OS LAB-11

```
#include<stdio.h>
struct node {
  int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode() {
  NODE x;
  x = (NODE) malloc (sizeof(struct node));
  if (x == NULL) {
    pf("mem full");
    exit(0);
  }
  return 0;
}
NODE concat (NODE first, NODE second) {
  NODE cur;
  if (first == NULL)
    return second;
  if (second == NULL)
    return first;
  cur = first;
```

```
while (cur->link != NULL)
        cur = cur->link;
        cur->link = second;
        return first;
}
NODE reverse (NODE first) {
    NODE cur, temp;
    cur = NULL;
    while (first != NULL) {
      temp = first;
      first = first->link
      temp->link = cur;
      cur = temp;
    } return cur;
}
NODE sort (NODE first) {
    int swapped;
    NODE ptr1, lptr = NULL;
    if (first == NULL)
      return NULL;
    do {
      swapped = 0;
      ptr1 = first;
      while (ptr1->link != lptr) {
```

Name of the Experiment :
Experiment No. :
Date :
Page No. :

```
if (ptr1->info > ptr1->link->info){
    int item = ptr1->info;
    ptr1->info = ptr1->link->info;
    ptr1->link->info = item;
    swapped = 1;
    }
    ptr1 = ptr1->link;
} lptr = ptr1;
} while (swapped);
return first;
}
NODE delete_rear (NODE first){
  NODE cur, prev;
  if (first == NULL){
   pf("empty");
   return first; }
  if (first->link == NULL){
    pf("deleted %d", first->info);
    free (first);
    return NULL;
  } prev = NULL;
    cur = first;
  while (cur->link != NULL){
    prev = cur;
    cur = cur->link; }
```

Signature.....

Name of the Experiment :
Experiment No. :
Date :
Page No. :

```
pf("deleted %d", cur->info);
free(cur);
prev->link = NULL;
return first; }
NODE delete_front(NODE first){
  NODE temp;
  if(first == NULL){
    pf("empty");
    return first; }
  temp = first;
  temp = temp->link;
  pf("deleted %d", first->info);
  free(first);
  return temp;
}
NODE insert_front(NODE first, int item){
    NODE temp;
    temp = getnode();
    temp->info = item;
    temp->link = NULL;
    if(first == NULL)
        return temp;
    temp->link = first;
    first = temp;
    return first; }
```


Signature.....

Name of the Experiment :
Experiment No. :
Date :
Page No. :

```
void main(){
  int item, choice, pos, i, n;
  NODE first = NULL, a, b;
  for(;;){
  pf("1 concat 2 reverse 3 sort 4 stack 5 queue\n");
  pf("enter choice");
  scanf("%d", &choice);
  switch(choice){
  case 1:
    pf("enter no of nodes in 1");
    scanf("%d", &n);
    a = NULL;
    for(i=0; i<n; i++){
      pf("enter item");
      sf("%d", &item);
      a = insert_front(a, item); }
    pf("enter no of nodes in 2");
    sf("%d", &n);
    b = NULL;
    for(i=0; i<n; i++){
      pf("enter item");
      sf("%d", &item);
      b = insert_front(b, item); }
    a = concat(a, b);
    display(a); break;
```

Name of the Experiment :
Experiment No. :
Date :
Page No. :

```
case 2: Rear = reverse(Rear); break;
case 3: Rear = sort(Rear); break;
case 4: for(;;) {
    pf("1: Insert 2: delete 3: display ");
    scanf("%d", &choice);
    switch(choice) {
    case 1: sf("%d", &item);
            Rear = Insert_front(Rear, item); break;
    case 2: Rear = delete_front(Rear); break;
    case 3: display(Rear); break;
    default: exit(0); } }
case 5: for(;;) {
    pf("1: Insert 2: delete 3: display ");
    sf("%d", &choice);
    switch(choice) {
    case 1: sf("%d", &item);
            Rear = Insert_front(Rear, item); break;
    case 2: Rear = delete_rear(Rear); break;
    case 3: display(Rear); break;
    default: exit(0);
    } }
}
}
}
```

Signature.....