

```
sindylinkedlist.c - CodeBlocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools- Plugins DebugBlocks Settings Help

Start here X sindylinkedlist.c X
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
12
13
14 void createList(int n) {
15     struct node *newNode, *temp = NULL;
16     int data, i;
17
18     if (n <= 0) {
19         printf("Number of nodes should be greater than 0\n");
20         return;
21     }
22
23     if (head != NULL) {
24         printf("Warning: Overwriting existing list.\n");
25         head = NULL;
26     }
27
28     for (i = 1; i <= n; i++) {
29         newNode = (struct node*)malloc(sizeof(struct node));
30         if (newNode == NULL) {
31             printf("Memory allocation failed.\n");
32             return;
33         }
34         printf("Enter data for node %d: ", i);
35         scanf("%d", &data);
36         newNode->data = data;
37         newNode->next = NULL;
38         if (head == NULL)
39             head = newNode;
40         else
41             temp->next = newNode;
42         temp = newNode;
43     }
44     printf("\nLinked list created successfully.\n");
45 }
46
47
48
49
50
51
52 void insertAtBeginning(int data) {
53     struct node *newNode = (struct node*)malloc(sizeof(struct node));
54     if (newNode == NULL) {
55         printf("Memory allocation failed.\n");
56         return;
57     }
58     newNode->data = data;
59     newNode->next = head;
60     head = newNode;
61     printf("Node inserted at the beginning.\n");
62 }
63
64
65 void insertAtEnd(int data) {
66     struct node *newNode = (struct node*)malloc(sizeof(struct node));
67     if (newNode == NULL) {
68         printf("Memory allocation failed.\n");
69         return;
70     }
71     newNode->data = data;
72     newNode->next = NULL;
73     if (head == NULL) {
74         head = newNode;
75     }
76     else {
77         struct node *temp = head;
78         while (temp->next != NULL)
79             temp = temp->next;
80         temp->next = newNode;
81     }
82     printf("Node inserted at the end.\n");
83 }
84
85
86 void insertAtPosition(int data, int pos) {
87     int i;
88     struct node *newNode, *temp = head;
89
90     if (pos < 1) {
91         printf("Invalid position. Position must be 1 or greater.\n");
92         return;
93     }
94     if (pos == 1) {
95         insertAtBeginning(data);
96         return;
97     }
98     for (i = 1; i < pos - 1 && temp != NULL; i++)
99         temp = temp->next;
100
101
102
103
104
105
```

```
D:\chehan\DP\sindylinkedlist.c C/C++ Windows (CR-LF) WINDOWS-1252 Line 27, Col 6, Pos 445 Insert Read/Write default 09:44 10-11-2023
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools- Plugins DebugBlocks Settings Help

Start here X sindylinkedlist.c X
52 void insertAtBeginning(int data) {
53     struct node *newNode = (struct node*)malloc(sizeof(struct node));
54     if (newNode == NULL) {
55         printf("Memory allocation failed.\n");
56         return;
57     }
58     newNode->data = data;
59     newNode->next = head;
60     head = newNode;
61     printf("Node inserted at the beginning.\n");
62 }
63
64 void insertAtEnd(int data) {
65     struct node *newNode = (struct node*)malloc(sizeof(struct node));
66     if (newNode == NULL) {
67         printf("Memory allocation failed.\n");
68         return;
69     }
70     newNode->data = data;
71     newNode->next = NULL;
72     if (head == NULL) {
73         head = newNode;
74     }
75     else {
76         struct node *temp = head;
77         while (temp->next != NULL)
78             temp = temp->next;
79         temp->next = newNode;
80     }
81     printf("Node inserted at the end.\n");
82 }
83
84
85 void insertAtPosition(int data, int pos) {
86     int i;
87     struct node *newNode, *temp = head;
88
89     if (pos < 1) {
90         printf("Invalid position. Position must be 1 or greater.\n");
91         return;
92     }
93     if (pos == 1) {
94         insertAtBeginning(data);
95         return;
96     }
97     for (i = 1; i < pos - 1 && temp != NULL; i++)
98         temp = temp->next;
99
100
101
102
103
104
105
```

```

singlylinkedlist.c - CodeBlocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools> Plugins DevBlocks Settings Help
Start here X singlylinkedlist.c X
103     temp = temp->next;
104
105
106     if (temp == NULL) {
107         printf("Position out of range: List is not long enough to reach position %d.\n", pos);
108         return;
109     }
110
111
112     newNode = (struct node*)malloc(sizeof(struct node));
113     if (newNode == NULL) {
114         printf("Memory allocation failed.\n");
115         return;
116     }
117     newNode->data = data;
118
119     newNode->next = temp->next;
120
121     temp->next = newNode;
122     printf("Node inserted at position %d\n", pos);
123 }
124
125
126 void displayList() {
127     struct node *temp = head;
128
129     if (head == NULL) {
130         printf("List is empty\n");
131         return;
132     }
133
134     printf("\nLinked list: ");
135     while (temp != NULL) {
136         printf("%d > ", temp->data);
137         temp = temp->next;
138     }
139     printf("\n");
140 }
141
142
143 int main() {
144     int choice, n, data, pos;
145
146     while (1) {
147         printf("\n---- Singly Linked List Operations ----\n");
148         printf("1. Create linked list\n");
149         printf("2. Insert at Beginning\n");
150         printf("3. Insert at any Position\n");
151         printf("4. Insert at End\n");
152         printf("5. Display list\n");
153         printf("6. Exit\n");
154         printf("Enter your choice: ");
155         scanf("%d", &choice);
156
157         if (choice != 6) {
158             while (getchar() != '\n')
159                 continue;
160             printf("Invalid input. Please enter a number.\n");
161             continue;
162         }
163
164         switch (choice) {
165             case 1:
166                 printf("Enter number of nodes: ");
167                 scanf("%d", &n);
168                 createList(n);
169                 break;
170             case 2:
171                 printf("Enter data to insert: ");
172                 scanf("%d", &data);
173                 insertAtBeginning(data);
174                 break;
175             case 3:
176                 printf("Enter data: ");
177                 scanf("%d", &data);
178                 printf("Enter position: ");
179                 scanf("%d", &pos);
180                 insertAtPosition(data, pos);
181                 break;
182             case 4:
183                 printf("Enter data to insert: ");
184                 scanf("%d", &data);
185                 insertAtEnd(data);
186                 break;
187             case 5:
188                 displayList();
189                 break;
190             case 6:
191                 printf("Exiting...\n");
192                 exit(0);
193             default:
194                 printf("Invalid choice. Try again.\n");
195         }
196     }
197     return 0;
198 }

