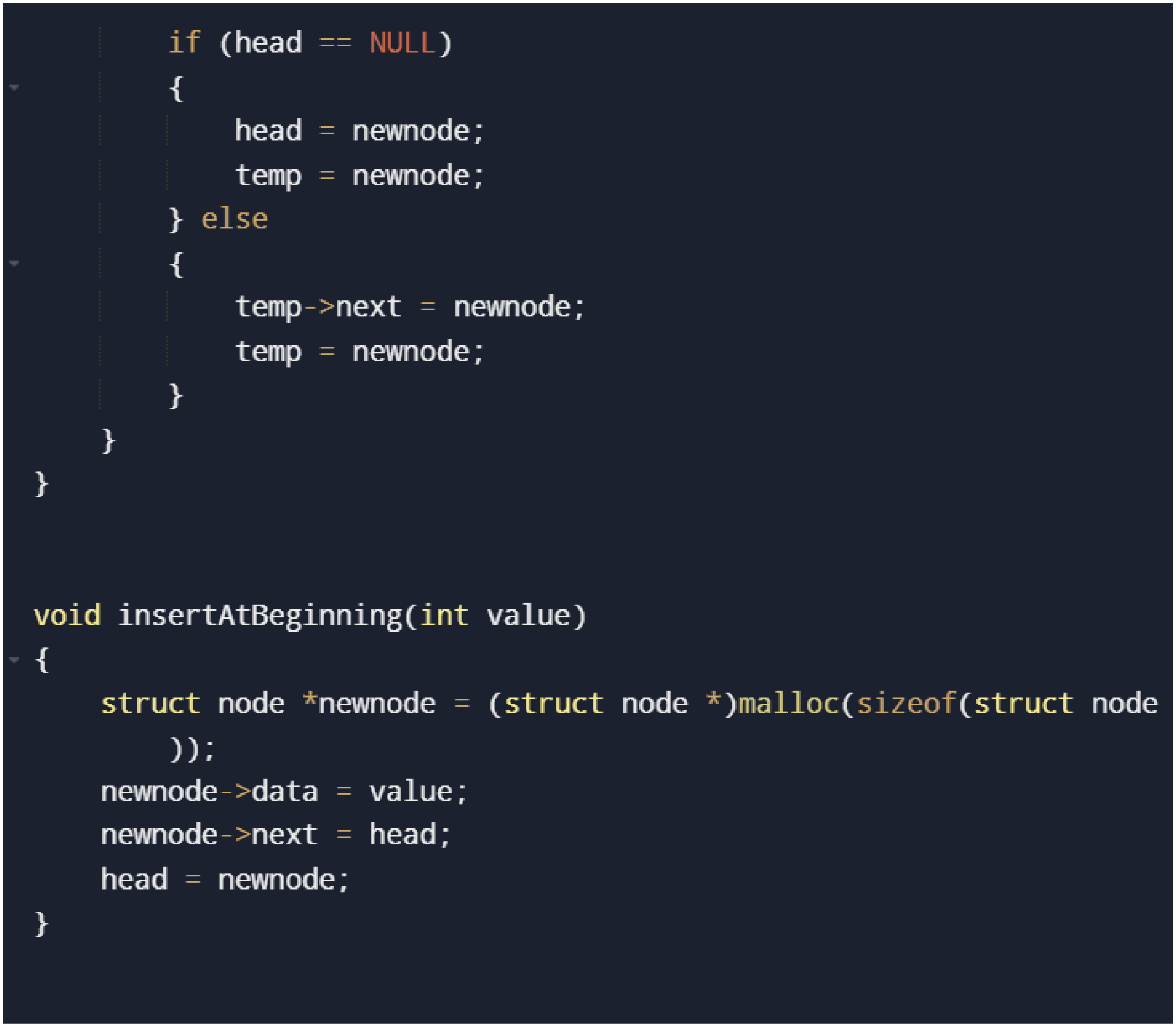
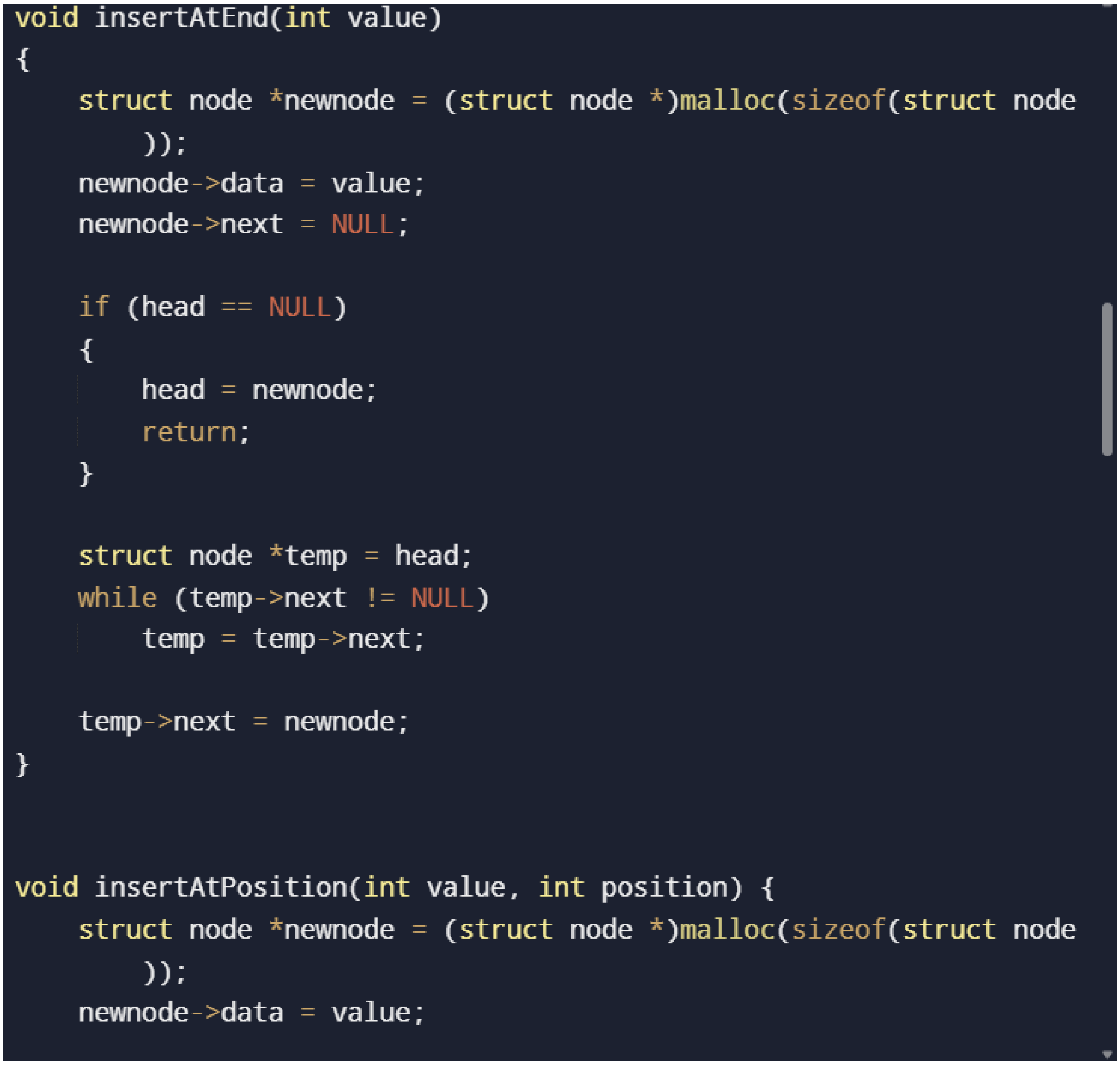


```
#include <stdio.h>
#include <stdlib.h>

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

struct node *head = NULL;


void createList(int n)
{
    int value;
    struct node *newnode, *temp;

    for (int i = 0; i < n; i++)
    {
        printf("Enter value for node %d: ", i + 1);
        scanf("%d", &value);

        newnode = (struct node *)malloc(sizeof(struct node));
        newnode->data = value;
        newnode->next = NULL;
```

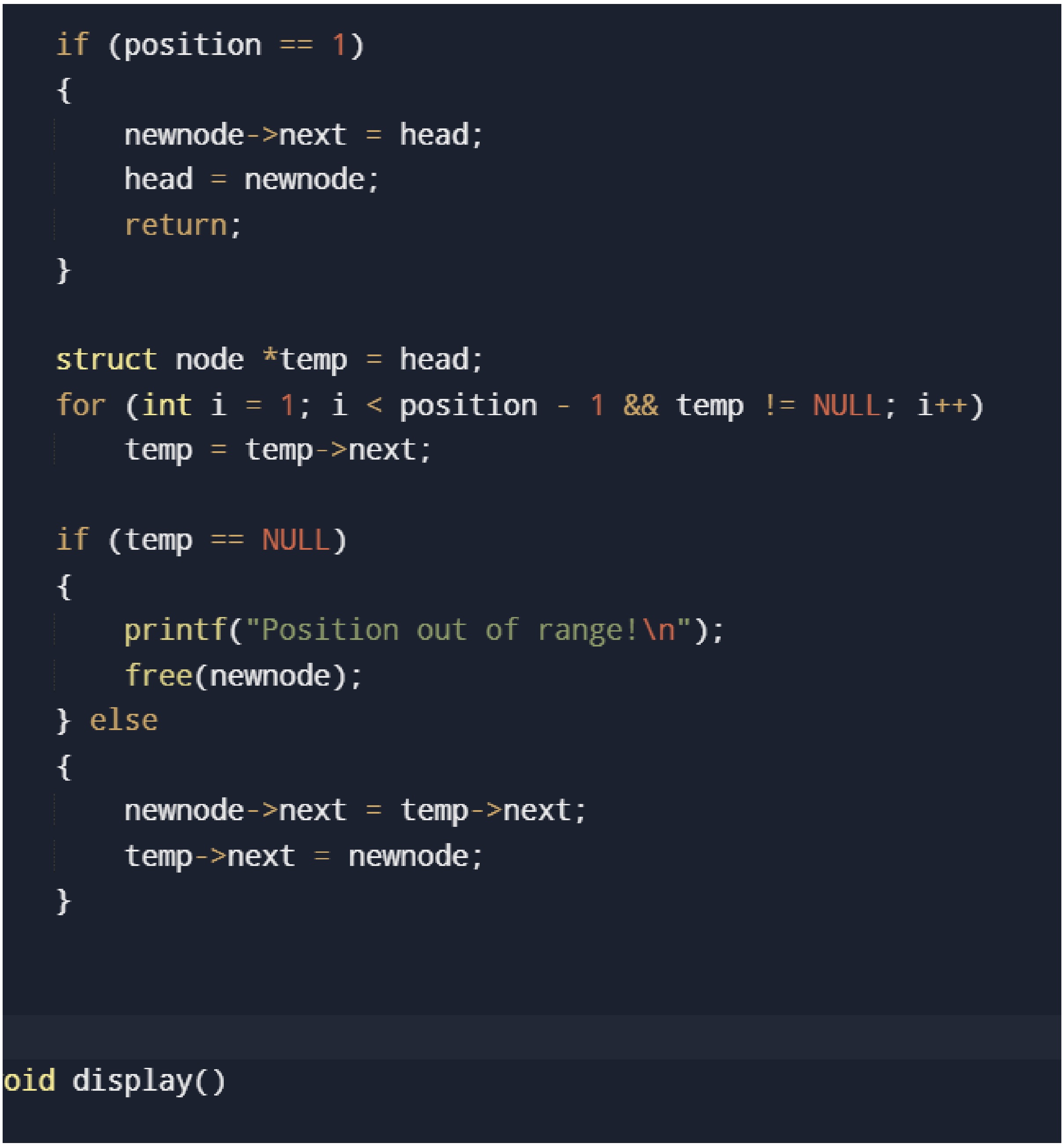
```
Enter number of nodes to create: 4
Enter value for node 1: 23
Enter value for node 2: 24
