DSA SERIES

- Learn Coding

<u>Topic to be Covered today</u>

Linked List

LETS START TODAY'S LECTURE

Linked List

- It is a fundamental data structure.
- It mainly allows efficient insertion and deletion operation compared to arrays.
- The data are stored in non-contiguous manner.
- The elements are accessed sequentially with the help of pointers.

- Each element in the linked list are connected with the other elements .
- No indexing works here as we deal with the help of pointers.
- Each node contains the data and the pointer to reference the other element connected with that particular element.

Structure

It is a user defined data type in C++ that allows you to combine variables of different data types into a single unit.

It is like a custom data container .

Syntax :

```
struct  structureName {

        dataType1 variable;
        dataType2 variable;

};
```

Example :

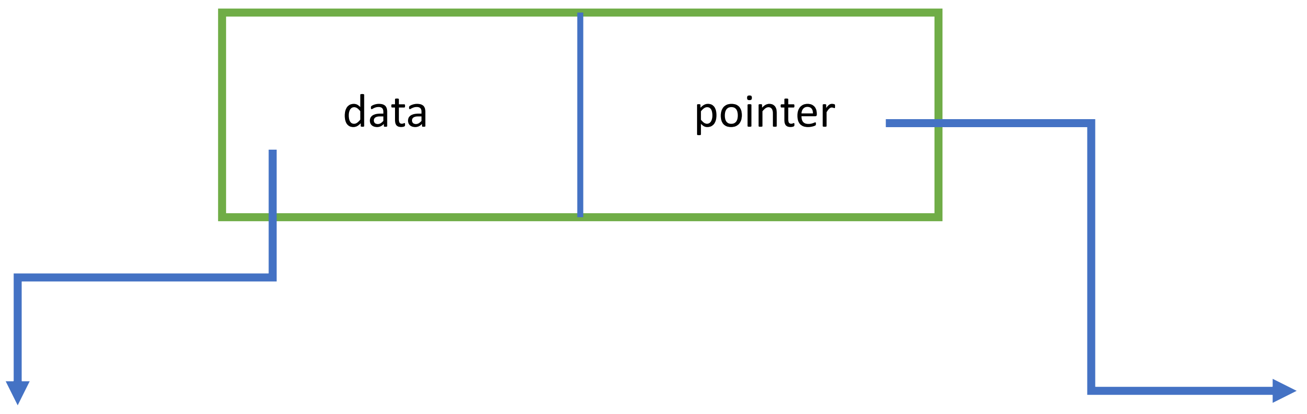
```
struct student {

        int rollNo;
        string name;
        float marks;
}


student  s1;
s1.rollNo = 68;
s1.name = “Ankit”;
s1.marks = 99;
```

Types of Linked List

1. **Singly Linked list**
2. **Doubly Linked list**
3. **Circular Linked list**

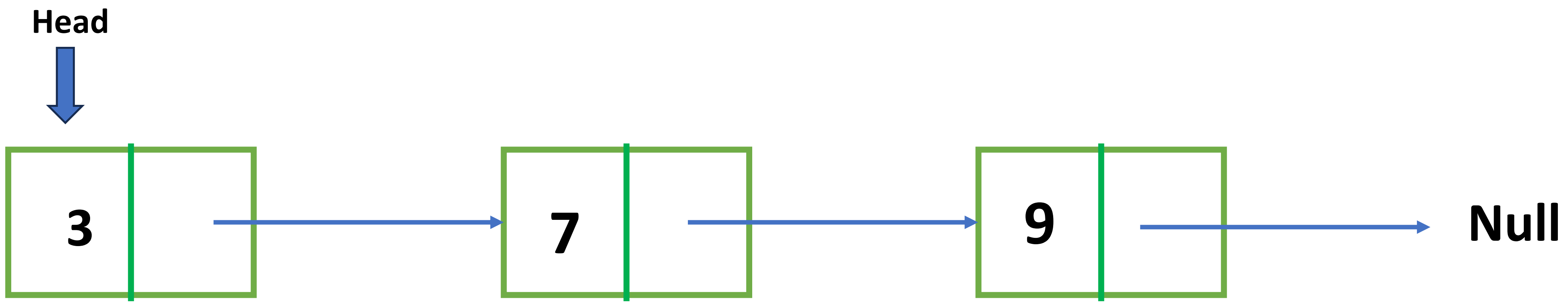
Singly Linked List

Each node contains data field and a reference to the next node in the linked list.



