NAME : Ramya Ramesh
USN : 1BM19CH038

DS Lab Program 7

"Singly linked List Operations"

Pseudocode :

i) Concatenation :

```
void concatenate (struct node *a, struct node *b)
{ if (a!=NULL && b!=NULL)
  {
      if (a→next == NULL)
          a→next = b;
      else
          concatenate (a→next, b); // Recursion
  }
  else
  {
      printf ("Either a or b is NULL\n");
  }
}
```

```
struct node * concat (struct node *start1, struct node *start2)
{
    struct node * ptr;
    if (start1 == NULL)
    {
        start1 = start2;
        return start1;
    }
```

```
if (start2 == NULL)
        return start1;
ptr = start1;
while (ptr -> link != NULL)
        ptr = ptr -> link;
ptr -> link = start2;
return start1;
```

ii) SORTING :

ii)
```
struct node * current = head, * index = NULL;
int temp;
if (head == NULL)
  {
    return;
  }
else {
    while (current != NULL) {
      index = current -> next;
    while (index != NULL) {
      if (current -> data > index -> data) {
        temp = current -> data;
        current -> data = index -> data;
        index -> data = temp;
      }
    index = index -> next;
    }
current = current -> next;
}
}
}
```

iii) REVERSE :

```
void reverse (struct * node current) {
 if (head == NULL) {
     printf (" list is empty\n ");
      return;
    }
else {
     if (current → next == NULL) {
         printf (" %d" , current → data);
       return;
     }

  reverse (current → next);          // Recursion
  printf(" %d" , current → data);
 }
}
```

iv) STACK IMPLEMENTATION :

```
void push(struct Node ** head_ref, int new_data)
{
   struct Node * new_node = (struct Node *) malloc
       (sizeof (struct Node));
  new_node → data = new_data;
  new_node → next = (* head_ref);
   (*head_ref) = new_node;
}

void Pop()
{
   struct node *ptr;
   if(head == NULL)
     {
        printf ("\n List is empty ");
     }
```

```
else
{
    ptr = head;
    head = ptr -> next;
    free(ptr);
    printf("\n Node deleted from the beginning");
}
}
```

v) QUEUE IMPLEMENTATION:

```
void Enqueue(item)
{
    struct node *ptr, *temp;
    ptr = (struct node *) malloc (sizeof(struct node));
    ptr -> data = item;
    ptr -> next = NULL;
    if (head == NULL)
    {
        head = ptr;
        printf("\n Node is inserted");
    }
else
    {
        temp = head;
        while (temp -> next != NULL)
        {
            temp = temp -> next;
        }
        temp -> next = ptr;
        printf("\n Node inserted");
    }
}
```

Date: YOUVA

```
Void Dequeue()
{
    struct node *ptr;
    if(head == NULL)
    {
        printf("\n List is empty");
    }
    else
    {
        ptr = head;
        head = ptr -> next;
        free(ptr);
        printf("\n Node deleted from beginning");
    }
}
```