```

```

singlylinkedlist.c - CodeBlocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools> Plugins DevBlocks Settings Help
Start here X singlydelete.c X singlylinkedlist.c X
148     while (1) {
149         printf("\n---- Singly Linked List Operations ----\n");
150         printf("1. Create linked list\n");
151         printf("2. Insert at Beginning\n");
152         printf("3. Insert at any Position\n");
153         printf("4. Insert at End\n");
154         printf("5. Display list\n");
155         printf("6. Exit\n");
156         printf("Enter your choice: ");
157         scanf("%d", &choice);
158
159         if (choice != 6) {
160             while (getchar() != '\n')
161                 continue;
162             printf("Invalid input. Please enter a number.\n");
163             continue;
164         }
165
166         switch (choice) {
167             case 1:
168                 printf("Enter number of nodes: ");
169                 scanf("%d", &n);
170                 createList(n);
171                 break;
172             case 2:
173                 printf("Enter data to insert: ");
174                 scanf("%d", &data);
175                 insertAtBeginning(data);
176                 break;
177             case 3:
178                 printf("Enter data: ");
179                 scanf("%d", &data);
180                 printf("Enter position: ");
181                 scanf("%d", &pos);
182                 insertAtPosition(data, pos);
183                 break;
184             case 4:
185                 printf("Enter data to insert: ");
186                 scanf("%d", &data);
187                 insertAtEnd(data);
188                 break;
189             case 5:
190                 displayList();
191                 break;
192             case 6:
193                 printf("Exiting...\n");
194                 exit(0);
195             default:
196                 printf("Invalid choice. Try again.\n");
197         }
198     }
199     return 0;
200 }