Reference to the next element

The value that need to be stored



1. The last element in the linked list points to the NULL.

2. We keep a head pointer to traversing the linked list .

3. The head pointer points to the first node of the linked list.

Implementation of Singly Linked list:

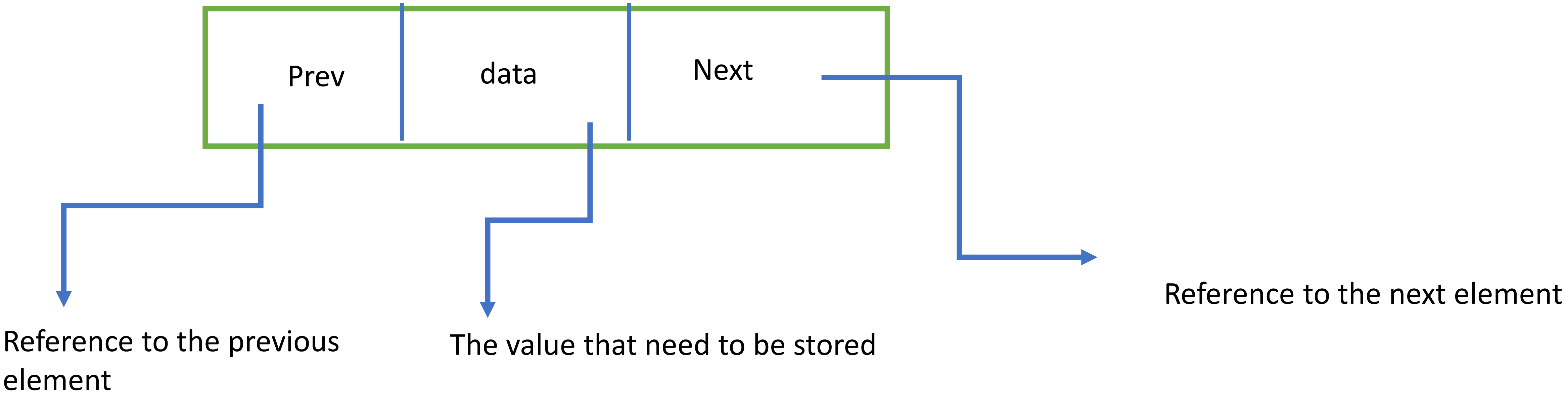
```
#include<iostream>
using namespace std;

// Create the sructure for the Node
// For creating the structure , we use the keyword (struct).

struct Node{
    int data ;
    Node* next;

    Node(int value){
        data = value;
        next= NULL;
    }

};
```

```
// Global variable that will be accessed throughout the code
Node* head = NULL;

// Insert the element at end
void insertAtEnd(int value){
    Node* newNode = new Node(value);

    //  if there was no element present before
    if(head==NULL){
        head = newNode;
        return;
    }

    Node* temp = head;
     while(temp->next != NULL){
        temp = temp->next;
    }
    temp->next = newNode;

}
```

```
void display(){
    Node* temp = head;

    while(temp!= NULL){
        cout<<temp->data<<" -> ";
        temp= temp->next;
    }
    cout<<"NULL";
}


int main(){
    insertAtEnd(10);
    insertAtEnd(20);
    insertAtEnd(1);
    insertAtEnd(8);

    display();

    return 0;
}
```

Doubly Linked List

Each node contains data field and a reference to the next node and the previous node in the linked list.


