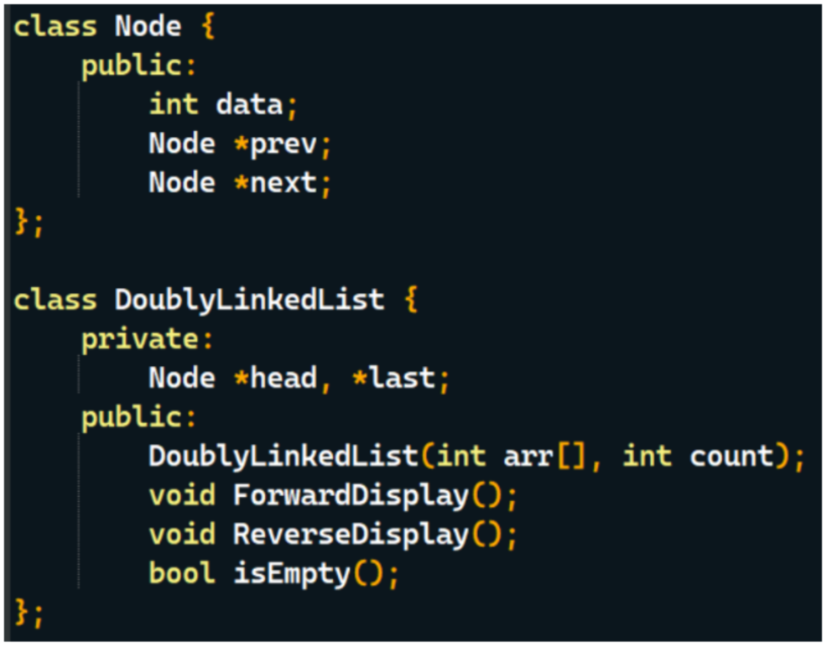
Summary

Sessions (10-02-2023)

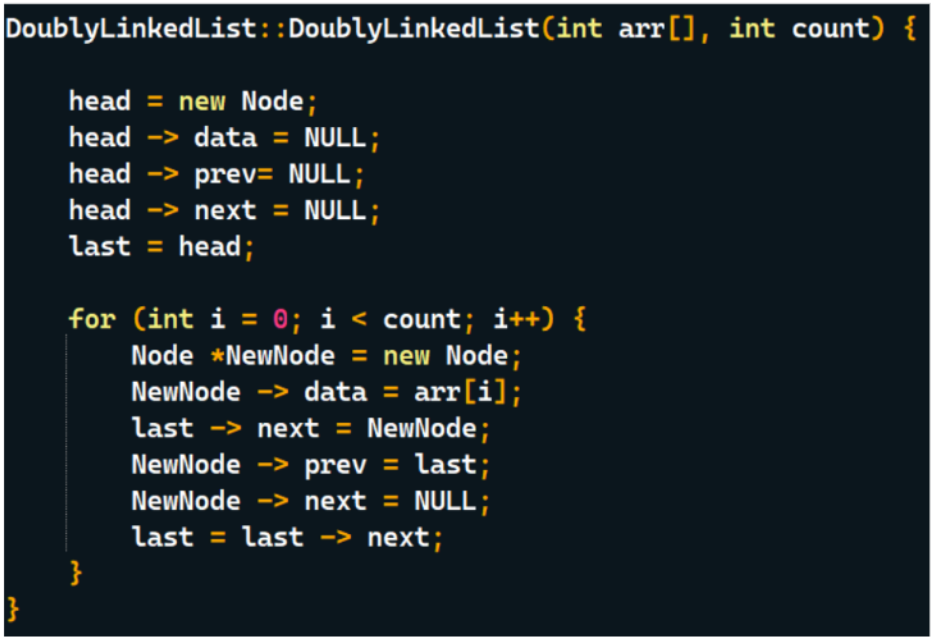
- **Create Doubly Linked List and manage their methods :**



```
class Node {
    public:
        int data;
        Node *prev;
        Node *next;
};

class DoublyLinkedList {
    private:
        Node *head, *last;
    public:
        DoublyLinkedList(int arr[], int count);
        void ForwardDisplay();
        void ReverseDisplay();
        bool isEmpty();
};
```

