1BM19CS216 7/12/2020

Lab 8: Single link list to Sort, Reverse & Concatenate implementation of Stack & Queue using LL

Algorithm

Sort()

```
Reverse()
while (current != NULL)
{
 next = current->next;
 current->next = prev;
  prev = current;
  current = next;
}
*head_ref = prev.
```

Concatenate

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

Queue

```
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 inserted");
}
else
    temp = head;
while (temp->next != NULL)
{
    temp = temp->next;
}
  temp->next = ptr;
printf (" Node is inserted ");
}
}
```

```
void Dequeue()
{
struct node *ptr;
if (head == NULL)
{
  ptr("In list is empty");
}
else
{
  ptr = head;
head = ptr->next;
free (ptr);
printf("In Node deleted from beg..");
}
}
```

Stack.

```
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)
{
  print ("In list is empty");
}
else.
{
    ptr = head;
    head = ptr -> next;
    free (ptr);
}
}
```