Prev
data
Next
Reference to the next element
Reference to the previous element
The value that need to be stored

Implementation of Doubly Linked list:

```
#include<iostream>
using namespace std;


struct Node {
    int data;
    Node* prev;
    Node* next;

    Node(int value){
        data = value;
        prev = NULL;
        next = NULL;
    }
};


Node* head = NULL;
```

```
void insertAtEnd(int value){
    Node* newNode = new Node(value);

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

    Node* temp = head;

    while(temp->next != NULL){
        temp = temp->next;
    }

    temp->next = newNode;
    newNode->prev = temp;
}
```

```
void displayForward(){
    Node* temp = head;
    while (temp!=NULL){
        cout<<temp->data << " <-> ";
        temp = temp ->next;
    }
    cout<<" NULL ";
}

void displayBackward(){
    if(head== NULL) return;

    // Go to the last node
    Node* temp = head;
    while(temp->next!=NULL){
        temp = temp->next;
    }

    while(temp!=NULL){
        cout<<temp->data<<" <-> ";
        temp = temp->prev;
    }
    cout<<" NULL ";
}
```

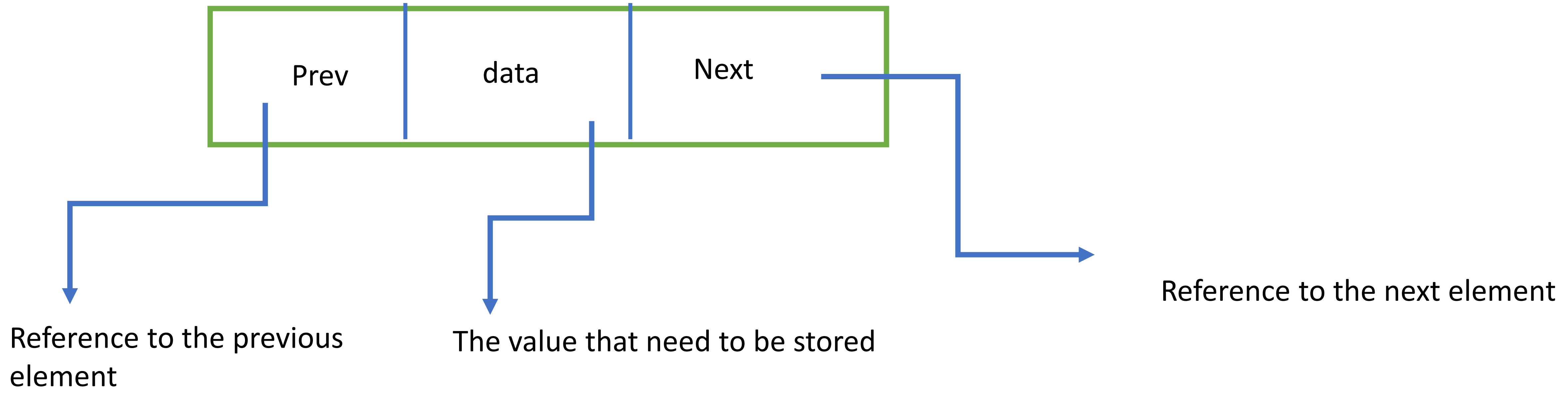
```
int main(){
    insertAtEnd(10);
    insertAtEnd(80);
    insertAtEnd(60);
    insertAtEnd(20);

    cout<<"Forward : ";
    displayForward();

    cout<<endl;

    cout<<"Backward : ";
    displayBackward();

    return 0;
}
```

Circular Linked List

Types:

1. **Singly circular linked list**
2. **Doubly circular linked list**

Last element -> first element


Prev
data
Next
Reference to the previous element
The value that need to be stored
Reference to the next element

Implementation of Circular Linked list:

```
#include<iostream>
using namespace std;

struct  Node
{
    int data;
    Node* next;

    Node(int value){
        data =value;
        next=NULL;
    }
};


Node* head = NULL;
```

```
void insertAtEnd(int value){
    Node* newNode = new Node(value);

    if(head == NULL){
        head = newNode;
        newNode->next = head;
        return ;
    }

    Node* temp = head;

    while(temp->next!=head){
        temp=temp->next;
    }

    temp -> next = newNode;
    newNode->next = head;

}
```

```
void display(){
    if(head == NULL) return;

    Node* temp = head;

    do{
        cout<<temp->data<<" -> ";
        temp = temp->next;

    }while(temp!=head);

    cout<<"(head)"<<endl;
}


int main(){

    insertAtEnd(10);
    insertAtEnd(20);
    insertAtEnd(30);
    insertAtEnd(40);

    display();

    return 0;
}
```

Learn coding

THANK YOU