12/12/20 Lab 7, Sudeshna Bhushan, 1BM19CS189

→ First we need to create a node
→ One more function to create node 2 to concatenate 2 strings.
→ Two display function for node 1 and node 2.

a) Sort the linked list

```
if (head == NULL)
  return;
do
{
   swapped=0;
   ptr1=head;
   while (ptr1->next != lptr)
   {
      if (ptr1->sem > ptr1->next->sem)
      {
          int temp = ptr1->sem;
          ptr1->sem = ptr1->next->sem;
          ptr1->next->sem = temp;
          swapped =1;
      }
      ptr1 =ptr1->next;
   }
   lptr = ptr1;
}
while (swapped);
}
```

Date ________
Page ________

b) <u>Reverse the linked list</u>

```
void reverse ()
{
  struct node * prev = NULL;
  struct node* current = head;
  struct node*  next  = NULL;
  while (current != NULL)
   {
     next=current -> next;
     current-> next = prev;
     prev =current;
     current =next;
   }
   head=prev;
}
```

c) <u>Concatenation of two linked list</u>

```
void concat ()
{
   struct node *ptr;
   if ( head == NULL)
   {
       head =head2;
   }
   if (head2 = = NULL)
   {     head2=head
   }
```

Scanned with CamScanner

```
ptr=head;
while (ptr->next != NULL)
        ptr=ptr->next;
  ptr->next =head2;
}
```

d) <u>Using stacks</u>

1. Create a node first and allocate memory to it.

2. If the list is empty then the item is to be pushed as the start node of the list. This includes assigning values to the data part of the node and assign null to the address part of the node.

3. If there are some nodes in the list already, then we have to add the new element in the beginning of the list. For this purpose, assign the address of the starting element to the address field of the new node and make the new node, the starting node of the list.