Enter value for node 3: 25
Enter value for node 4: 26

Menu:
1. Insert at Beginning
2. Insert at Any Position
3. Insert at End
4. Display List
5. Exit
Enter choice: 4
Linked List: 23 -> 24 -> 25 -> 26 -> NULL

Menu:
1. Insert at Beginning
2. Insert at Any Position
3. Insert at End
4. Display List
5. Exit
Enter choice:
```



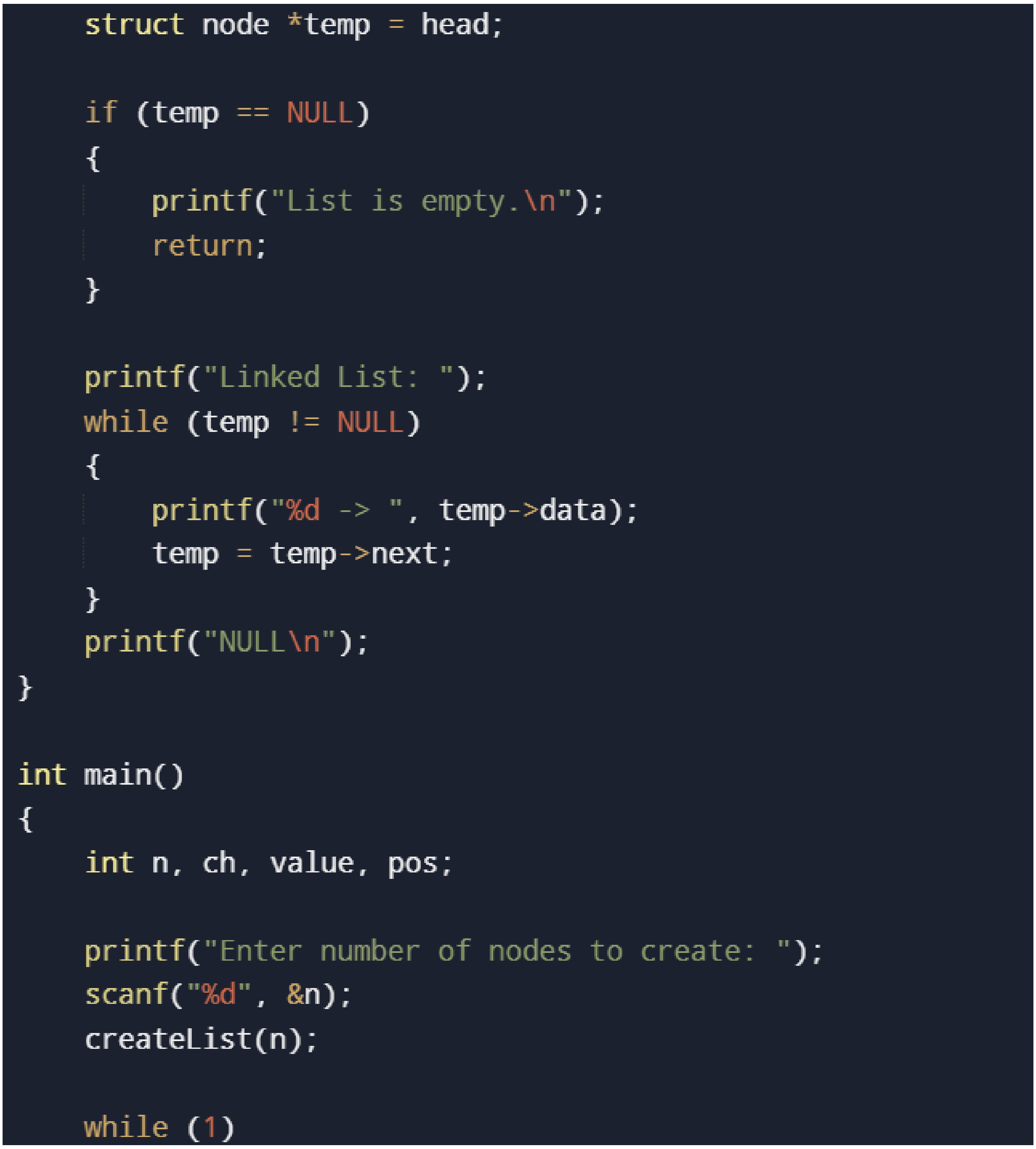
```
        if (head == NULL)
        {
            head = newnode;
            temp = newnode;
        } else
        {
            temp->next = newnode;
            temp = newnode;
        }
    }
}


void insertAtBeginning(int value)
{
    struct node *newnode = (struct node *)malloc(sizeof(struct node
