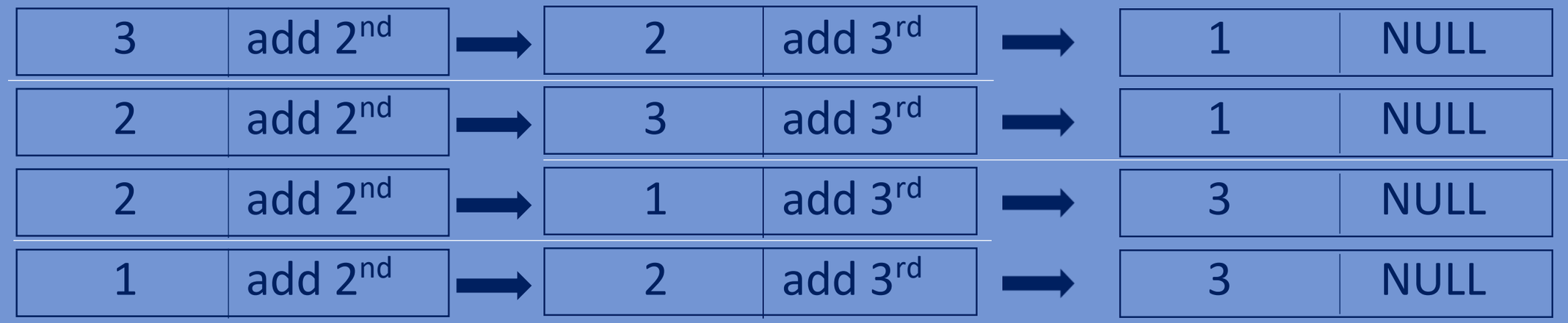
Data Structures

Sorting of a linked list

Sorting of a linked list:



- Bubble sort for sorting of the linked list.
- Traverse each node of linked list .
- Inside the while loop ,again start traversing from head node using another While loop till last but one node.
- Compare the values of data in consecutive nodes.
- After comparing swap the data between consecutive nodes if needed.
- Complexity is n^2 where n is the total no of nodes.

Function to sort the elements of a linked list.

- return type is void.

```
//sorting linkedlist using bubble sort
void sort(lin_list *head){
    lin_list *temp,*temp1;
    temp=head;
    while(temp){
        temp1=head;
        while(temp1->next!=NULL){
            if(temp1->data > temp1->next->data){
                swap(temp1,temp1->next);
            }
            temp1=temp1->next;
        }
        temp=temp->next;
    }
}
```

Swap function:

- swap data between the two nodes
- return type is void.

```
//swaping of data of two nodes.
void swap(lin_list *temp,lin_list *temp1){
    int tempo=temp->data;
    temp->data=temp1->data;
    temp1->data=tempo;
}
```

Whole program:

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

```c
//inserting nodes
lin_list *insertnode(lin_list *head,int data) {
    lin_list *newnode=(lin_list*)malloc(sizeof(lin_list));
    newnode->data=data;
    newnode->next=head;
    head=newnode;
    return head;
}
//printing the linked list.
void PrintElements(lin_list *head){
    //base condition
    if(head==NULL){
        return;
    }
    printf("%d ",head->data);
    PrintElements(head->next);
}

//swaping of data of two nodes.
void swap(lin_list *temp,lin_list *temp1){
    int tempo=temp->data;
    temp->data=temp1->data;
    temp1->data=tempo;
}
```

```c
//bubble sort of linked list
void sort(lin_list *head){
    lin_list *temp,*temp1;
    temp=head;
    while(temp){
        temp1=head;
        while(temp1->next!=NULL){
            if(temp1->data > temp1->next->data){
                swap(temp1,temp1->next);
            }
            temp1=temp1->next;
        }
        temp=temp->next;
    }
}
//main
int main(){
    lin_list *head=NULL;
    head=insertnode(head,1);
    head=insertnode(head,2);
    head=insertnode(head,3);
    head=insertnode(head,4);
    PrintElements(head);printf("\n");
```

```
    sort(head);
    PrintElements(head);
    return 0;
}
```

Output:

4 3 2 1

1 2 3 4