```

```

D:\chethan\OIP\lindy\lindy.exe
---- Singly Linked list Operations ----
1. Create linked list
2. Insert at Beginning
3. Insert at any Position
4. Insert at End
5. Display list
6. Exit
Enter your choice: 1
Enter number of nodes: 2
Enter data for node 1: 3
Enter data for node 2: 4
linked list created successfully

---- Singly Linked list Operations ----
1. Create linked list
2. Insert at Beginning
3. Insert at any Position
4. Insert at End
5. Display list
6. Exit
Enter your choice: 5
linked list: 3 -> 4 -> NULL

---- Singly Linked list Operations ----
1. Create linked list
2. Insert at Beginning
3. Insert at any Position
4. Insert at End
5. Display list
6. Exit
Enter your choice: 2
Enter data to insert: 344
Node inserted at the beginning

---- Singly Linked list Operations ----
1. Create linked list
2. Insert at Beginning
3. Insert at any Position
4. Insert at End
5. Display list
6. Exit
Enter your choice: 5
linked list: 344 -> 3 -> 4 -> NULL

---- Singly Linked list Operations ----
1. Create linked list
2. Insert at Beginning
3. Insert at any Position
4. Insert at End
5. Display list
6. Exit
Enter your choice: 4
Enter data to insert: 878
Node inserted at the end

---- Singly Linked list Operations ----
1. Create linked list
2. Insert at Beginning
3. Insert at any Position

```

chethan558/DSC | LeetCode - The World's Le... | Favorite - LeetCode | Progress - LeetCode | Middle of the Linked List - L... | LeetCode middle node solu... | +

leetcode.com/problems/middle-of-the-linked-list/

class 3 - Google Drive | online master ca no... | YouTube | Maps | Gmail

Problem List | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 36 / 36 testcases passed
chethanmsce25 submitted at Dec 08, 2025 21:07

Runtime: 0 ms | Beats 100.00% | Memory: 8.69 MB | Beats 12.13%

Analyze Complexity

Code | C

```
1 struct ListNode* middleNode(struct ListNode* head) {
2     struct ListNode* temp = head;
3
4     // Fast & Slow pointer method
5     while (temp != NULL && temp->next != NULL) {
6         head = head->next; // slow moves 1 step
7     }
8     return head;
9 }
10
11
12
```

Saved | Ln 12, Col 1

Testcase | Test Result

22°C Partly cloudy | 21:08 | 08-12-2025

- 11) Write a program to implement ~~linked~~ singly linked list with the following
- create a linked list
 - Insertion at a node at first position * any position * end of the list
 - Display the contents of linked list

pseudocode

Structure node

data

next \rightarrow node

end structure

\Rightarrow empty linked list = head \leftarrow null

\Rightarrow New node create

create node (value)

newnode \leftarrow allocate memory for node

newnode.data \leftarrow value

newnode.next \leftarrow null

return node

end function

\Rightarrow Insert at first position

firstposition (value)

newnode \leftarrow create node (value)

newnode.next \leftarrow head

head \leftarrow newnode

end function

\Rightarrow Insert at end

insertatend (value)

newnode \rightarrow create node (value)

if head = null then

head \leftarrow newnode

return



Shot on OnePlus

chE_Reddy

\leftarrow head

```
while temp.next ≠ null  
temp ← temp.next  
end while  
temp.next ← newnode  
end function
```

```
⇒ Insert at any position:  
Insertatanyposition (targetvalue, newvalue)  
temp ← head  
while temp ≠ null and temp.data ≠ targetvalue  
temp ← temp.next  
end while  
if temp = null then  
print "target not found"  
return  
endif  
newnode ← create node (newvalue)  
newnode.next ← temp.next  
temp.next ← newnode  
end
```

```
⇒ Display  
Display ()  
temp ← head  
while temp ≠ null  
print temp.data  
temp ← temp.next  
end while  
end function
```




```

#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node* next;
};
struct node* head = NULL;
void createlist (int n) {
    struct node* newnode, * temp = NULL;
    int data, i;
    if (n <= 0) {
        printf("Number of nodes should be greater than 0\n");
        return;
    }
    if (head != NULL) {
        printf("Warning: Overwriting existing list.\n");
        head = NULL;
    }
    for (i=1; i<=n; i++) {
        newnode = (struct node*) malloc (sizeof (struct node));
        if (newnode == NULL) {
            printf("Memory allocation failed.\n");
            return;
        }
        printf("Enter data for node %d: ", i);
        scanf ("%d", &data);
        newnode->data = data;
        newnode->next = NULL;
        if (head == NULL)
            head = newnode;
        else
            temp->next = newnode;
        temp = newnode;
    }
}