        ));
    newnode->data = value;
    newnode->next = head;
    head = newnode;
}
```



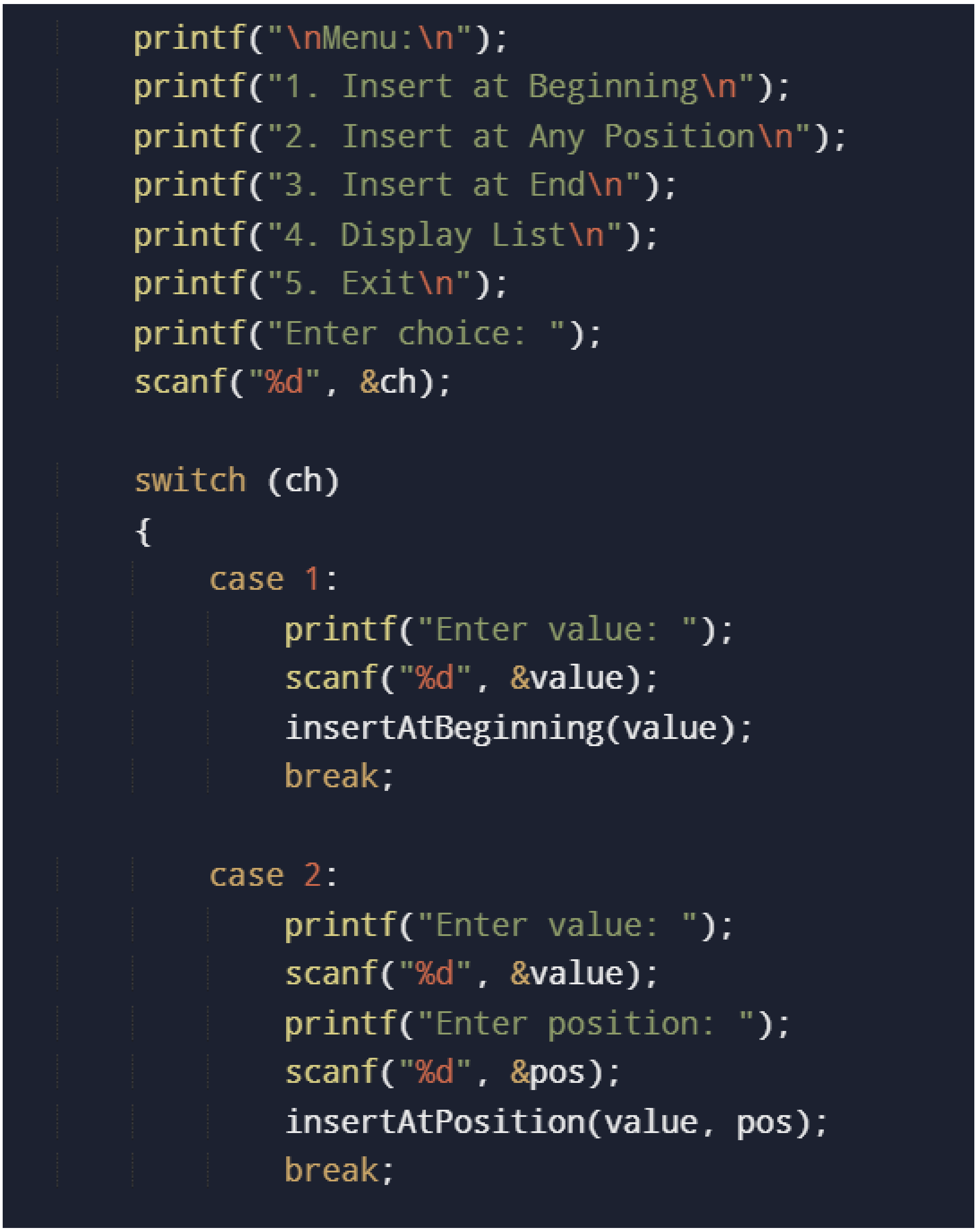
```c
void insertAtEnd(int value)
{
    struct node *newnode = (struct node *)malloc(sizeof(struct node
        ));
    newnode->data = value;
    newnode->next = NULL;

    if (head == NULL)
    {
        head = newnode;
        return;
    }

    struct node *temp = head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = newnode;
}


void insertAtPosition(int value, int position) {
    struct node *newnode = (struct node *)malloc(sizeof(struct node
        ));
    newnode->data = value;
