DATE PAGE

LAB 7

Stacks and Queues Implementation using linked lists

Q.

Ans

```
sort() {
    struct node *ptr = *head;
    struct node *temp = NULL;
    int i;

    if (head == NULL) return
    else {
        while (ptr != NULL) {
            temp = ptr -> next;
            while (temp != NULL) {
                if (ptr -> data > temp -> data) {
                    i = ptr -> data;
                    ptr -> data = temp -> data;
                    temp -> data = i;
                }
                temp = temp -> next;
            }
            ptr = ptr -> next;
        }
    }
}

concatenate (struct node *ptr1, struct node *ptr2) {
    if (ptr1 != NULL && ptr2 != NULL) {
        if (ptr1 -> next == NULL)
            ptr1 -> next = ptr2;
        else
            concatenate (ptr1 -> next, ptr2);
    }
}
```

DATE: PAGE:

```
    else {
        print ( Either ptr1 or ptr2 is Null);
    }
}

struct node *concat (struct node *start1, struct node *start2) {
    struct node *ptr;
    if (start1 == Null) {
        if start1 = start2;
        return
    if (start2 == Null) {
        return start1;  ptr = start1;
        while(ptr -> next != Null)
            ptr = ptr -> next;
            ptr -> next = start2;
            return start1;
        {

}

push (struct node **head, int new_data) {
    struct node *new_node = (struct node *) malloc (sizeof (struct node));
    new_node -> data = new_data;
    new_datnode -> next = *head;
    *head = new_node;

pop() {
    struct node *ptr;
    if (head == Null)
        printf("List Empty");
    else {
        ptr = head;
        head = ptr -> next;
        free(ptr);
    }
}
```

DATE: PAGE:

```
enqueue(item){
    struct node *ptr, *temp
    ptr = (struct node*)malloc(sizeof(struct node));
    ptr->data = item;
    ptr->next = NULL;

    if(head == NULL)
        head = ptr;
    else{
        temp = head;
        while(temp->next != NULL){
            temp = temp->next;
        }
        temp->next = ptr;
    }
}

dequeue(){
    struct node *ptr;
    if(head == NULL)
        print list Empty
    else
        ptr = head;
        head = ptr->next;
        free(ptr);
}
```