DATE
PAGE

Implementation of stacks & queues using linked list

Program :-

```
#include <stdio.h>
#include <stdlib.h>

typedef struct node {
    int info;
    struct node *link;
    }
    Node;

Node *temp1, *temp2;
    int n, i;
printf("Enter number of nodes:\n");
 scanf("%d", &n);
  for(i=0; i<n; i++)
  {
      temp1 = (Node*) malloc (sizeof(Node));
        printf("Enter value for node %d\n", i+1);
         scanf("%d", &temp1->info);
              temp1->link = NULL;
               if(start == NULL)
                start = temp1;
               else
```

```
{
temp2 = start;
while (temp2 -> link != NULL)
  temp2 = temp2 -> link;
  temp2 -> link = temp1;
  }
  }
  return start;
}

void disp (Node *start)
{
  Node *t;
  printf ("\n Elements of the list : \n");
  for (t = start; t != NULL; t = t -> link)
  {
    if (t -> link != NULL)
      printf ("%d", t -> info);
      else
        printf ("%d", t -> info);
      }
      {
      void push ();
      void pop();
      void display_s();

      struct node *head;
```

```
void push () {
 int info;
 Node *ptr = (Node*) malloc (sizeof (Node));
  if (ptr == NULL)
 {
 printf (" Stack empty !.\n");
  scanf ("%d", &info);
  if (head == NULL)
  {
     ptr -> info = info;
      ptr -> link = NULL;
      head = ptr;
    }
    else
     {
        ptr -> info = info;
        ptr -> link = head;
          head = ptr;
      }
    }
  }


 void pop (){
    int item;
     Node *ptr;
     if (head == NULL)
     {
```

```
        printf ("Stack Underflow\n");
        }
    else
    {
     item = head -> info ;
      ptr = head;
      head = head -> link;
      free (ptr);
       printf ("Item popped\n");
     }
}

void display S(){
 int i;
 Node *ptr;
 ptr = head;
 if (ptr == NULL) {
  printf(" Stack empty \n");
 }
else
{
 printf (" Elements are \n");
  while (ptr != NULL)
   {
   printf ("%od", ptr->info);
      ptr = ptr -> link;
    }
  }
}
```

```
void enQueue ();
void deQueue ();
void display_Q();
struct node *front;
struct node *rear;

void enQueue (){
Node *ptr;
int item;
ptr =(Node*) malloc (sizeof (Node));
if (ptr == NULL){
printf(" Queue Overflow \n");
return;
}
else
{
printf (" Enter value \n");
scanf(" %d", &item);
ptr->info = item;
if(front == NULL)
{
front = ptr;
rear = ptr;
front->link = NULL;
rear->link = NULL;
}
else
{
```

```
rear->link = ptr;
rear = ptr;
rear->link = NULL;
}
}
}
void dequeue() {
Node *ptr;
if ( front == NULL)
{
printf ("Queue underflow \n");
return;
}
else
{
ptr = front;
front = front->link;
free (ptr);
printf("Item deleted \n");
}
}

void display_Q() {
Node *ptr;
ptr = front;
if ( front == NULL)
{
printf("Queue empty \n");
}
```

```
else
{
printf(" The queue elements are :\n");
        while (ptr1 = NULL)
        {
        printf("%d", ptr -> info) ;
            ptr = ptr -> link;
         }
      }
     }

int main () {
    Node *s1, *s2;
     int ch;
     s1 = NULL;
     s2 = NULL;
     while (1) {
     printf("\n1. create \n2. Stack (push) \n 3. Stack (Pop) \n4. Stack (display) \n5. Enqueue \n 6. Dequeue \n 7. Queue (Display) \n 8. Exit \n");
     printf(" Enter choice \n");
     scanf("%d", &ch);
     switch (ch)
     {
       case 1: s1 = create (s1);
               break;
       case 2: push();
               break;
```

```
case 3: pop();
        break;
case 4: display_S();
        break;
case 5: enQueue();
        break;
case 6: deQueue();
        break;
case 7: display_Q();
        break;
case 8: exit(0);
        break;
default: printf("Wrong choice!");
        break;
}
}
}
```

Expected output :-

1. Create
2. Stack (push)
3. Stack (pop)
4. Stack (display)
5. Enqueue
6. dequeue
7. Queue (display)
8. Exit

Enter choice :
1

Enter the value :
8

Enter choice :
1

Enter value :
0

Enter choice :
2

Enter value :
3

```
Enter choice :
4
The stack elements are :
3 0 8

Enter choice :
3
Item popped

Enter choice :
4
The stack elements are :
0 8

Enter choice :
5
Enter value
4

Enter choice :
5
Enter value
7

Enter choice :
5
Enter value :
6
```

Enter choice :

7

The queue elements are :

4 5 7 6

Enter choice :

6

Item deleted

Enter choice :

7

The queue elements are :

5 7 6