

ASSIGNMENT



TASK NO: _____ 8

SUBJECT: _____ data structure and algorithm

SUBMITTED TO: _____ madam hijab durrani

SUBMITTED BY: _____ misbah ullah jan

CLASS CODE: _____ bsse-2024A

SEMESTER: _____ 3rd

DATE: _____ 12/22/25

DEPARTMENT OF: _____ software engineering



CECOS UNIVERSITY HAYATABAD PESHAWAR

LAB TASK 1:

1) Algorithm (Step by Step)

Algorithm: INSERT_AT_END(head, value)

1. New node N banao
2. N.data = value set karo
3. N.next = NULL set karo
4. Agar head == NULL ho
 - o head = N
 - o Stop
5. Warna temp = head rakho
6. Jab tak temp.next != NULL ho
 - o temp = temp.next
7. temp.next = N
8. Stop

Algorithm: DISPLAY_LIST(head)

1. temp = head rakho
2. Jab tak temp != NULL ho
 - o temp.data print karo
 - o temp = temp.next
3. Stop

The screenshot shows a code editor window with the following details:

- File:** main.cpp
- Toolbar:** Includes icons for copy, paste, share, and run.
- Sidebar:** A vertical sidebar on the left contains icons for various languages: Python, R, SQL, HTML, CSS, Java, C, C++, JavaScript, TypeScript, Go, and R.
- Code Content:**

```
1 #include <iostream>
2 using namespace std;
3
4 // Node structure
5 struct Node {
6     int data;
7     Node* next;
8 };
9
10 Node* head = NULL;
11
12 // Insert at end
13 void insertAtEnd(int value) {
14     Node* newNode = new Node();
15     newNode->data = value;
16     newNode->next = NULL;
17
18     // If list is empty
19     if (head == NULL) {
20         head = newNode;
21         return;
22     }
23
24     Node* temp = head;
25     while (temp->next != NULL) {
26         temp = temp->next;
27     }
28     temp->next = newNode;
```

```
29  }
30
31 // Display list
32 void displayList() {
33     Node* temp = head;
34     while (temp != NULL) {
35         cout << temp->data << " -> ";
36         temp = temp->next;
37     }
38     cout << "NULL" << endl;
39 }
40
41 int main() {
42     insertAtEnd(10);
43     insertAtEnd(20);
44     insertAtEnd(30);
45
46     displayList();
47
48     return 0;
49 }
```

www.programiz.com/html/online-compiler/

OUTPUT:

The screenshot shows the output of a C++ program. The output window has a header 'Output' and a 'Clear' button. The text in the window reads:
10 -> 20 -> 30 -> NULL
== Code Execution Successful ==

LAB TASK 2:

1) Algorithm

Algorithm: INSERT_AT_BEGINNING(head, value)

1. New node N banao
2. N.data = value set karo
3. N.prev = NULL set karo
4. N.next = head set karo
5. Agar head != NULL ho
 - o head.prev = N
6. head = N update karo
7. Stop

Algorithm: TRAVERSE_FORWARD(head)

1. temp = head rakho
2. Jab tak temp != NULL ho
 - o temp.data print karo
 - o temp = temp.next
3. Stop

Algorithm: TRAVERSE_BACKWARD(head)

1. temp = head rakho
2. Jab tak temp.next != NULL ho
 - o temp = temp.next
3. Ab jab tak temp != NULL ho
 - o temp.data print karo
 - o temp = temp.prev
4. Stop

```
main.cpp

1 #include <iostream>
2 using namespace std;
3
4 // Node structure for Doubly Linked List
5 struct Node {
6     int data;
7     Node* prev;
8     Node* next;
9 };
10
11 Node* head = NULL;
12
13 // Insert at beginning
14 void insertAtBeginning(int value) {
15     Node* newNode = new Node();
16     newNode->data = value;
17     newNode->prev = NULL;
18     newNode->next = head;
19
20     if (head != NULL) {
21         head->prev = newNode;
22     }
23     head = newNode;
24 }
25
26 // Traverse forward
27 void traverseForward() {
28     Node* temp = head;
```

```
29     cout << "Forward Traversal: ";
30     while (temp != NULL) {
31         cout << temp->data << " -> ";
32         temp = temp->next;
33     }
34     cout << "NULL" << endl;
35 }
36
37 // Traverse backward
38 void traverseBackward() {
39     Node* temp = head;
40
41     // Move to last node
42     while (temp != NULL && temp->next != NULL) {
43         temp = temp->next;
44     }
45
46     cout << "Backward Traversal: ";
47     while (temp != NULL) {
48         cout << temp->data << " -> ";
49         temp = temp->prev;
50     }
51     cout << "NULL" << endl;
52 }
53
```

```
54 int main() {
55     insertAtBeginning(10);
56     insertAtBeginning(20);
57     insertAtBeginning(30);
58
59     traverseForward();
60     traverseBackward();
61
62     return 0;
63 }
64
```

OUTPUT:

Output**Clear**

```
Forward Traversal: 30 <-> 20 <-> 10 <-> NULL  
Backward Traversal: 10 <-> 20 <-> 30 <-> NULL
```

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

LAB TASK 3:

1) Algorithm

Algorithm: INSERT_NODE_CIRCULAR(head, value)

1. New node N banao
2. N.data = value set karo
3. Agar head == NULL ho
 - o head = N
 - o N.next = head
 - o Stop
4. Warna temp = head rakho
5. Jab tak temp.next != head ho
 - o temp = temp.next
6. temp.next = N set karo
7. N.next = head set karo
8. Stop

main.cpp

Share

Run

```
1 #include <iostream>
2 using namespace std;
3
4 // Node structure for Circular Linked List
5 struct Node {
6     int data;
7     Node* next;
8 };
9
10 Node* head = NULL;
11
12 // Insert node at end in Circular Linked List
13 void insertNode(int value) {
14     Node* newNode = new Node();
15     newNode->data = value;
16
17     // If list is empty
18     if (head == NULL) {
19         head = newNode;
20         newNode->next = head;
21         return;
22     }
23
24     Node* temp = head;
25     while (temp->next != head) {
26         temp = temp->next;
27     }
28 }
```

```
29     temp->next = newNode;
30     newNode->next = head;
31 }
32
33 // Display Circular Linked List
34 void display() {
35     if (head == NULL) {
36         cout << "List is empty";
37         return;
38     }
39
40     Node* temp = head;
41     do {
42         cout << temp->data << " -> ";
43         temp = temp->next;
44     } while (temp != head);
45
46     cout << "(Back to Head)" << endl;
47 }
48
49 int main() {
50     insertNode(10);
51     insertNode(20);
52     insertNode(30);
53
54     display();
55
56     return 0;
57 }
58
```

OUTPUT:

Output**Clear**

```
10 -> 20 -> 30 -> (Back to Head)
```

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