ASSIGNMENT NO:8 **DATE:20/03/2015**

PROGRAM TITLE:Implement Queue using Linked list.

THEORY: Each node of a single linked list holds the address of only the next node in the linked list. The simple queue has 2 pointers front and rear. The insert operation occurs in the rear and the items are deleted from the front. Queue follows the FIFO(First In First Out) logic.

PROGRAM ALGORITHM:

```
Algo_insert(front,rear,item)
{
      allocate temp node with item;
      if(inter=NULL)
      {
            print overflow; //This only occurs when the memory is full
            return;
      }
      if(rear=NULL) //i.e when the queue is empty
      {
            set front and rear both as inter;
      }
      else
      {
            next(rear)=inter;
            rear=inter;
      }
      return;
}

Algo_delete(front,rear)
{
      if (front=NULL)
      {
            print underflow;
            return;
      }
      else if(front points to the only node that exists)
      {
            reset front and rear;
      }
      else
      {
            front=next(front);
      }
      return;
}
```

PROGRAM CODE:

```
/*C Program to Implement a Queue using Linked List*/
#include <stdio.h>
#include <stdlib.h>
struct Node
```

```
{
      int data;
      struct Node *next;
};
typedef struct Node *NODEPTR;
NODEPTR allocate_node(int item);//Allocates memory space for a new node
int freenode(NODEPTR p);//Deallocates memory space
int insert(NODEPTR *front,NODEPTR *rear,int item);//creates a new node at
rear, can also create start node if NULL is sent
int delete(NODEPTR *front,NODEPTR *rear);//deletes the node at front
//*************************************************************MAIN
FUNCTION*****************************************************************
int main()
{
      NODEPTR front=NULL,rear=NULL;
      int ch=0,tmp;
     system("clear");
      while(ch!=3)
     {
            printf("\n\tMenu::\n\t1.Insert\n\t2.Delete\n\t3.Exit\n\tYour
choice:: ");
         scanf("%d",&ch);
         switch(ch)
         {
            case 1: printf("\n\tEnter data item:: ");
                      scanf("%d",&tmp);
                      insert(&front,&rear,tmp);
                      break;
              case 2: tmp=delete(&front,&rear);
                  if(tmp!=-9999)
                      printf("\n\tThe item deleted is:: %d",tmp);
                      break;
              case 3: printf("\n\tProgram Terminated\n");
                      exit(0);
                      break;
              default:printf("\n\tIncorrect value entered. Enter choice
again");
         }
      }
      return 0;
}
//*************************************************************MEMBER
FUNCTIONS*****************************************************************
NODEPTR allocate_node(int item)
{
      NODEPTR temp = (NODEPTR)malloc(sizeof(struct Node));
      temp->data=item;
      temp->next=NULL;
      return temp;
}
int freenode(NODEPTR p)
{
     free(p);
     p=NULL;
     return 0;
```

```
}
int insert(NODEPTR *front,NODEPTR *rear,int item)
{
    NODEPTR inter=allocate_node(item);
     if(inter==NULL)
     {
          printf("\n\tMemory couldnt be allocated from the heap.
Overflow occurs.");
          return 1;
    }
    if(*rear==NULL)
     {
          *front=*rear=inter;
     }
     else
     {
          (*rear)->next=inter;
          *rear=inter;
     }
     return 0;
}
int delete(NODEPTR *front,NODEPTR *rear)
{
    NODEPTR p = *front;
    if(*front==NULL)
    {
     printf("\n\tThe queue is empty. Underflow occurs.");
     return -9999;
    }
    else if((*front)->next==NULL)
    {
     *front=*rear=NULL;
    }
    else
    {
     *front=(*front)->next;
    }
    int item =p->data;
    freenode(p);
    return item;
}
```

OUTPUT:

```
     Menu::
     1.Insert
     2.Delete
     3.Exit
     Your choice:: 1

     Enter data item:: 10

     Menu::
     1.Insert
     2.Delete
```

```
3.Exit
Your choice:: 1

Enter data item:: 20

Menu::
1.Insert
2.Delete
3.Exit
Your choice:: 2

The item deleted is:: 10
Menu::
1.Insert
2.Delete
3.Exit
Your choice:: 2

The item deleted is:: 20
Menu::
1.Insert
2.Delete
3.Exit
Your choice:: 2

The queue is empty. Underflow occurs.
Menu::
1.Insert
2.Delete
3.Exit
Your choice:: 3

Program Terminated
```

DISCUSSION:

1.The complexity of insertion is O(1).
2.The complexity of deletion is O(1).
3.Overflow occurs only when the program cannot be allocated any more memory by the system.