4/12/20

LAB PROGRAM-9

NAME : SAI PRAVEEN MARNI
USN : 1BM19CS138

```
#include <stdio.h>
#include <stdlib.h>
#include <process.h>
struct node
{
   int info;
   struct node *rlink;
   struct node *llink;
};
   typedef struct node *NODE;
    NODE getnode()
{
   NODE x;
   x = (NODE) malloc(sizeof(struct node));
   if (x==NULL)
{
   printf("mem full \n");
      exit(0);
}
   return x;
}
void freenode(NODE x)
{
   free(x);
}
NODE insert_rear(NODE head, int item)
{
      NODE temp, cur;
      temp = getnode();
      temp->rlink = NULL;
      temp->llink = NULL;
      temp->info = item;
      cur = head->llink;
```

```
temp->llink = cur;
cur->rlink = temp;
head->llink = temp;
temp->rlink = head;
head->info = head->info + 1;
return head;
}
NODE insert_leftpos(int item, NODE head)
{
NODE temp, cur, prev;
if(head->rlink==head)
{
printf("list empty\n");
return head;
}
cur=head->rlink;
while(cur!=head)
{
if(item==cur->info) break;
cur=cur->rlink;
}
if(cur==head)
{
printf("key not found\n");
return head;
}
prev=cur->llink;
printf("Enter no.of nodes left of %d=", item);
temp=getnode();
scanf("%d", &temp->info);
prev->rlink=temp;
temp->llink=prev;
cur->llink=temp;
temp->rlink=cur;
return head;
}
NODE delete_all_key(int item, NODE head)
```

```
{
    NODE prev, cur, next;
    int count;
    if (head->rlink == head)
    {
        printf ("List Empty");
        return head;
    }
    count = 0;
    cur = head->rlink;
    while (cur != head)
    {
        if (item != cur->info)
            cur = cur->rlink;
        else
        {
            count++;
            prev = cur->llink;
            next = cur->rlink;
            prev->rlink = next;
            next->llink = prev;
            freenode(cur);
            cur = next;
        }
    }
    if (count == 0)
        printf ("Key not found");
    else
        printf ("Key found at %d positions and are deleted\n",
                count);
    return head;
}
NODE ddelete_rear(NODE head)
{
    NODE cur, prev;
    if (head->rlink == head)
    {
        printf("List is empty\n");
        return head;
    }
```

```
        cur = head->llink;
        prev = cur->llink;
        head->llink = prev;
        prev->rlink = head;
printf("the node deleted is %d", cur->info);
    freenode(cur);
    return head;
}
    void display(NODE head)
    {
      NODE temp;
      if(head->rlink == head)
      {
        printf("list empty\n");
        return;
      }
    for(temp = head->rlink; temp != head; temp = temp->rlink)
      printf("%d", temp->info);
    }
      void main()
    {
      int item, choice, key;
        NODE head, ten;
        head = getnode();
      head->rlink = head;
      head->llink = head;
    for(;;)
{ printf("\n1. insert_real  2. insert_key  3. display  4. delete key
        5. delete_real  6. exit\n");
    printf("Enter the choice:");
      scanf("%d", &choice);
      switch(choice);
    {
      case 1: printf("Enter the item:");
              scanf("%d", &item);
```

```
        head = insert_rear (head, item);
            break;
    case 2: printf ("Enter the key item:");
            scanf ("%d", &item);
        head = insert_leftpos (item, head);
            break;
    case 3: display (head);
            break;
    case 4: printf ("Enter the key item:");
            scanf ("%d", &item);
        head = delete_all_key (item, head);
            break;
    case 5: head = ddelete_rear (head);
            break;
    case 6:
    default: exit (0);
            break;
    }
  }
}
```

```
1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 1
enter the item : 12

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 1
enter the item : 23

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 2
enter the key item : 12
enter towards left of 12=11

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 3
11
12
23

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 2
enter the key item : 11
enter towards left of 11=10

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 3
10
11
12
23

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 4
enter the key item : 23
key found at 1 positions and are deleted

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 3
10
11
12

1.insert_rear  2.insert_key  3.display  4.delete key  5.delete_rear
 6.exit
enter the choice : 6

[Program finished]
```