```



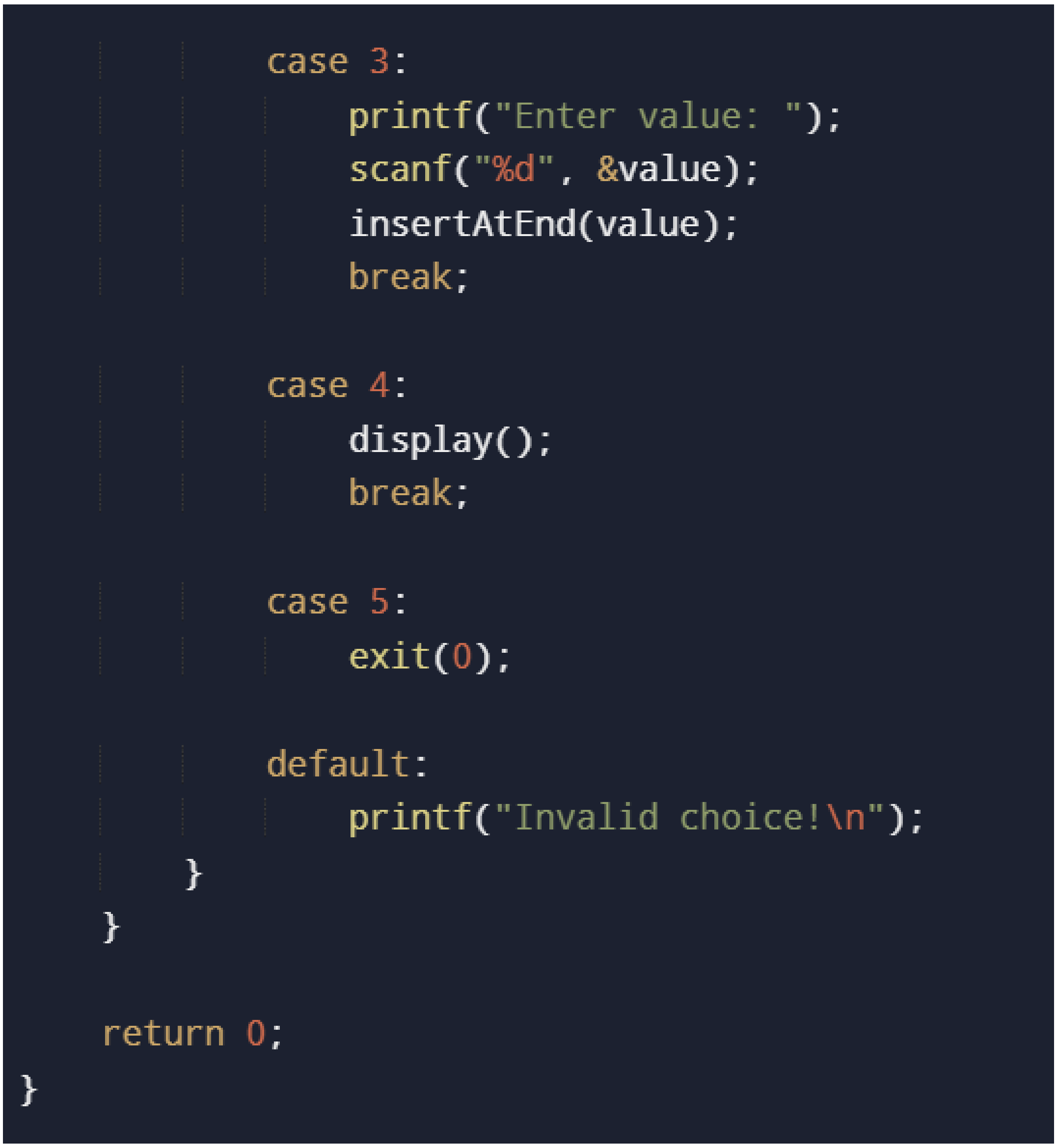
```
   if (position == 1)
   {
       newnode->next = head;
       head = newnode;
       return;
   }

   struct node *temp = head;
   for (int i = 1; i < position - 1 && temp != NULL; i++)
       temp = temp->next;

   if (temp == NULL)
   {
       printf("Position out of range!\n");
       free(newnode);
   } else
   {
       newnode->next = temp->next;
       temp->next = newnode;
   }



oid display()
```



```c
    struct node *temp = head;

    if (temp == NULL)
    {
        printf("List is empty.\n");
        return;
    }

    printf("Linked List: ");
    while (temp != NULL)
    {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

int main()
{
    int n, ch, value, pos;

    printf("Enter number of nodes to create: ");
    scanf("%d", &n);
    createList(n);

    while (1)
```



```c
        printf("\nMenu:\n");
        printf("1. Insert at Beginning\n");
        printf("2. Insert at Any Position\n");
        printf("3. Insert at End\n");
        printf("4. Display List\n");
        printf("5. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &ch);

        switch (ch)
        {
            case 1:
                printf("Enter value: ");
                scanf("%d", &value);
                insertAtBeginning(value);
                break;

            case 2:
                printf("Enter value: ");
                scanf("%d", &value);
                printf("Enter position: ");
                scanf("%d", &pos);
                insertAtPosition(value, pos);
                break;
```



```c
            case 3:
                printf("Enter value: ");
                scanf("%d", &value);
                insertAtEnd(value);
                break;

            case 4:
                display();
                break;

            case 5:
                exit(0);

            default:
                printf("Invalid choice!\n");
        }
    }

    return 0;
}
```