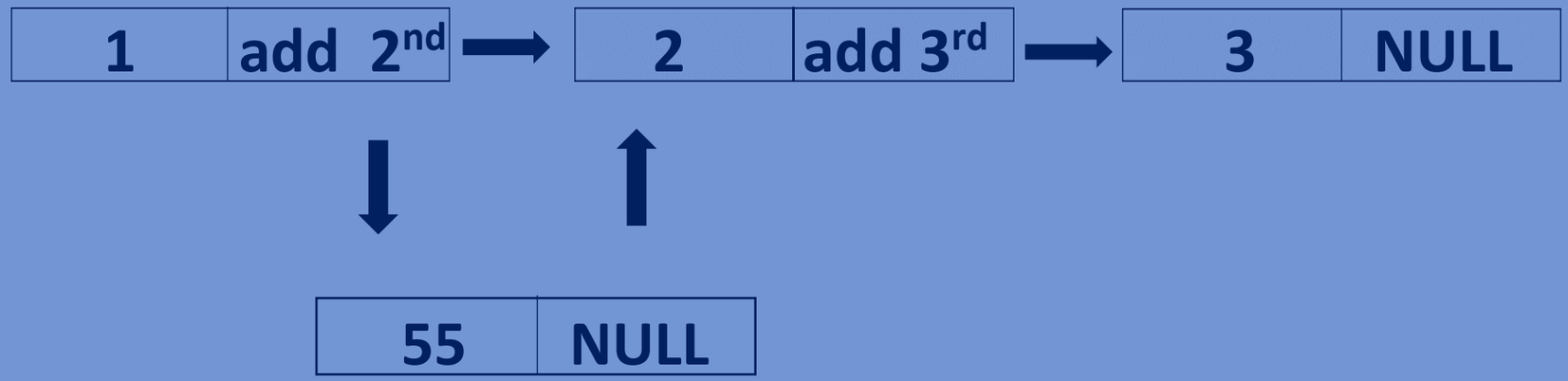
Data Structures

Inserting node at a particular position

inserting at a position:



- Create a new node.
- If linked list is empty make the new node as head.
- if position is equal to one,then point the next pointer of new node to the head node and make the new node as head.
- In other cases where the position is greater than 1,we will traverse the Linked list till position-1 th node and point the next of new node to next of position-1 th node and then point the next of position-1 th node to new node.
- Return the head(starting) node.

Function for inserting node at position:

```
lin_list *insertAtPosition(lin_list *head, int pos, int value)
{
    int i;
    lin_list *temp=head, *newnode=(lin_list*)malloc(sizeof(lin_list));
    newnode->data=value;
    newnode->next=NULL;
    if(head==NULL){
        head=newnode;
        return head;
    }
    if(pos==1){
        newnode->next=head;
        return newnode;
    }
    for(i=1; i<=pos-2; i++)
    {
        head = head->next;
    }
    newnode->next  = head->next;
    head->next = newnode;
    return temp;
}
```

Recursive function for printing the elements of a linked list:

- return type for the function is void.
- If linkedlist is NULL ,return nothing(void function).
- Print the data of the node.
- Recursively call the function with next node.

```
void PrintElements(lin_list *head){
    if(head==NULL){
        return;
    }
    printf("%d",head->data);
    PrintElements(head->next);
}
```

Whole program:

```c
#include<stdio.h>
#include<stdlib.h>
//creating a node.
typedef struct lin_list{
    int data;
    struct lin_list *next;
}lin_list;

lin_list *insertAtPosition(lin_list *head, int pos, int value)
{
    int i;
    lin_list *temp=head, *newnode=(lin_list*)malloc(sizeof(lin_list));
    newnode->data=value;
    newnode->next=NULL;
    if(head==NULL){
        head=newnode;
        return head;
    }
```

```
    if(pos==1){
        newnode->next=head;
        return newnode;
    }
    for(i=1; i<=pos-2; i++)
    {
        head = head->next;
    }
    newnode->next  = head->next;
    head->next = newnode;
    return temp;
}


void PrintElements(lin_list *head){
    if(head==NULL){
        return;
    }
    printf("%d ",head->data);
    PrintElements(head->next);
}

int main(){
    lin_list *head=NULL;
```

```
    head=insertAtPosition(head,0,1);
    PrintElements(head);printf("\n");
    head=insertAtPosition(head,1,2);
    PrintElements(head);printf("\n");
    head=insertAtPosition(head,2,3);
    PrintElements(head);printf("\n");
    head=insertAtPosition(head,3,55);
    PrintElements(head);
    return 0;
}
```

Output:

1

2 1

2 3 1

2 3 55 1