Page No.
Date

NAME: ANSHUL H. SUBANA
USN: 1BM19CS020

LAB-8

Q-8) WAP to implement Stack & Queues using Linked Representation.

→ Stack

```
#include <stdio.h>
#include <stdlib.h>
struct node *top= NULL;
 struct node
{     int data;
      struct node *next;
};
void stack();
void push(int);
int pop();
 void display();
 int main()
{     stack();
      return 1;
}
void stack()
{ int choice=0, ele=0;
  do
{ printf("\n Enter the choice: \n 1. Push.
  \n 2. Pop \n 3. Display \n 4. Exit \n Cho
scanf("%d", &choice);
switch(choice)
{ case 1: printf("\n Enter the element to Push
          scanf("%d", &ele);
          push(ele);
```


Anshul H. Subana

NAME : ANSHUL H. SURANA
USN : 1BM19CS020

```
        break;
case 2: ele = pop();
        printf("\n %d was deleted from Queue", ele);
        break;
case 3: display();
        break;
case 4: exit(0);
default:
        printf("\n Input Error Try Again");
        Stack();
    }
}while(1);
}
void push (int ele)
{ struct node* newnode;
  newnode = (struct node*) malloc (sizeof (struct node));
  newnode -> data = ele;
  newnode -> next = NULL;
  if (top == NULL)
{ top = newnode; }
  else
{ newnode -> next = top;
    top = newnode;
}
printf ("\n element %d was inserted \n", ele);
}
int pop ()
{ int ele;
  struct node * temp;
  if (top == NULL)
{ printf("\n Stack Underflow, Stack is empty");
    Stack(); }
```


Anshul H. Surana

```
ele = top -> data;
temp = top;

if (top -> next == NULL)
{   top = NULL; }
else
{   top = top -> next; }
free (temp);
return ele;
}
void display()
{   struct node * i;
    if (top == NULL)
  { printf("\n Stack is empty \n");
    stack();
  }
Printf("\n Stack Contains: \n Top ->");
for (i = top; i != NULL; i = i -> next)
    printf("\t %d", i -> data);
}
}
```

Queue.

```
#include <stdio.h>
#include <stdlib.h>
struct node
{ int data;
  struct node * next; };
void insert();
void display();
void del();
```


Anshul H. Supana

NAME : ANSHUL H. SUPANA
USN : 1BM19CS020

```
struct node * real = NULL, * front = NULL;

int main (int argc, char ** argv)
{   int choice;
    char ch = 'y';
do
{ printf("\n Queue implementation using linked list\n");
   printf("\n1. Create \n2. Display \n3. Delete
   \n4. Exit \n");
printf("\n Enter your choice");
scanf("%d", &choice);
switch (choice)
{ case 1: insert(); break;
   case 2: display(); break;
   case 3: del(); break;
   case 4:
      ch = 'n';
      break;
}
}while (ch=='y' || ch=='Y');
}
void insert()
{  struct node * newnode;
   newnode = (struct node *) malloc (sizeof(struct node));
   printf("\n Enter the element: \n");
   scanf("%d", &newnode->data);
   newnode->next = NULL;

   if (real == NULL)
{  real = newnode;
   front = newnode;
}
```


Anshul H. Supana

```
else
{ rear->next = newnode;
    rear = newnode;
}
}
void del()
{ if (front == NULL)
  { printf("Queue is empty\n"); return;
  }
  else
{ printf("Deleted ele is %d", front->data);
   if (front == rear)
   { printf("Queue is empty\n");
     front = NULL;      rear = NULL;
   }
   else
     front = front->next;
   }
}
void display()
{
   struct node *temp;
   if (front == NULL)
  {  printf("Queue is empty");
       return;
  }
   temp = front;
   while (temp != NULL)
  { printf("%d", temp->data);
    temp = temp->next;
  }
}
```


Anshul H. Surana