- **Insert data into Doubly Linked List :**

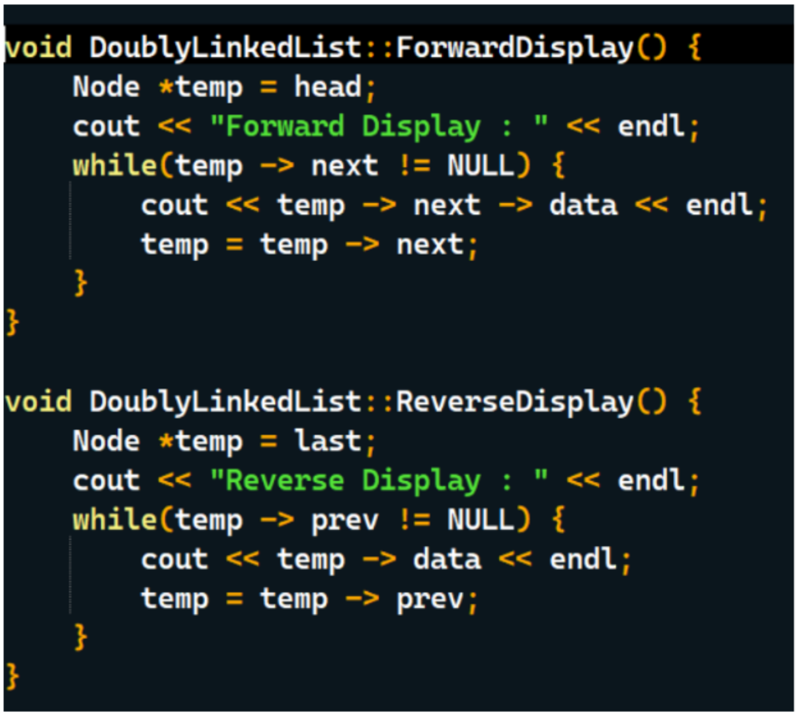


```
DoublyLinkedList::DoublyLinkedList(int arr[], int count) {

    head = new Node;
    head -> data = NULL;
    head -> prev= NULL;
    head -> next = NULL;
    last = head;

    for (int i = 0; i < count; i++) {
        Node *NewNode = new Node;
        NewNode -> data = arr[i];
        last -> next = NewNode;
        NewNode -> prev = last;
        NewNode -> next = NULL;
        last = last -> next;
    }
}
```

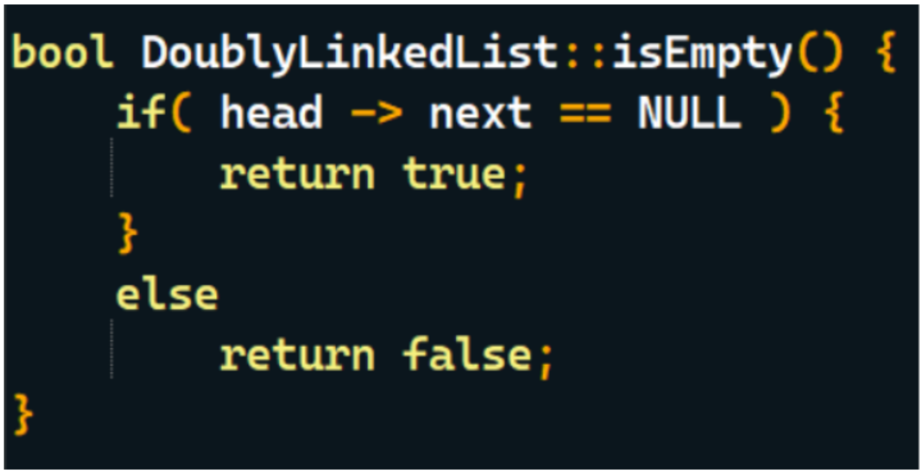
- **Forward Display & Reverse Display :**



```
void DoublyLinkedList::ForwardDisplay() {
    Node *temp = head;
    cout << "Forward Display : " << endl;
    while(temp -> next != NULL) {
        cout << temp -> next -> data << endl;
        temp = temp -> next;
    }
}

void DoublyLinkedList::ReverseDisplay() {
    Node *temp = last;
    cout << "Reverse Display : " << endl;
    while(temp -> prev != NULL) {
        cout << temp -> data << endl;
        temp = temp -> prev;
    }
}
```

- **Check Doubly Linked List is empty or not :**



```
bool DoublyLinkedList::isEmpty() {
    if( head -> next == NULL ) {
        return true;
    }
    else
        return false;
}
```