{ IMPLIMENT Queue using linked list }

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

Sicuct node {
    int value ;
    Struct node * rext ;
    };
NODE getnode() {
 NODE new_node = (NODE) malloc (sizeof(struct node));
    if (new_node == NULL) {
     printf ("Memory allocation failed. \n");
      exit (1) ;
     }
     return new_node;
 }

Void display(NODE first) {
 NODE temp;
if (first == NULL) {
     printf ("Linked list is empty \n")
     return;
}
  temp = first;
while (temp != NULL) {
    print ("%d", temp -> value);
    temp = temp -> next;
}
  printf ("NULL \n");
}
```

```
NODE insert_beg (int item, NODE first) {
 NODE new = getnode();
   new -> Value = item;
   new -> next  = first;
   return new;
}
NODE delete_end (NODE first) {
   if (first == NULL) {
    printf ("LL is empty \n");
    return NULL;
   }
   if (first -> next == NULL) {
       free (first);
       return NULL;
  }
NODE prev = NULL, last = first;
while (last -> next != NULL) {
     prev = last;
     last = last -> next;
}
     prev -> next = NULL;
      free (last);
      return first;
}
```

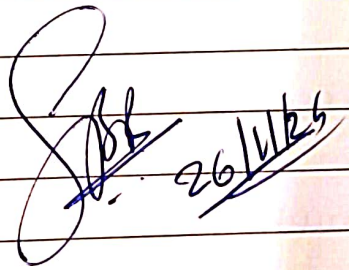
```
int main() {
int choice, item;
NODE first = NULL;
NODE item_del;
while(1) {
printf("enter you choice :\n1 insert \n2. delete
                \n3 display \n4. exit");
scanf("%d", &choice);

switch (choice) {
case 1:
    printf("enter item to insert at begining\n");
    scanf("%d", &item);
    first = insert_beg(item, first);
    break;

case 2:
    first = delete_end(first);
    break;

case 3:
    printf("the linked list is being displayed:\n");
    display(first);
    break;

case 4:
    exit(0);

default:
    printf("wrong choice\n");
    exit(0);
}
}
return 0; }
```

Output:

Enter your choice

1. INSERT
2. DELETE
3. DISPLAY
4. EXIT.

enter item to insert : 5
enter your choice : 1
enter item to insert : 10
enter your choice : 1
enter item to insert : 15
enter your choice : 2.
element 5 has been poped
enter your choice : 3
10 15
enter your choice 4.

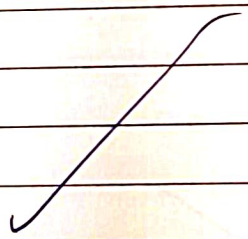


{ implement stack using linked list }

```
#include <stdio.h>
#include <stdlib.h>
struct node {
  int value;
  struct node * next;
  };
type def struct node * NODE;

NODE get node (){
  NODE new_node = (NODE) malloc (size of (struct node));
  if (new_node == NULL){
   printf("memory allocation failed \n");
   exit(1);
  }
   return new_node;
}
void display (NODE first ){
  if (first == NULL){
  printf ("LL is empty");
  return;
  }
  NODE temp = first;
  while (temp != NULL) {
   printf ("%d", temp -> value );
    temp = temp -> next;
  }
  printf ("\n");
}
```

```
NODE insert_beg (int item , NODE first ) {
NODE new = getnode ( );
new -> value = item;
new -> next = first;
return new;
}

NODE delete_first (NODE first ) {
if ( first = NULL ) {
  printf ("LL is empty , nothing to delete \n ");
  return NULL;
}
NODE temp = first;
first = first -> next;
printf ("item deleted : %d \n ", temp->value );
free (temp);
return first;
}

int main() {
int choice, item;
NODE first = NULL;

while (1) {
printf ("enter you choice : \n1. push \n2. pop \n3 display \n4 Exit \n");
scanf ("%d", &choice);
```

```
switch (choice) {
case 1:
    printf ("enter item to push : ")
    scanf ("%d", &item);
    first = insert_beg (item, first);
    break;
case 2:
    first = delete_first (first);
    break;
case 3:
    printf ("the stack is being displayed : ");
    display (first);
    break;
case 4:
    exit (0);
default:
    printf ("invalid choice \n");
}
}
return 0;
}
```



Enter your choice :
1. push
2. pop
3. display
4. Exit

1.
Enter item to push 10
Enter your choice :
1. push
2. pop
3. display
4. Exit.

1
enter item push 20
enter choice 1
enter item push 30.

3.
30 20 10

2.
~~30~~ 20 10

4.

20/11/25

exit.