```



Shot on OnePlus
chE_Reddy

DATE: _____

```

    printf("\n linked list created \n");
}

void insertatBeginning (int data) {
    struct node * newnode = (struct node *) malloc (sizeof (struct node));
    if (newnode == NULL) {
        printf("Memory allocation failed \n");
        return;
    }
    newnode->data = data;
    newnode->next = head;
    head = newnode;
    printf("node inserted at the beginning \n");
}

```

```

void insertatend (int data) {
    struct node * newnode = (struct node *) malloc (sizeof (struct node));
    if (newnode == NULL) {
        printf("Memory allocation failed \n");
        return;
    }
    newnode->data = data;
    newnode->next = NULL;
    if (head == NULL) {
        head = newnode;
    } else {
        struct node * temp = head;
        while (temp->next != NULL)
            temp = temp->next;
        temp->next = newnode;
    }
}

```



printf("\n node inserted at the end \n");


```

void insertAtPosition(int data, int pos) {
    int i;
    struct node * newnode, * temp = head;
    if (pos <= 1) {
        printf("Invalid position\n");
        return;
    }
    if (pos == 1) {
        insertAtBeginning(data);
        return;
    }
    for (i = 1; i < pos - 1; i++) {
        temp = temp->next;
    }
    if (temp == NULL) {
        printf("position out of range\n", pos);
        return;
    }
    newnode = (struct node *) malloc (sizeof (struct node));
    if (newnode == NULL) {
        printf("Memory allocation failed\n");
        return;
    }
    newnode->data = data;
    newnode->next = temp->next;
    temp->next = newnode;
    printf("Node inserted at position\n", pos);
}

```

```

void displayList() {
    struct node * temp = head;
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }
}

```




```

printf("In linked list");
while (temp != NULL) {
    printf("%d->", temp->data);
    temp = temp->next;
}
printf("NULL\n");
}

```

```

int main() {
    int choice, n, data;
    while (1) {
        printf("\n 1. create linked list\n");
        printf("\n 2. Insert at Beginning\n");
        printf("\n 3. Insert at any position\n");
        printf("\n 4. Insert at End\n");
        printf("\n 5. Display list\n");
        printf("\n 6. Exit\n");
        if (scanf("%d", &choice) != 1) {
            while (getchar() != '\n');
            printf("Invalid input. please enter a number\n");
            continue;
        }
    }
}

```

```

switch (choice) {

```

```

    case 1:

```

```

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

```

```

    case 2:

```

```

        printf("Enter data to insert");
        scanf("%d", &data);
        insertAtBeginning(data);
        break;

```



Shot on OnePlus
chE_Reddy

CASE 3:-

```
printf("Enter data:");  
scanf("%d", &data);  
printf("Enter position:");  
scanf("%d", &pos);  
insertatpositen(data, pos);  
break;
```

CASE 4:-

```
printf("Enter data to insert:");  
scanf("%d", &data);  
insertatend(data);  
break;
```

CASE 5:-

```
displaylist();  
break;
```

CASE 6:-

```
printf("Exiting");  
exit(0);
```

}

}

Output:-

singly linked list operation.

1. create linked list
2. Insert at Beginning
3. Insert at any position
4. Display a list
5. Insert at End
6. Exit.

Enter your choice:-

Enter number of nodes:- 3

Enter data for node 1:- 10

Enter data for node 2:- 20



Enter data for node 3: 30

Enter your choice is

10 20 30

Enter your choice is

Enter your data is

Enter position is

Node inserted at position 2

Enter your choice is

Enter data to insert is

Node inserted at the end

~~12/1/25~~
Seen

